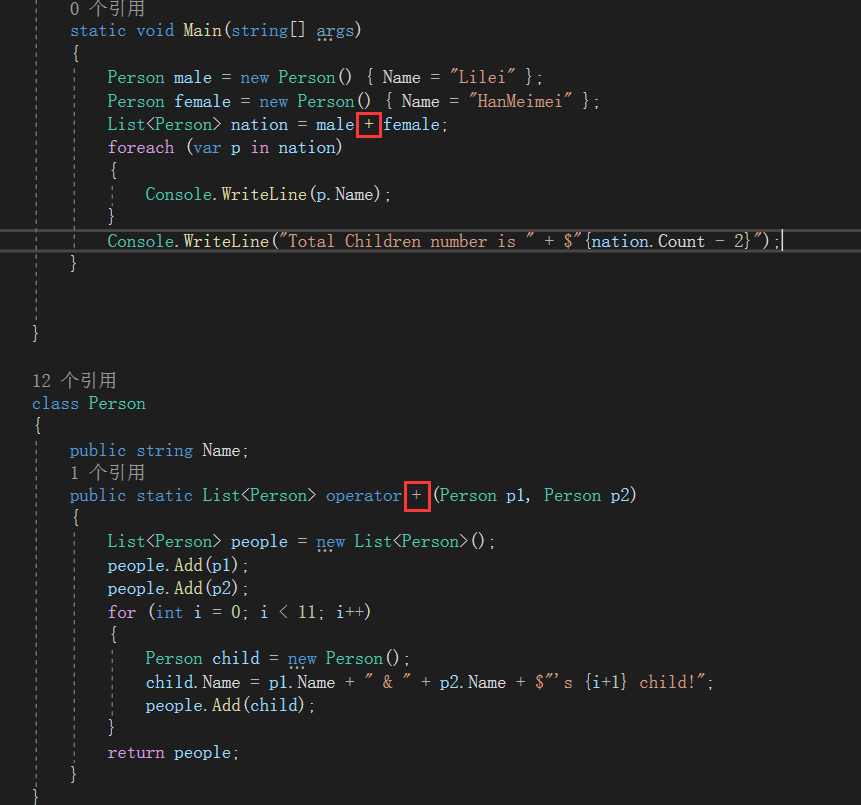
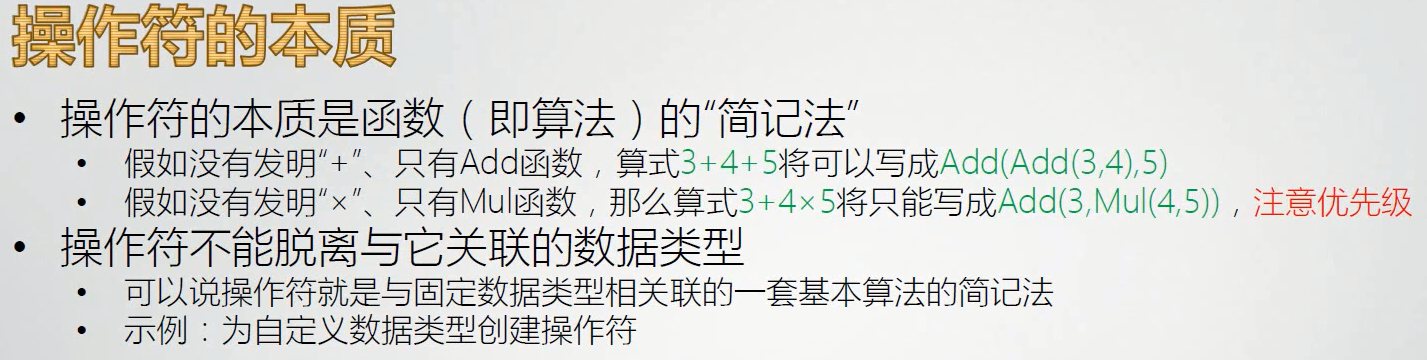
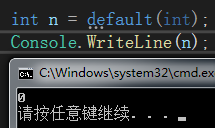


自定义操作符

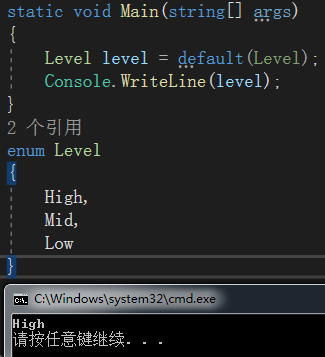
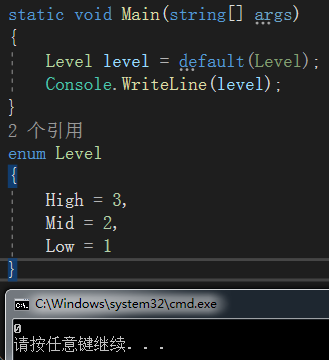




