CSDN 学院

C++语言基础

迁者 - 贺利坚

http://blog.csdn.net/sxhelijian/

http://edu.csdn.net

本节主题：

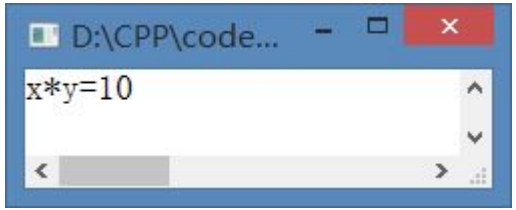
用const实施保护

共用数据的保护的问题

```
class  Test
{
private:          //private保护数据只用于类内
    int x, y;     //x和y是Test类中所有成员函数的共用数据
public:
   Test(int a, int b){x=a;y=b;}
   void printxy(void)
   {
        x=2;
        cout<<"x*y="<<x*y<<endl;
   }
};
```

```
int main()
{
    Test p1(3,5);
     p1.printxy( );
}
```

问题：如何避免无意之中的误操作，意外地改变有关数据？



解决方案：使用const

- 使用const，把有关的数据定义为常量（常变量/常对象/常函数）
- 既要使数据能在一定范围内共享，又要保证它不被任意修改
  - 常 对象
  - 常 对象成员
  - 指向对象的常 指针
  - 指向常对象的 指针变量
  - 对象的常 引用

常对象

- 回顾：常变量
  - const int PIRCE = 40
- 凡希望保证数据成员不被改变的对象，可以声明为常对象
- 常对象中的所有成员的值都不能被修改
- 常对象两种等价的定义形式
  - 类名 **const** 对象名[(实参表列)];
  - **const** 类名 对象名[(实参表列)];
- 常对象必须要有初值，如
  - Time **const** t1(12,34,46);
  - **const** Time t1(12,34,46);

```
class Time
{
public:
   void set_time(int h,int m,int s)
   {
      hour=h;
      minute=m;
      sec=s;
   };
private:
   int hour;
   int minute;
   int sec;
};
```

```
int main( )
{
   const Time t1(10,10,10);
   t1.set_time(20,20,20);
   return 0;
}
```

常对象成员——常成员函数

- 要引用常对象中的数据成员，需将该成员函数声明为const型函数——常成员函数。
- 常成员函数可以访问常对象中的数据成员，但不允许修改常对象中数据成员的值。

```
class Time
{
public:
   void show_time( ) const;
   ……
private:
   int hour, minute, second;
};

void Time::show_time( ) const
{
   cout<<hour<<":"<<minute<<":"<<sec<<endl;
}
```

const是函数类型的一部分，在声明函数和定义函数时都要有const关键字

```
int main( )
{
   const Time t1(10,10,10);
   t1.show_time();
   return 0;
}
```

discards qualifiers
缺少限定符

常对象成员——常数据成员

- 用关键字const来声明常数据成员，限定其值不能改变

```
class Time
{
   const int hour;
   int minute; ……
}
```

- 只能通过构造函数的参数初始化表对常数据成员进行初始化

```
Time(int h):hour(h){ }      //在类内定义
Time::Time(int h):hour(h){ } //在类外定义
```

- 不能在构造函数中用赋值的方法对常数据成员初始化

```
Time::Time(int h){hour=h;} //非法
```

- 不能用成员函数改变常数据成员的值

```
void Time::setHour(int h){hour=h;} //非法
```

- 对比常对象

```
Time const t1(12,34,46);
```

常对象的数据成员都是常数据成员。

不同成员函数对数据成员访问的限制

- 常对象中的成员函数未加const，编译系统认为其是非const成员函数。
- 成员函数对成员的访问（从const角度）

```
class Time
{
public:
   void show_time( ) const;
   ……
private:
   int hour, minute, second;
};

void Time::show_time( ) const
{
   cout<<hour<<":"<<minute<<":"<<sec<<endl;
}
```

成员函数对数据成员的访问规则

数据成员	非const成员函数	const成员函数
非const数据成员	可引用 可改变值	可引用 不可改变值
const数据成员	可引用 不可改变值	可引用 不可改变值
const对象的数据成员	不可引用 不可改变	可引用 不可改变值

对常对象修改的限制

- **如果一个对象被声明为常对象，则不能调用该对象的非const型的成员函数**(除了由系统自动调用的隐式的构造函数和析构函数)。
  - const Time t1(10,15,36);
  - t1.get_time( ); //非法
- 效果
  - 编译系统只检查函数的声明，只要发现调用了常对象的成员函数，而且该函数未被声明为const，就报错。
  - 防止函数修改常对象中数据成员的值；
  - 编译系统对不安全的因素予以拦截，不必仅依靠编程者的细心来保证程序不出错，
  - 用常成员函数引用常变量

设计策略

- 如果在一个类中，有些数据成员的值允许改变，另一些数据成员的值不允许改变，则
  - 将一部分数据成员声明为const，以保证其值不被改变，用非const的成员函数引用和修改非const数据成员的值。
- 如果要求所有的数据成员的值都不允许改变
  - 将所有的数据成员声明为const，或将对象声明为const(常对象)，用const成员函数引用数据成员。
- 如果已定义一个常对象，为访问对象中的数据成员
  - 将常对象中所有成员函数都声明为const成员函数，但应确保在函数中不修改对象中的数据成员。
- 一定要修改常对象中的某个数据成员，该数据成员声明为mutable
  - mutable int count; //count为可变的数据成员，可以用const成员函数修改其值

CSDN 学院

THANKS

本课程由 迁者-贺利坚 提供

CSDN网站：www.csdn.net
企业服务：http://ems.csdn.net/
人才服务：http://job.csdn.net/
CTO俱乐部：http://cto.csdn.net/
高校俱乐部：http://student.csdn.net/
程序员杂志：http://programmer.csdn.net/

CODE平台：https://code.csdn.net/
项目外包：http://www.csto.com/
CSDN博客：http://blog.csdn.net/
CSDN论坛：http://bbs.csdn.net/
CSDN下载：http://download.csdn.net/