【1.委托 事件】

IL 便于看本质

委托的本质是类，继承自MulticaseDelegate

委托三部曲

声明

实例化

调用

为什么要用委托

可以把方法包裹成变量，在Invoke时再调用

core 3.0的管道处理模型

逻辑解耦，保持稳定

代码复用，保证规范性

【2.lambda】

其实就是一个匿名方法

()=>{}

lambda的参数类型可以省略，可以根据委托声明推算出

基于lambda注册的多播委托无法移除

因为匿名，即使方法内容一样，但也属于不同的，移除不掉

【Action & Func】

Fram3.0推出的内置委托

【Action没有参数，没有返回值

Action action=()=>{};

泛型委托

Action<string> a1=s=>{};

Action<int,int> a2=(a,b)=>{}

最多可以指定16个参数

【Func有返回

Func<int> f0=()=>1+2;

lambda除了可以用于委托，

也可以做语法糖

c#6.0

public string Name=>"lambda"; 属性

public string GetName()=>"abd"; 方法

还可以用于表达式目录树，翻译成sql

vs插件Expression Tree Viewer

【3.Linq】

System.Linq

.Where

自己实现Where

但对于耗时的，框架的Where会依次出来，自己实现的会一次性冒出来

是因为IEnumberable + yield

迭代器升级后，可以按需获取，延迟加载

自己实现的按需加载，调试会发现，在打印输出的时候在一个一个进入

状态机模式

【4.事件】

委托都是多播委托

委托，+=，-=,决定一个方法是否匹配，由两个因素决定，method和target

所以注册时new Class和lambda无法移除

method.BeginInvoke，异步多线程

method.GetInvocationList()可以获取注册列表

甩锅大法-把不稳定的地方移除，交给别人负责

Event可以限制委托的操作，外部只能+=,-=，安全限制

event与委托区别、联系

委托是一个类

事件，是委托的实例

安全限制