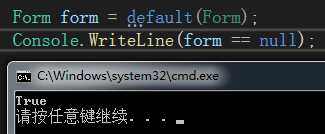
default() 获取数据类型的默认值



结构体类型的默认值是0

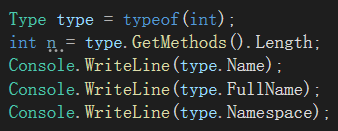
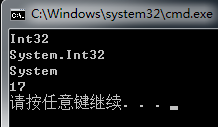
  

枚举类型的默认值是值为0的项，若赋值为非0，则返回0

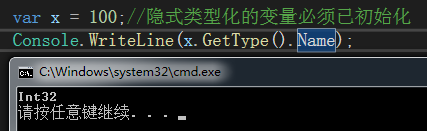


引用类型的默认值是null

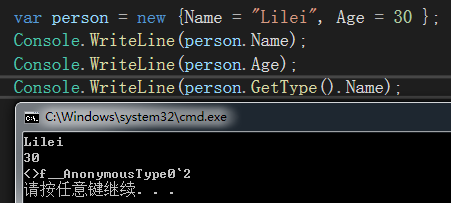
typeof() 获取类型的内部结构

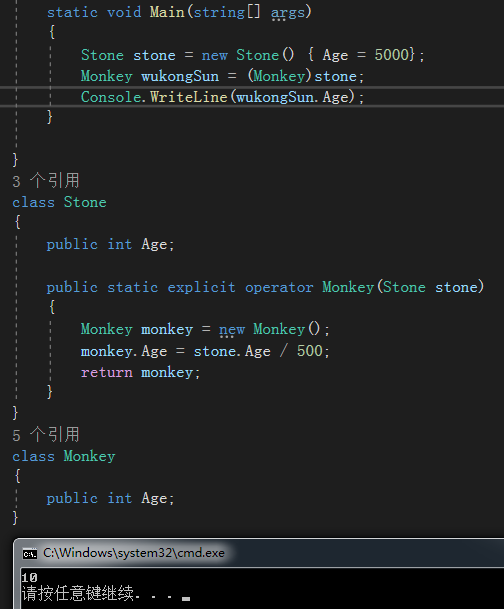
隐式类型



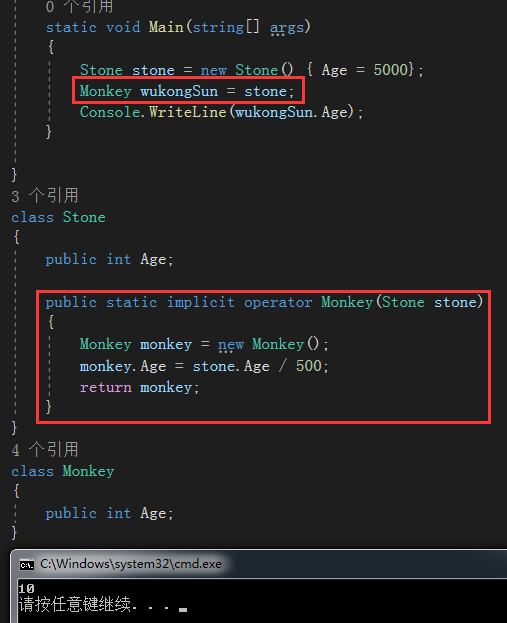
匿名类型创建对象 用var的隐式类型引用



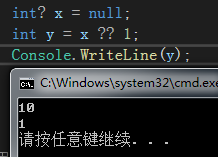
checked()检查一个数是否溢出 unchecked()即不检查（默认）

自定义类型的显式类型转换  


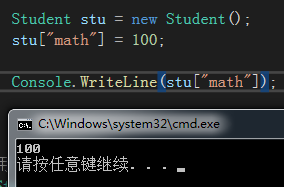
自定义类型隐式类型转换：



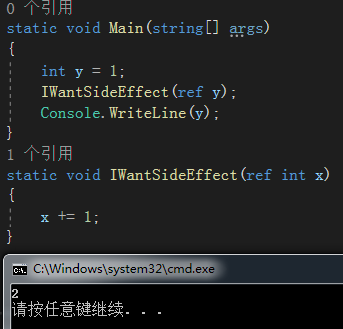
Int? 等价于 Nullable<int> 可以赋null值的整型 ?? 用于检查是否为null值



