## Function template

2018年4月20日 14:21

1. Template Argument Deduction

type conversions during type deduction 注意:自动类型转换在类型推导时会被限制

- 当声明使用传引用调参时,类型推导甚至不会应用细小的转换。
- 当声明使用传值调参时,只有细小的转换——退化会被支持:忽略限定词 cosnt,volatile,引用转化为引用类型,数组或函数转化为相应的指针类型。
- 2. Multiple Template Parameter

c++提供了不同的方法来处理返回类型的问题:

- 引入第三个模板参数用作返回类型
- 让编译器找出返回类型
- 将返回类型声明为两个参数类型的"common type"

显示指定模板参数

3. Default Template Argument

你也可以定义模板参数的默认值,它可以和任意模板一起使用(c++11)。

4. Overloading Function Templates

c++编译通过一些额外的信息来决定调用重载函数中的一个,几乎都是通过调用参数的类型信息来决定。

http://zh.cppreference.com/w/cpp/language/overload resolution

5. Pass by Value or by Reference?

你可能会想,为什么声明函数参数一般使用传值的方式而不是传引用。一般来说,对于那些不是简单类型(基本类型或std::string\_view)的类型推荐传引用方式,因为传值会产生不必要的拷贝。

但是,由于以下的原因,一般来说传值更好:

- 语法简单
- 编译器优化更好
- 移动语义经常降低拷贝开销
- 有些时候根本没有拷贝或移动

另外,对于模板,还有这些方面的作用:

- 一个模板可能会用于简单类型和复杂类型这两者,所以如果选择适合复杂类型的方式, 也许对于简单类型会适得其反。
- 作为函数调用者,你仍然可以决定使用引用传参,使用std::ref(), std::cref()。

• 尽管传递字符串或数组总是会变成一个问题,但是使用引用的方式传递它们经常被认为会变成一个更大的问题。

## 6. 小结

- 模板函数为不同的模板实参定义了一个函数家族
- 当你传递模板实参时,函数模板会推导模板参数来实例化相应的参数类型
- 你可以显示指定模板参数
- 你可以定义模板参数的默认值
- 你可以重载函数模板
- 当你用其他函数模板重载函数模板时,你应当确保任何调用有唯一匹配
- 当你重载函数模板时,将你的改变限制在显示指定模板参数
- 确保在你调用函数模板之前,编译器能看到所有重载版本