c++使用 noncopyable 类禁止类的拷贝

继承 noncopyable 类

如果某些事物是独一无二的，那么其相应的类就应该禁止拷贝，也就是要使类的 copy 构造函数和 copy assignment 操作符不起作用。比如说，地球就是独一无二的，那么可以这么定义地球：

[cpp] view plain copy

```cpp
class Earth {
public:
 ......//member 函数或 friend 函数
private:
 Earth(const Earth&);//只声明
 Earth& operator=(const Earth&);//只声明
};
```

有了上述 class 定义，当客户企图拷贝 Earth 对象，编译器会阻挠他。如果你不慎在 member 函数或 friend 函数之内那么做，轮到连接器发出抱怨。但是一种更好的做法是将连接期错误移至编译期，毕竟愈早侦测出错误愈好。此时可以专门设计一个阻止 copying 动作的 base class，如下所示：

[cpp] view plain copy

```cpp
class noncopyable
{
protected:
 noncopyable() {}
 ~noncopyable() {}
private:// emphasize the following members are private
 noncopyable( const noncopyable& );
 noncopyable& operator=( const noncopyable& );
};
```

对于构造函数为什么声明成 protected 呢？首先肯定不能为 private，不然无法构造子类实例。如果为 public，那么外部是可以创建 noncopyable 这么一个实例的，可是这个实例是完全没有意义的，该类只有在被继承之后才有意义。所以此处声明为 protected 是非常恰当合适的，既保证外部无法直接构造一个无意义的 noncopyable 实例，又不影响构造子类实例。**声明 protect 的原因是保证外部无法直接构造一个无意义的 noncopyable 实例，又不影响构造子类实例。**

现在，为了阻止 Earth 对象被拷贝，我们唯一需要做的就是继承 Uncopyable。当然，此时 Earth 内部不需要再声明 copy 构造函数和 copy assignment 操作符了。

[cpp] view plain copy

```
class Earth:private noncopyable
{
 ......
};
```