## 通过事件使用委托

事件在类中声明且生成，且通过使用同一个类或其他类中的委托与事件处理程序关联。包含事件的类用于发布事件。这被称为 **发布器（publisher）** 类。其他接受该事件的类被称为 **订阅器（subscriber）** 类。事件使用 **发布-订阅（publisher-subscriber）** 模型。

**发布器（publisher）** 是一个包含事件和委托定义的对象。事件和委托之间的联系也定义在这个对象中。发布器（publisher）类的对象调用这个事件，并通知其他的对象。

**订阅器（subscriber）** 是一个接受事件并提供事件处理程序的对象。在发布器（publisher）类中的委托调用订阅器（subscriber）类中的方法（事件处理程序）。

## 声明事件（Event）

在类的内部声明事件，首先必须声明该事件的委托类型。例如：

public delegate void BoilerLogHandler(string status);

然后，声明事件本身，使用 **event** 关键字：

// 基于上面的委托定义事件

public event BoilerLogHandler BoilerEventLog;

上面的代码定义了一个名为 *BoilerLogHandler* 的委托和一个名为 *BoilerEventLog* 的事件，该事件在生成的时候会调用委托。

## 实例 1

// 事件发布器

class DelegateBoilerEvent

{

//定义委托

public delegate void BoilerLogHandler(string status);

// 基于上面的委托定义事件

public event BoilerLogHandler BoilerEventLog;

public void LogProcess()

{

string remarks = "O. K";

Boiler b = new Boiler(100, 12);

int t = b.getTemp();

int p = b.getPressure();

if(t > 150 || t < 80 || p < 12 || p > 15)

{

remarks = "Need Maintenance";

}

OnBoilerEventLog("Logging Info:\n");

OnBoilerEventLog("Temparature " + t + "\nPressure: " + p);

OnBoilerEventLog("\nMessage: " + remarks);

}

//调用事件的方法

protected void OnBoilerEventLog(string message)

{

if (BoilerEventLog != null)

{

BoilerEventLog(message);

}

}

}

// 事件订阅器

public class RecordBoilerInfo

{

static void Logger(string info)

{

Console.WriteLine(info);

}//end of Logger

static void Main(string[] args)

{

//实例化发布器

DelegateBoilerEvent boilerEvent = new DelegateBoilerEvent();

//实例化一个发布器的委托，并添加到发布器的事件中

boilerEvent.BoilerEventLog += new

DelegateBoilerEvent.BoilerLogHandler(Logger);

//调用发布器的方法，特定条件下该方法会调用 事件调用方法 来调用事件

boilerEvent.LogProcess();

}//end of main

}//end of RecordBoilerInfo