定义索引器



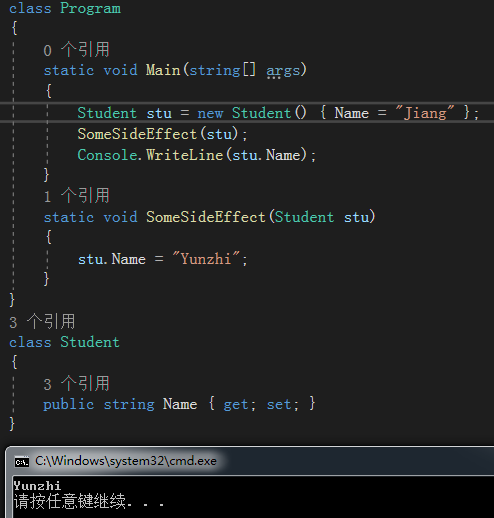
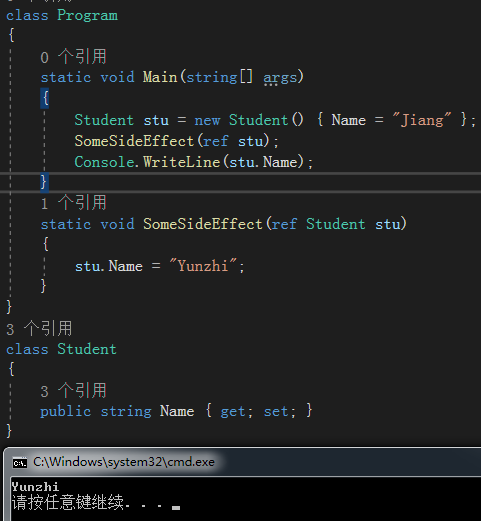
引用变量改变传入的参数值



