元编程库

C++ 提供元编程设施，诸如类型特性、编译时有理数算术，以及编译时整数序列。

类型特性

类型特性定义编译时基于模板的接口，以查询类型的属性。

试图特化在标头 <type_traits> 定义且在本页列出的模板会导致未定义行为，除了可以按描述要求特化 std::common_type 和 std::basic_common_reference (C++20 起)。

除非另有说明，可以用不完整类型实例化在标头 <type_traits> 定义的模板，尽管通常禁止以不完整类型实例化标准库模板。

基类

大部分非变换类型特性需要无歧义地从 std::integeral_constant 公开派生，以满足*一元类型特性 (UnaryTypeTrait)* 或*二元类型特性 (BinaryTypeTrait)* 的要求。

在标头 <type_traits> 定义

integral_constant (C++11)	具有指定值的指定类型的编译期常量 (类模板)
bool_constant (C++17)	

标准提供 std::integral_constant 对类型 bool 的两个特化：

在标头 <type_traits> 定义

类型	定义
true_type	std::integral_constant<bool, true>
false_type	std::integral_constant<bool, false>

一元类型特性

一元类型特性可以用于在编译时查询类型的布尔属性。

所有这些类型特性都满足*一元类型特性 (UnaryTypeTrait)*，每个类型特性的基特征是 std::true_type 和 std::false_type 之一，取决于是否达到对应的条件。

基础类型类别

在标头 <type_traits> 定义

is_void (C++11)	检查类型是否为 void (类模板)
is_null_pointer (C++14)	检查类型是否为 std::nullptr_t (类模板)
is_integral (C++11)	检查类型是否为整数类型 (类模板)

is_floating_point (C++11)	检查类型是否是浮点类型 (类模板)
is_array (C++11)	检查类型是否是数组类型 (类模板)
is_enum (C++11)	检查类型是否是枚举类型 (类模板)
is_union (C++11)	检查类型是否为联合体类型 (类模板)
is_class (C++11)	检查类型是否非联合类类型 (类模板)
is_function (C++11)	检查是否为函数类型 (类模板)
is_pointer (C++11)	检查类型是否为指针类型 (类模板)
is_lvalue_reference (C++11)	检查类型是否为*左值引用* (类模板)
is_rvalue_reference (C++11)	检查类型是否为*右值引用* (类模板)
is_member_object_pointer (C++11)	检查类型是否为指向非静态成员对象的指针 (类模板)
is_member_function_pointer (C++11)	检查类型是否为指向非静态成员函数的指针 (类模板)

复合类型类别

在标头 <type_traits> 定义

is_fundamental (C++11)	检查是否是基础类型 (类模板)
is_arithmetic (C++11)	检查类型是否为算术类型 (类模板)
is_scalar (C++11)	检查类型是否为标量类型 (类模板)
is_object (C++11)	检查是否是对象类型 (类模板)
is_compound (C++11)	检查是否为复合类型 (类模板)
is_reference (C++11)	检查类型是否为*左值引用*或*右值引用* (类模板)
is_member_pointer (C++11)	检查类型是否为指向非静态成员函数或对象的指针类型 (类模板)

类型属性

在标头 <type_traits> 定义

is_const (C++11)	检查类型是否为 const 限定 (类模板)
is_volatile (C++11)	检查类型是否为 volatile 限定 (类模板)

