C++ 中函数的参数加了&, const 和不加 有什么区别

转载 technologyleader 2019-07-22 09:44:44 1471 收藏 4
分类专栏: c++

如下函数fun(int* in, const std::string& str)

{

}

一个参数都不加， 如 std：：string str 和 std::sring& str 的区别是什么呢?

std:: string str 和 const::string str 使用的区别是什么呢?

如果把 const 和& 都加上又有什么不同呢?

std::string str和std::sring& str(单加 &)和const::string str (单加 const)和const::string& str(const和 &都加了) 4者有何区别呢？分别都是什么情况用?

class 成员 加上const 或者 & 有什么特别作用吗?

不加引用的话，str则被复制一份，函数中对str的操作实质上是对其复制品的操作，所以即使在函数中修改了str，调用层的原str并不会被改变。

加了引用的话，传入的str即是调用层的实际参数，这样省却了复制过程，效率会有提高。但如果函数中修改了str，则原str也会改变，因为其实是同一个东西。

有时候为了追求效率，又希望避免改变原来的str，则可在引用的基础上加const修饰，这样函数中就不能再修改str的内容（否则会编译出错）。

一个参数都不加， 如 std：：string str 和 std::sring& str 的区别是什么呢?
std:: string str 和 const::string str 使用的区别是什么呢?
答：std:: string str ，则str可以被修改，而const::string str ，则str不能被修改。
如果把 const 和& 都加上又有什么不同呢?

答：对于有&，则不会调用拷贝构函造数，也就不会生成副本，函数里实际操作的是参实本身。

对于没有&，则会调用拷贝构造函数，生成实参的本，副你实际操作的是实参的副本，而不是实参本身。

std::string str和std::sring& str(单加 &)和const::string str (单加 const)和const::string& str(const和 &都加了) 4者有何区别呢？分别都是什么情况用?
答：std::string str，str可以被修改，而且会调用拷贝构造函数。
std::sring& str，str可以被修改，但不会调用拷贝构造函数。
const::string str ， str不能被修改，但会调用拷贝构造函数。
const::string& str，str不能被修改，而且也不会调用拷贝构造函数。

c++ 函数传参 string string& const string &三者