若方法的形参是类型参数时，无论是否加ref（引用）都会修改原来的值

区别在于不加ref时是传值，会在栈上保存一个实参的一个副本，即实参和形参在栈上有不同的地址，实参和形参的内容都是实例在堆上的地址，

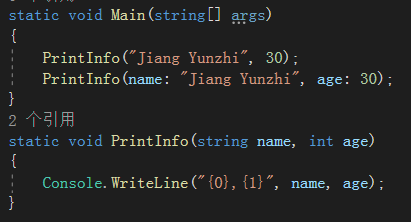
而加了ref，栈上不会生成一个副本，即形参和实参指向同一个内存地址

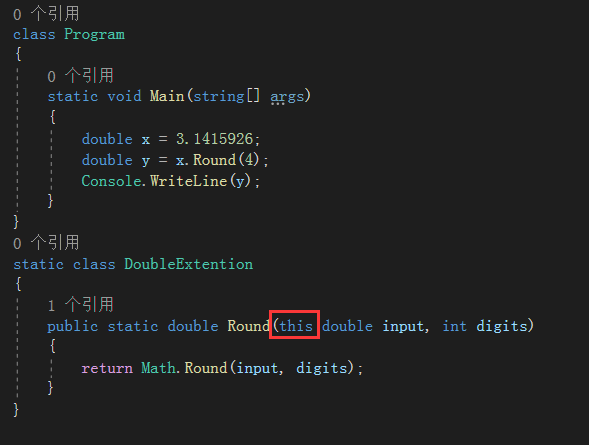
数组参数

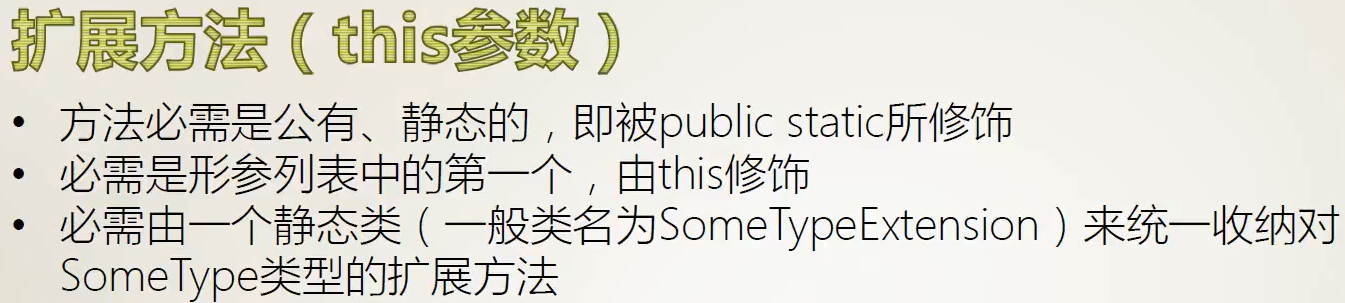


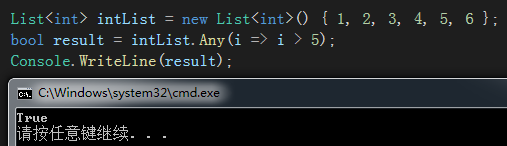
具名参数：挺高代码可读性，可以改变参数的先后位置

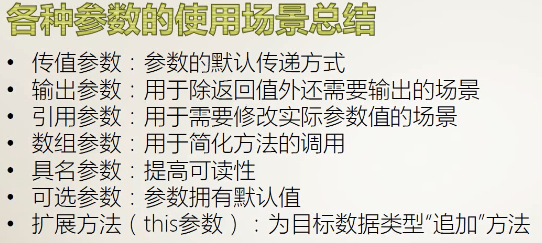


扩展方法：例如LINQ方法

