Lecture 6 – Events

Mr. Yousif Garabet Arshak
Computer Science Department
University of Zakho
yousif.arshak@uoz.edu.krd

Outlines

- Introduction
- Events
- Delegates
- Event handlers
- Static and dynamic event handlers
- Raising multiple events

Introduction

- Events in .NET are based on the delegate model. The delegate model follows the observer design pattern, which enables a subscriber to register with and receive notifications from a provider. An event sender pushes a notification that an event has happened, and an event receiver receives that notification and defines a response to it.

Events

- An event is a message sent by an object to signal the occurrence of an action. The action can be caused by user interaction, such as a button click, or it can result from some other program logic, such as changing a property's value.
- The object that raises the event is called the *event sender*.
- The event sender doesn't know which object or method will receive (handle) the events it raises. The event is typically a member of the event sender; for example, the Click event is a member of the Button class, and the PropertyChanged event is a member of the class that implements the INotifyPropertyChanged interface.

- To define an event, you use the C# event or the Visual Basic Event keyword in the signature of your event class, and specify the type of delegate for the event.
- Typically, to raise an event, you add a method that is marked as protected and virtual (in C#)
- Name this method OnEventName; for example, OnDataReceived. The method should take one parameter that specifies an event data object, which is an object of type EventArgs or a derived type. You provide this method to enable derived classes to override the logic for raising the event. A derived class should always call the OnEventName method of the base class to ensure that registered delegates receive the event.
- The following example shows how to declare an event named ThresholdReached. The event is associated with the EventHandler delegate and raised in a method named OnThresholdReached.

```
class Counter
{
   public event EventHandler ThresholdReached;
   protected virtual void OnThresholdReached(EventArgs e)
   {
      EventHandler handler = ThresholdReached;
      handler?.Invoke(this, e);
   }  // provide remaining implementation for the class
}
```

Delegates

- A delegate is a type that holds a reference to a method. A delegate is declared with a signature that shows the return type and parameters for the methods it references, and it can hold references only to methods that match its signature.
- Delegates have many uses in .NET. In the context of events, a delegate is an intermediary (or pointer-like mechanism) between the event source and the code that handles the event.

Event handlers

- To respond to an event, you define an event handler method in the event receiver. This method must match the signature of the delegate for the event you are handling.

```
class Program
{
    static void Main()
    {
        var c = new Counter();
        c.ThresholdReached += c_ThresholdReached;
        // provide remaining implementation for the class
    }
    static void c_ThresholdReached(object sender, EventArgs e)
    {
        Console.WriteLine("The threshold was reached.");
    }
}
```

Static and dynamic event handlers

1. C# is slower to run. This is somewhat taken care of when using WPF, although currently the launching of WPF application is still a bit slow. However, after the program is launched, the animation effects are all very smooth.

2. C# is less flexible than C++. C# depends greatly on .NET framework, anything that is not found in the .NET framework will be difficult to implement.

Raising multiple events

1. If your class raises multiple events, the compiler generates one field per event delegate instance. If the number of events is large, the storage cost of one field per delegate may not be acceptable. For those situations, .NET provides event properties that you can use with another data structure of your choice to store event delegates.

For more info: [Events, Protocols and Delegates in Xamarin.iOS - Xamarin | Microsoft Docs]

Any Questions?