is_trivial (C++11)	检查类型是否平凡 (类模板)
is_trivially_copyable (C++11)	检查类型是否可平凡复制 (类模板)
is_standard_layout (C++11)	检查是否是一个标准布局类型 (类模板)
is_pod (C++11)(C++20 中弃用)	检查类型是否为简旧数据（POD）类型 (类模板)
is_literal_type (C++11) (C++17 中弃用) (C++20 中移除)	检查类型是否为字面类型 (类模板)
has_unique_object_representations (C++17)	检查是否该类型对象的每一位都对其值有贡献 (类模板)
is_empty (C++11)	检查类型是否为类（但非联合体）类型且无非静态数据成员 (类模板)
is_polymorphic (C++11)	检查类型是否为多态类类型 (类模板)
is_abstract (C++11)	检查类型是否为抽象类类型 (类模板)
is_final (C++14)	检查类型是否为 `final` 类类型 (类模板)
is_aggregate (C++17)	检查类型是否聚合类型 (类模板)
is_implicit_lifetime (C++23)	检查类型是否为隐式生存期类型 (类模板)
is_signed (C++11)	检查类型是否为有符号算术类型 (类模板)
is_unsigned (C++11)	检查类型是否为无符号算术类型 (类模板)
is_bounded_array (C++20)	检查类型是否为有已知边界的数组类型 (类模板)
is_unbounded_array (C++20)	检查类型是否为有未知边界的数组类型 (类模板)
is_scoped_enum (C++23)	检查类型是否为有作用域枚举类型 (类模板)

受支持操作

在标头 `<type_traits>` 定义

is_constructible (C++11) **is_trivially_constructible** (C++11) **is_nothrow_constructible** (C++11)	检查类型是否带有针对特定实参的构造函数 (类模板)
is_default_constructible (C++11) **is_trivially_default_constructible** (C++11) **is_nothrow_default_constructible** (C++11)	检查类型是否有默认构造函数 (类模板)

is_copy_constructible (C++11) **is_trivially_copy_constructible** (C++11) **is_nothrow_copy_constructible** (C++11)	检查类型是否拥有复制构造函数 (类模板)
is_move_constructible (C++11) **is_trivially_move_constructible** (C++11) **is_nothrow_move_constructible** (C++11)	检查类型是否能从右值引用构造 (类模板)
is_assignable (C++11) **is_trivially_assignable** (C++11) **is_nothrow_assignable** (C++11)	检查类型是否拥有针对特定实参的赋值运算符 (类模板)
is_copy_assignable (C++11) **is_trivially_copy_assignable** (C++11) **is_nothrow_copy_assignable** (C++11)	检查类型是否拥有复制赋值运算符 (类模板)
is_move_assignable (C++11) **is_trivially_move_assignable** (C++11) **is_nothrow_move_assignable** (C++11)	检查类型是否有拥有移动赋值运算符 (类模板)
is_destructible (C++11) **is_trivially_destructible** (C++11) **is_nothrow_destructible** (C++11)	检查类型是否拥有未被弃置的析构函数 (类模板)
has_virtual_destructor (C++11)	检查类型是否拥有虚析构函数 (类模板)
is_swappable_with (C++17) **is_swappable** (C++17) **is_nothrow_swappable_with** (C++17) **is_nothrow_swappable** (C++17)	检查一个类型的对象是否能与同类型或不同类型的对象交换 (类模板)
reference_constructs_from_temporary (C++23)	检查在直接初始化中引用是否绑定到临时对象 (类模板)
reference_converts_from_temporary (C++23)	检查在复制初始化中引用是否绑定到临时对象 (类模板)

属性查询

属性查询特性可以用于在编译时查询类型的整数属性。

所有这些类型特性都满足*一元类型特性 (UnaryTypeTrait)*，每个类型特性的基特征是 `std::integral_constant<size_t, Value>`，其中 `Value` 是对应特性的查询结果。

在标头 <type_traits> 定义

alignment_of (C++11)	获取类型的对齐要求 (类模板)
rank (C++11)	获取数组类型的维数 (类模板)
extent (C++11)	获取数组类型在指定维度的大小 (类模板)

类型关系

类型关系特性可以用于在编译时查询类型之间的关系。

所有这些类型特性都满足*二元类型特性 (BinaryTypeTrait)*，每个类型特性的基特征是 `std::true_type` 和 `std::false_type` 之一，取决于是否达到对应的条件。