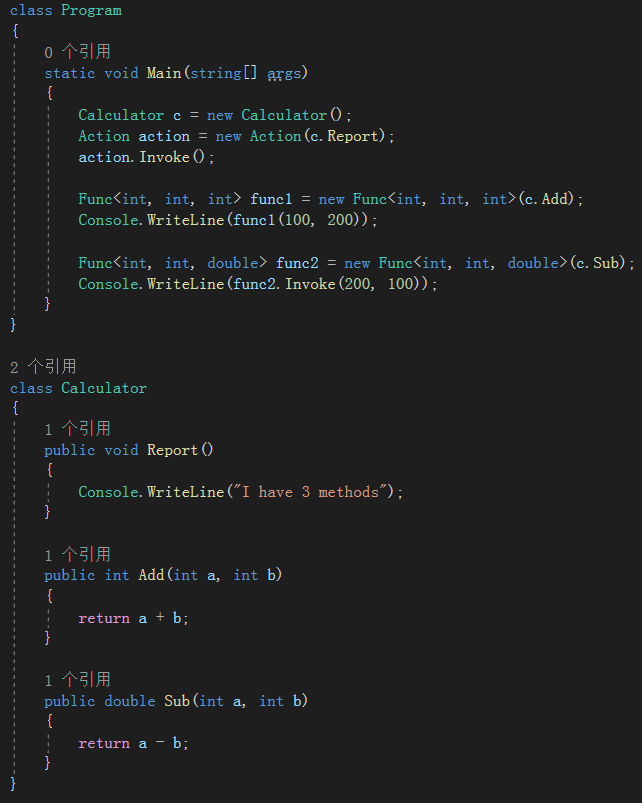


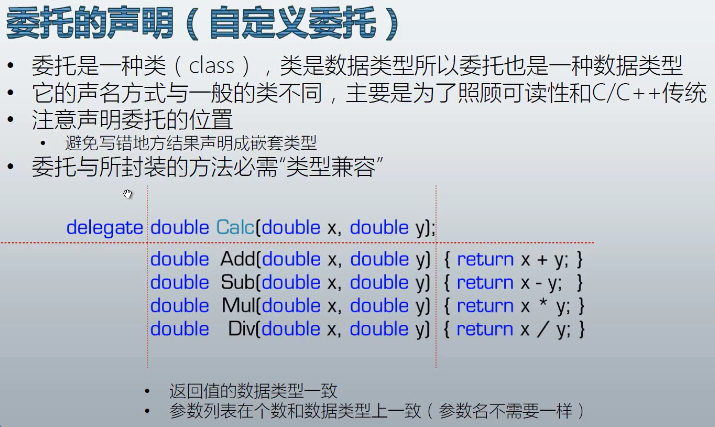


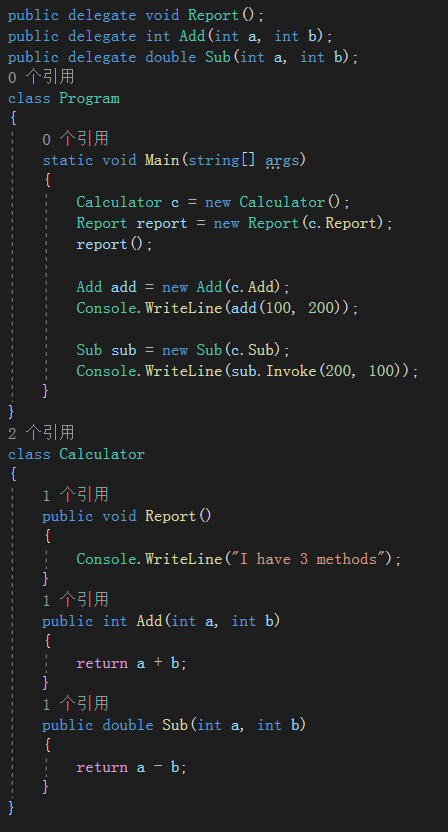


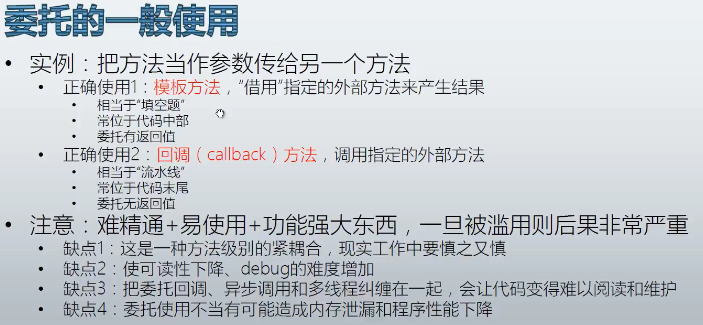


委托

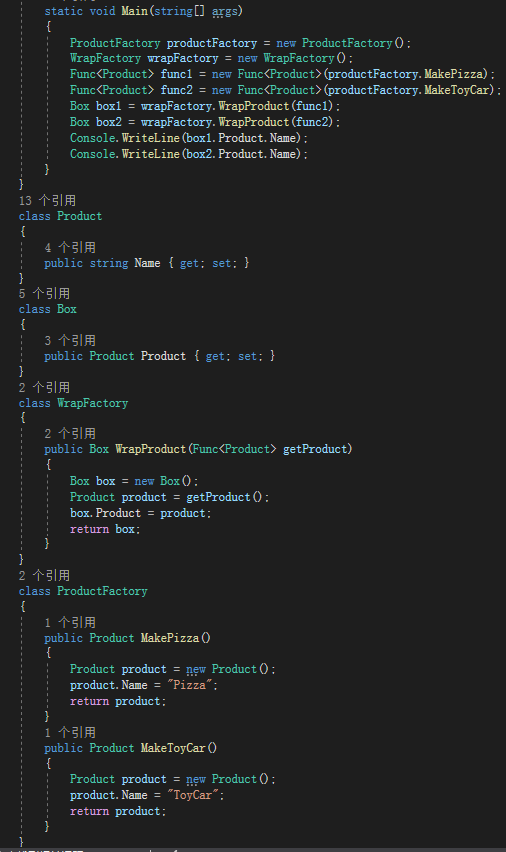




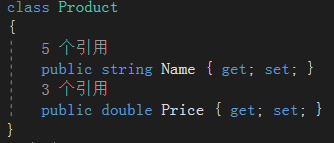


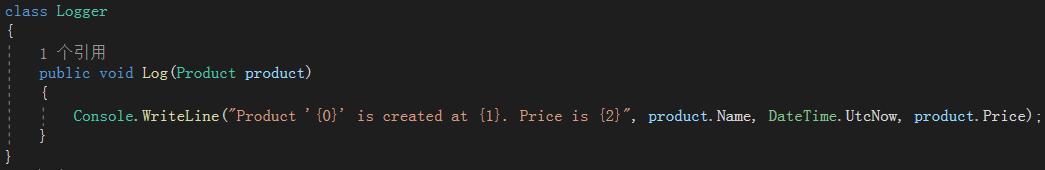


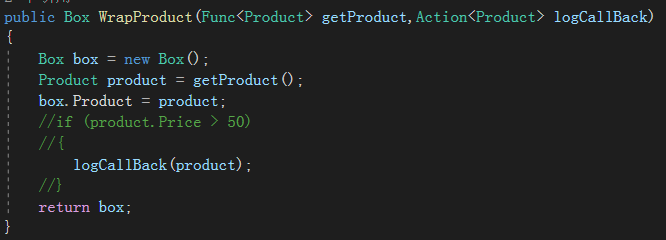
模版方法：

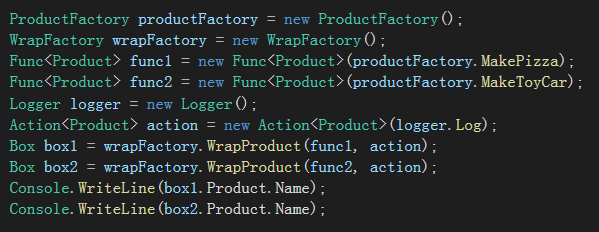


