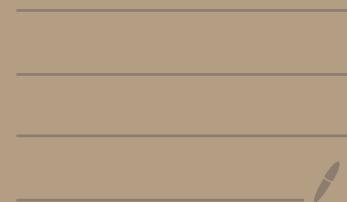

07. 06. 23

北島



C# 因角解 12k 第十五章

事件

15. | 相关 basic 概念

- 发布者 (publisher) 发布某个事件的类或结构，其他类可以在该事件发生时得到通知。
- 订阅者 (subscriber) 注册并在事件发生时得到通知的类或结构。
- 事件处理器 (event handler) 由订阅者注册到事件的方法，在发布者触发事件时执行。事件处理器方法可以定义在事件所在的类或结构中，也可以定义在不同的类或结构中。
- 触发 (raise) 事件 调用 (invoke) 或触发 (fire) 事件的术语。当事件被触发时，所有注册到它的方法都会被依次调用。

上一章介绍了委托。事件的很多部分都与委托类似。实际上，事件就像是专门用于某种特殊用途的简单委托。委托和事件的行为之所以相似，是有充分理由的。事件包含了一个私有的委托，如图 15-2 所示。

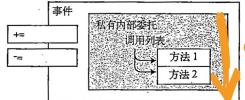


图 15-2 事件有被封装的委托

这一模型
包含 private delegate

- 委托类型声明 事件和事件处理程序必须有共同的签名和返回类型，它们通过委托类型进行描述。
- 事件处理器声明 订阅者类中会在事件触发时执行的方法声明。它们不一定是显式命名的方法，还可以是第 14 章描述的匿名方法或 Lambda 表达式。
- 事件声明 发布者类必须声明一个订阅者类可以注册的事件成员。当类声明的事件为 public 时，称为发布了事件。
- 事件注册 订阅者必须注册事件才能在事件被触发时得到通知。这是将事件处理程序与事件相连的代码。
- 触发事件的代码 发布者类中“触发”事件并导致调用注册的所有事件处理程序的代码。



图 15-4 使用事件时的 5 个源代码组件

event EventHandle :

这里签名实际上未包含
方法

Publisher:

(1) 声明事件

public event E..H.. EH=new dek...

发布

(4) 设置触发

Subscriber

(2) 实例化 Publisher 对象后为 EH
添加/删除方法(显/暗)
(Lambda)

(3) 声明实现上文方法

都可直接在各自
的类中直接写，不必在两个
类中一直切着写

增加泛型：主要是针对 EventHandle 的原型中
的 EventArgs 不够等问题

- 构造方法为 `EventArgs.Empty()` `void EventHandle(object? sender, EventArgs e)`

泛生出泛生类 `class ExEventHandle : EventHandle`

实现化例：`public event EventHandle<ExEventHandle> E`
`=new delegate{ };`

事件访问器太高级，本人学基础先 skip 3