在标头 <type_traits> 定义

is_same (C++11)	检查两个类型是否相同 (类模板)
is_base_of (C++11)	检查一个类型是否派生自另一个类型 (类模板)
is_convertible (C++11) **is_nothrow_convertible** (C++20)	检查是否能转换一个类型为另一类型 (类模板)
is_layout_compatible (C++20)	检查二个类型是否*布局兼容* (类模板)
is_pointer_interconvertible_base_of (C++20)	检查一个类型是否为另一类型的*指针可互转换*（起始）基类 (类模板)
is_invocable **is_invocable_r** **is_nothrow_invocable** **is_nothrow_invocable_r** (C++17)	检查类型能否以给定的实参类型调用（如同以 `std::invoke`） (类模板)

类型变换

类型变换特性按照预定义对着将一个类型变换到另一个。

所有这些类型特性都满足*变换特性 (TransformationTrait)*。

常性/易变性说明符

在标头 <type_traits> 定义

remove_cv (C++11) **remove_const** (C++11) **remove_volatile** (C++11)	从给定类型移除 `const` 和/或 `volatile` 限定符 (类模板)
add_cv (C++11) **add_const** (C++11) **add_volatile** (C++11)	添加 `const` 和/或 `volatile` 限定符到给定类型 (类模板)

引用

在标头 <type_traits> 定义

remove_reference (C++11)	从给定类型移除引用 (类模板)
add_lvalue_reference (C++11) **add_rvalue_reference** (C++11)	向给定类型添加*左值*或*右值*引用 (类模板)

符号修改

在标头 <type_traits> 定义

make_signed (C++11)	使给定的整数类型有符号 (类模板)
make_unsigned (C++11)	使给定的整数类型无符号 (类模板)

数组

在标头 <type_traits> 定义

remove_extent (C++11)	从给定数组类型移除一个维度 (类模板)
remove_all_extents (C++11)	移除给定数组类型的所有维度 (类模板)

指针

在标头 <type_traits> 定义

remove_pointer (C++11)	移除给定类型的一层指针 (类模板)
add_pointer (C++11)	对给定类型添加一层指针 (类模板)

其他变换

在标头 <type_traits> 定义

aligned_storage (C++11)(C++23 中弃用)	定义适于用作给定大小的类型的未初始化存储的类型 (类模板)
aligned_union (C++11)(C++23 中弃用)	定义适于用作所有给定类型的未初始化存储的类型 (类模板)
decay (C++11)	实施当按值传递实参给函数时所进行的类型变换 (类模板)
remove_cvref (C++20)	将 std::remove_cv 与 std::remove_reference 结合 (类模板)
enable_if (C++11)	条件性地从重载决议移除函数重载或模板特化 (类模板)
conditional (C++11)	基于编译时布尔值选择一个类型或另一个 (类模板)
common_type (C++11)	确定一组类型的公共类型 (类模板)
common_reference **basic_common_reference** (C++20)	确定类型组的共用引用类型 (类模板)
underlying_type (C++11)	获取给定枚举类型的底层整数类型 (类模板)
result_of (C++11)(C++20 中移除) **invoke_result** (C++17)	推导以一组实参调用一个可调用对象的结果类型 (类模板)

void_t (C++17)	void 变参别名模板 (别名模板)
type_identity (C++20)	返回不更改的类型实参 (类模板)

逻辑运算

逻辑运算符特性将逻辑运算符应用到其他类型特性上。

在标头 <type_traits> 定义

conjunction (C++17)	变参的逻辑与元函数 (类模板)
disjunction (C++17)	变参的逻辑或元函数 (类模板)
negation (C++17)	逻辑非元函数 (类模板)

编译时有理数算术

标头 <ratio> 提供操作和存储编译时比例的类型和函数。

编译时整数序列

在标头 <utility> 定义

integer_sequence (C++14)	实现编译时整数数列 (类模板)

来自“https://zh.cppreference.com/mwiki/index.php?title=cpp/meta&oldid=78524”