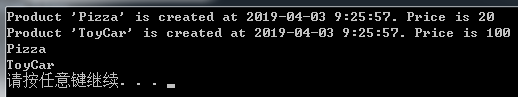
回调（CallBack）：

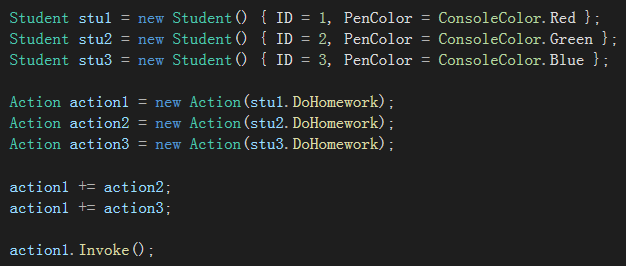




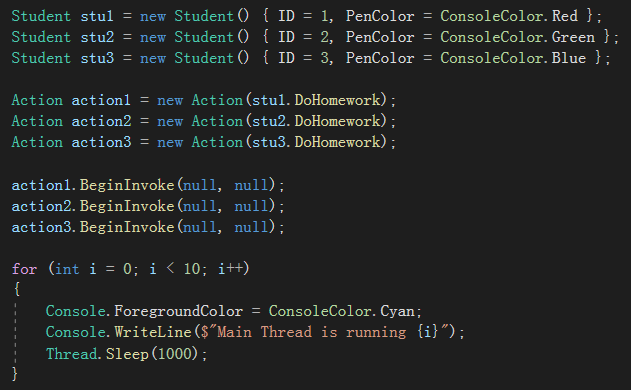
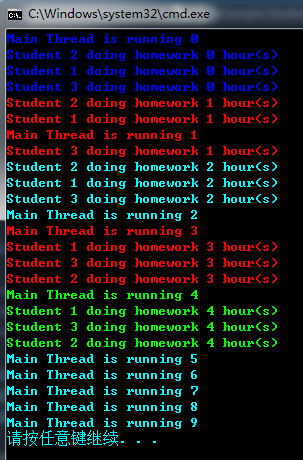




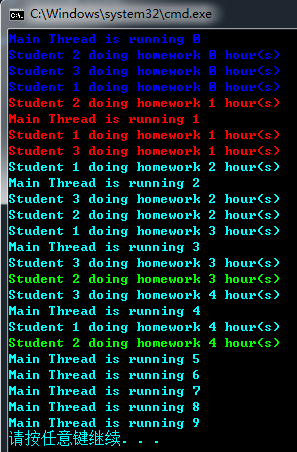
多播委托（同步调用）

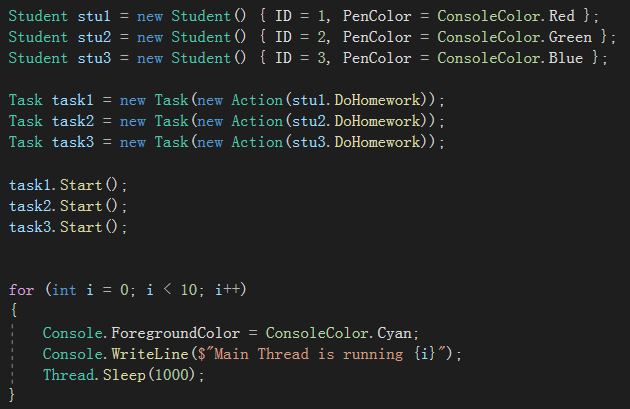


隐式异步调用

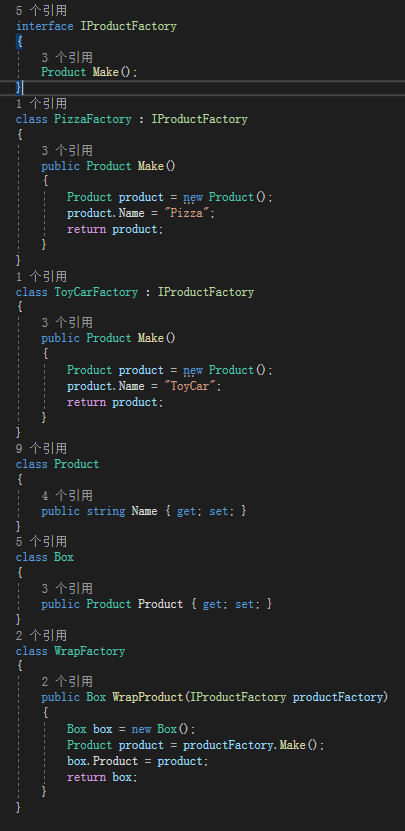
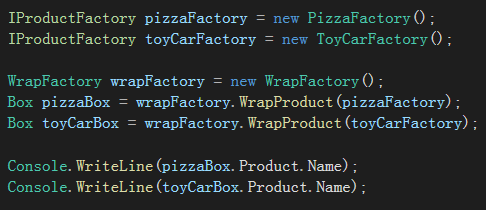
 

