C++中const关键字用法总结

http://blog.csdn.net/eric\_jo/article/details/4138548

1.const常量的作用域：

const与否对作用域没有影响。

2.cosnt常量的意义

1）const常量有数据类型，而宏常量没有数据类型。

2）防止意外的修改。

3）为函数重载提供了一个参考。

4）提高了效率。编译器将它们保存在符号表中。

3.const的定义常量

TYPE const ValueName = value;

const TYPE ValueName = value;

const修饰的类型为TYPE的变量value是不可变的。

const常量只有在初始化的时候可以赋值。

4.指针使用const

1)指针本身是常量不可变

char\* const pContent;

2)指针所指向的内容是常量不可变

const char \*pContent; (等价于char const \*pContent;)

3)两者都不可变

const char\* const pContent;

5.函数中使用const

1)const修饰函数参数

a.传递过来的参数在函数内不可以改变(无意义，因为Var本身就是形参)

void function(const int Var);

b.参数指针所指内容为常量不可变

void function(const char\* Var);

c.参数指针本身为常量不可变(也无意义，因为char\* Var也是形参)

void function(char\* const Var);

d.参数为引用，为了增加效率同时防止修改。修饰引用参数时：

void function(const Class& Var); //引用参数在函数内不可以改变

(2)const 修饰函数返回值  
    const修饰函数返回值其实用的并不是很多，它的含义和const修饰普通变量以及指针的含义基本相同。  
    a.const int fun1() //这个其实无意义，因为参数返回本身就是赋值。  
    b. const int \* fun2() //调用时 const int \*pValue = fun2();   
                          //我们可以把fun2()看作成一个变量，即指针内容不可变。  
    c.int\* const fun3()   //调用时 int \* const pValue = fun2();   
                          //我们可以把fun2()看作成一个变量，即指针本身不可变。

一般情况下，函数的返回值为某个对象时，如果将其声明为const时，多用于操作符的重载。通常，不建议用const修饰函数的返回值类型为某个对象或对某个对象引用的情况。原因如下：如果返回值为某个对象为const（const A test = A 实例）或某个对象的引用为const（const A& test = A实例） ，则返回值具有const属性，则返回实例只能访问类A中的公有（保护）数据成员和const成员函数，并且不允许对其进行赋值操作，这在一般情况下很少用到。

6.类相关CONST

1)const修饰成员变量

const修饰类的成员函数，表示成员常量，不能被修改，同时它只能在初始化列表中赋值。

class A

{

const int nValue; //成员常量不能被修改

A(int x): nValue(x) { } ; //只能在初始化列表中赋值

}

2)const修饰成员函数

const成员函数不被允许修改它所在对象的任何一个数据成员,const成员函数能够访问对象的const成员,而其他成员函数不可以。

一般写在函数的最后来修饰。

const修饰成员函数的最重要作用是限制对于const对象的使用。

class A

{   
       void function()const; //常成员函数, 它不改变对象的成员变量.

//也不能调用类中任何非const成员函数。

}

3)const修饰类对象/对象指针/对象引用

const修饰类对象表示该对象为常量对象，其中的任何成员都不能被修改。对于对象指针和对象引用也是一样。

 const修饰的对象，该对象的任何非const成员函数都不能被调用，因为任何非const成员函数会有修改成员变量的企图。