

Windows Presentation Foundation

Lesson 5: WPF Event Model

Lesson Objectives


- In this lesson, you will learn:
 - WPF Event Model
 - Event Routing, Bubbling & Tunneling



5.1: WPF Event Model

Event Handling

- Events are messages that are sent by an object (such as a WPF element) to notify the code when something significant occurs
- WPF classes define events where handlers can be added.
 - Example: MouseEnter, MouseLeave, MouseMove, Click, and so on
- This is based on the events and delegates mechanism on .NET.

Capgemini
CONNECTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 3

WPF Design Principles: Event Handling in WPF:

Event Handling:

As you have already learnt, a Button can contain graphics, list boxes, another button, and so on.

What happens if a CheckBox is contained inside a Button, and you click the CheckBox? Where should the event arrive?

The answer is that the event is bubbled. First, the Click event arrives with the CheckBox, and it then bubbles up to the Button. This way you can handle the Click event for all elements that are inside the Button with the Button.

5.1: WPF Event Model

Event Handling

- With WPF, you can assign the event handler either with XAML or in the code behind

5.1: WPF Event Model

Event Handling

- The event-handling mechanism for WPF is based on .NET events. However, WPF enhances the .NET event model with a new concept of event routing
- The event-handling mechanism for WPF is based on .NET events but extended with bubbling and tunneling features
- Some events are tunneling events, others are bubbling events



Copyright © Capgemini 2015. All Rights Reserved 5

WPF Design Principles: Event Handling in WPF:

Event Handling (contd.):

As you have already learnt, a Button can contain graphics, list boxes, another button, and so on. What happens if a CheckBox is contained inside a Button and you click the CheckBox? Where should the event arrive?

The answer is that the event is bubbled. First, the Click event arrives with the CheckBox, and it then bubbles up to the Button. In this way, you can handle the Click event for all elements that are inside the Button with the Button.

5.1: WPF Event Model

Types of Routed Events

- Bubbling events are events that travel up the containment hierarchy
 - For example, MouseDown is a bubbling event
- It is raised first by the element that is clicked
- Next, it is raised by that element's parent, and then by that element's parent, and so on, until WPF reaches the top of the element tree

5.1: WPF Event Model

Types of Routed Events

- Tunneling events are events that travel down the containment hierarchy
- They give you the chance to preview (and possibly stop) an event before it reaches the appropriate control
 - For example, PreviewKeyDown allows you to intercept a key press
- First at the window level, and then in increasingly more specific containers until you reach the element that had focus when the key was pressed

5.1: WPF Event Model

Types of Routed Events

- When you register a routed event using the `EventManager.RegisterEvent()` method, you pass a value from the `RoutingStrategy` enumeration that indicates the event behavior you want to use for your event

5.1: WPF Event Model

Event Handling

- Event routing allows an event to originate in one element but be raised by another one
 - For example, event routing allows a click that begins in a toolbar button to rise up to the toolbar and then to the containing window before it is handled by your code
- A tunneling event first arrives with the outer element and tunnels to the inner elements
- Bubbling events start with the inner element and bubble to the outer elements

5.1: WPF Event Model

Routed Events

- Just as WPF adds more infrastructure on top of the simple notion of .NET properties, it also adds more infrastructure on top of the simple notion of .NET events
- Routed events are events that are designed to work well with a tree of elements
- When a routed event is raised, it can travel up or down the visual and logical tree, getting raised on each element in a simple and consistent fashion, without the need for any custom code

5.1: WPF Event Model

Event Handling

- Tunneling and bubbling events are usually paired
- Tunneling events are prefixed with Preview, for example, PreviewMouseMove.
 - This event tunnels from the outer controls to the inner controls
- After the PreviewMouseMove event, the MouseMove event occurs
- You can stop tunneling and bubbling by setting the Handled property of the event argument to true



Copyright © Capgemini 2015. All Rights Reserved 11

WPF Design Principles: Event Handling in WPF:

Event Handling (contd.):

You can stop tunneling and bubbling by setting the Handled property of the event argument to true.

The Handled property is a member of the RoutedEventArgs class. All event handlers that participate with the tunneling and bubbling facility have an event argument of type RoutedEventArgs or a type that derives from RoutedEventArgs.

5.1: WPF Event Model

Routed Events

```
public abstract class ButtonBase : ContentControl, ...
{
    // The event definition.
    public static readonly RoutedEvent ClickEvent;
    // The event registration.
    static ButtonBase()
    {
        ButtonBase.ClickEvent = EventManager.RegisterRoutedEvent(
            "Click", RoutingStrategy.Bubble,
            typeof(RoutedEventHandler), typeof(ButtonBase));
        ...
    }
}
```



Copyright © Capgemini 2015. All Rights Reserved 12

5.1: WPF Event Model


Routed Events


```
// The traditional event wrapper.  
public event RoutedEventHandler Click  
{  
    add  
    {  
        base.AddHandler(ButtonBase.ClickEvent, value);  
    }  
    remove  
    {  
        base.RemoveHandler(ButtonBase.ClickEvent, value);  
    }  
}
```

5.1: WPF Event Model

Demo

▪ Demo of Event Handling



Capgemini
CONNECTING TECHNOLOGY. OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 14

Summary

- In this lesson, we had a look at:
 - WPF Event Model
 - What is Event Routing, Event Bubbling & Tunneling