显式异步调用（创建子线程）

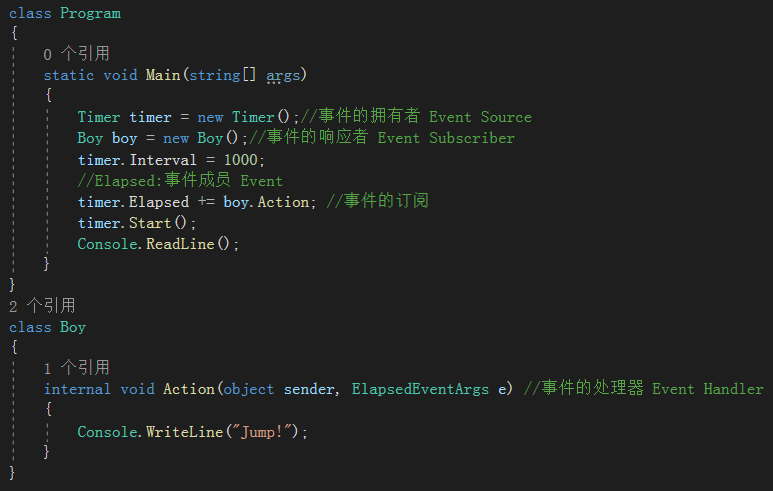
 

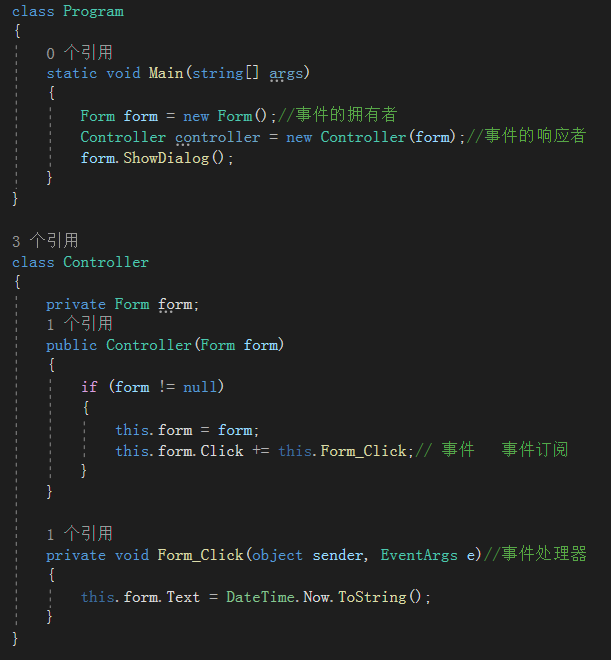


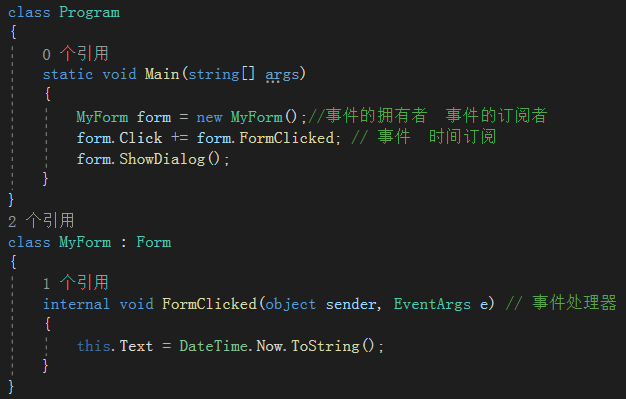
使用接口取代委托

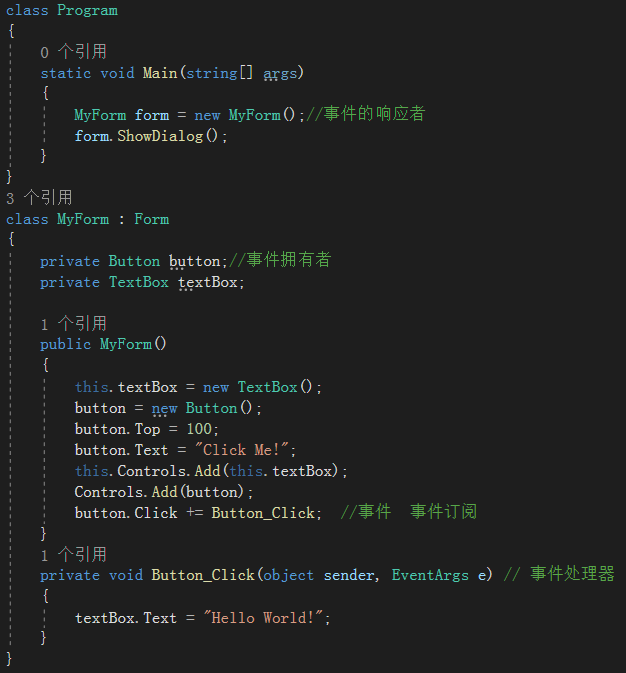
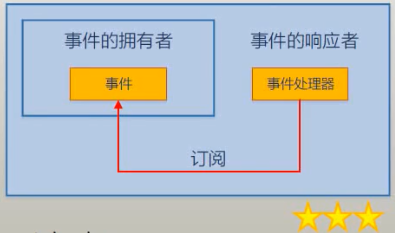
 

事件的组成部分



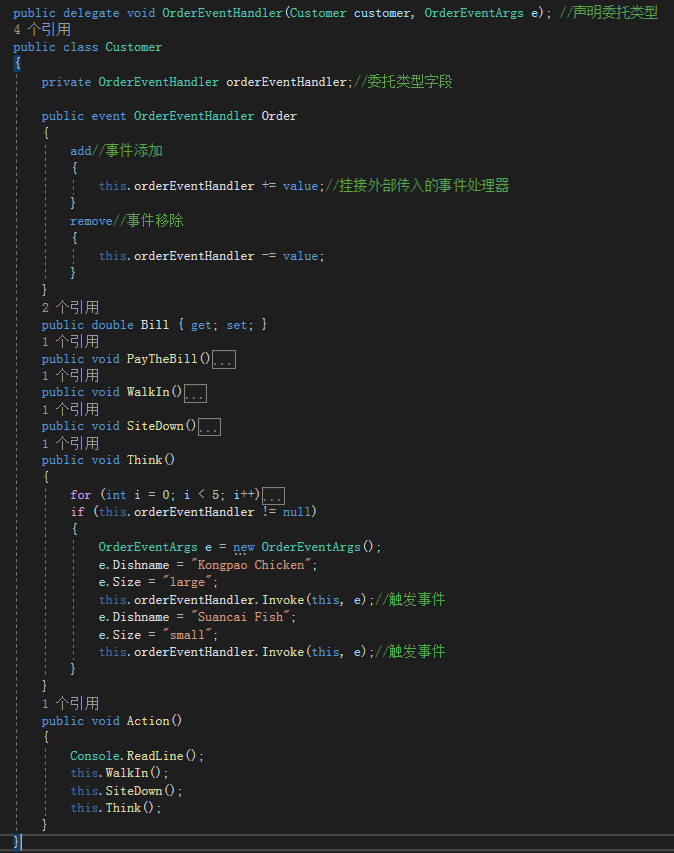
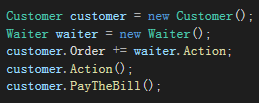
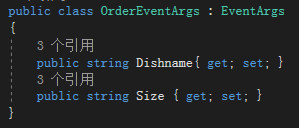
 

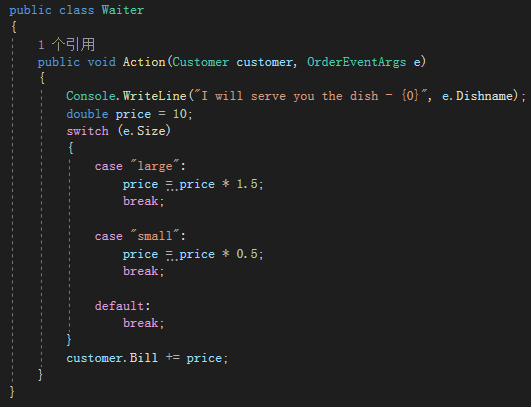
 

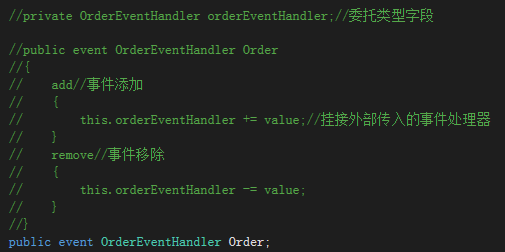
事件基于委托

事件完成声明：





事件简略声明：（field-like）



可以使用EventHandler代替自定义的委托

sealed修饰的类是不能被继承的

一个类只能有一个基类，但是可以有多个基接口

子类的访问级别不能超越父类

重写 多态

具体类 -> 抽象类 -> 接口

接口隔离原则

