

Module 9

"WPF Testing and Debugging"



Agenda

- ▶ **Testing**
- ▶ Debugging
- ▶ Performance Measuring



Unit Testing WPF Applications

- ▶ Your WPF will be unit testable if you apply
 - Architectural pattern:
 - MVVM
 - Design Patterns
 - Strategy, Abstract Factory, Repository, Null Object, ...
- ▶ Unit Test
 - Model (M)
 - ViewModels (VM)
- ▶ Views:
 - Needs e.g. UIAutomation for automatic test



UI Automation

- ▶ UI testing framework
 - **System.Windows.Automation** namespace for WPF
- ▶ **XxxAutomationPeer** is the UI Automation representation of the **Xxx** control
 - **GetChildren()**
 - **GetName()**
 - **GetParent()**
 - **GetPattern()**
- ▶ Pattern interfaces (add reference to **UIAutomationProvider.dll**)
 - **IInvokeProvider**
 - **IToggleProvider**
 - + many, many more...
- ▶ Step-by-step:
 1. Create **AutomationPeer** class for control to interact with
 2. Retrieve pattern interface
 3. Interact with control through methods of the pattern interface



Agenda

- ▶ Testing
- ▶ **Debugging**
- ▶ Performance Measuring



Introducing the WPF Tree Visualizer

► Logical Tree

- View in Visual Studio with
 - **View → Other Windows → Document Outline**
 - Very small view icon 😊
- Essential for eventing

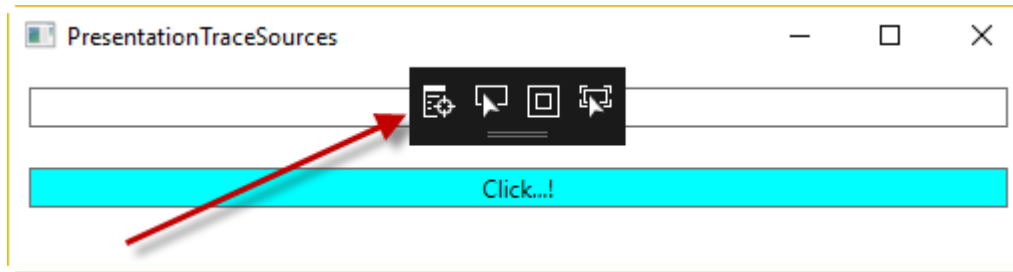
► Visual Tree

- Elements deriving from **Visual** and **Visual3D**
- View in Visual Studio with “WPF Tree Visualizer”
 - Access from Locals, Autos, or Watch window
- Essential for styling and templating



UI Debugging Tools in Visual Studio

- ▶ Visual Studio 2015 added UI Debugging Tools



- ▶ Live Visual Tree
- ▶ Enable Selection
- ▶ Display Layout Adorners
- ▶ Track Focused Element



The screenshot displays the Visual Studio IDE with two panels open: the Live Visual Tree and the Live Property Explorer.

Live Visual Tree: This panel shows a hierarchical tree of the application's visual elements. The tree is rooted at [MainWindow] (204). It includes a [Border] (203) which contains an [AdornerDecorator] (202). The [AdornerDecorator] contains a [ContentPresenter] (200), which in turn contains a [DockPanel] (199). The [DockPanel] contains a [ParticipantsView] (150), which contains another [Border] (149). This [Border] contains a [ContentPresenter] (148), which contains a [ListBox] (147). The [ListBox] contains a [Bd [Border]] (146), which contains a [ScrollViewer] (145). The [ScrollViewer] contains a [Grid] (144), which contains a [Rectangle] (123), a [ScrollContentPresenter] (17), a [ScrollBar] (16), and a [Bg [Grid]] (16). The [Bg [Grid]] contains a [PART_LineUpButton [RepeatButton]] (3), a [PART_Track [Track]] (6), and a [PART_LineDownButton [RepeatButton]] (3). The [Border] element at (146) is currently selected.

Live Property Explorer: This panel shows the properties of the selected [Border] element. The Name is "<No Name>" and the Type is "Border". The properties are organized into sections: Coercion, Local, and Inherited. The Local section includes properties like Background, BorderBrush, BorderThickness, Grid.Row, and RenderSize. The Inherited section includes properties like DataContext, SnapsToDevicePixels, and XmlAttributeProperties.XmlNameDic... The properties are displayed with their current values and a "New" button is visible at the bottom right.



Debugging Data Bindings

- ▶ Set tracing for data binding directly

```
<Window ...  
    xmlns:diag="clr-  
namespace:System.Diagnostics;assembly=WindowsBase">  
    <Button x:Name="button" Background="{Binding  
ElementName=otherButton, Path=Width,  
diag:PresentationTraceSources.TraceLevel=High}">Press...!</Butto  
n>  
    ...  
</Window>
```

```
System.Windows.Data Warning: 67 : BindingExpression  
(hash=48835636): Resolving source  
System.Windows.Data Warning: 70 : BindingExpression  
(hash=48835636): Found data context element: <null> (OK)  
...
```



.NET Trace Sources

- ▶ **Debug** and **Trace** classes in **System.Diagnostics**
- ▶ .NET 2.0 introduced the **TraceSource** class
 - **TraceSource.Switch**
- ▶ **SourceSwitch.Level** of type **SourceLevels**
 - Off
 - Critical
 - Error
 - **Warning**
 - Information
 - Verbose
 - **ActivityTracing**
 - **All**
- ▶ Can be configured programmatically or in .config file



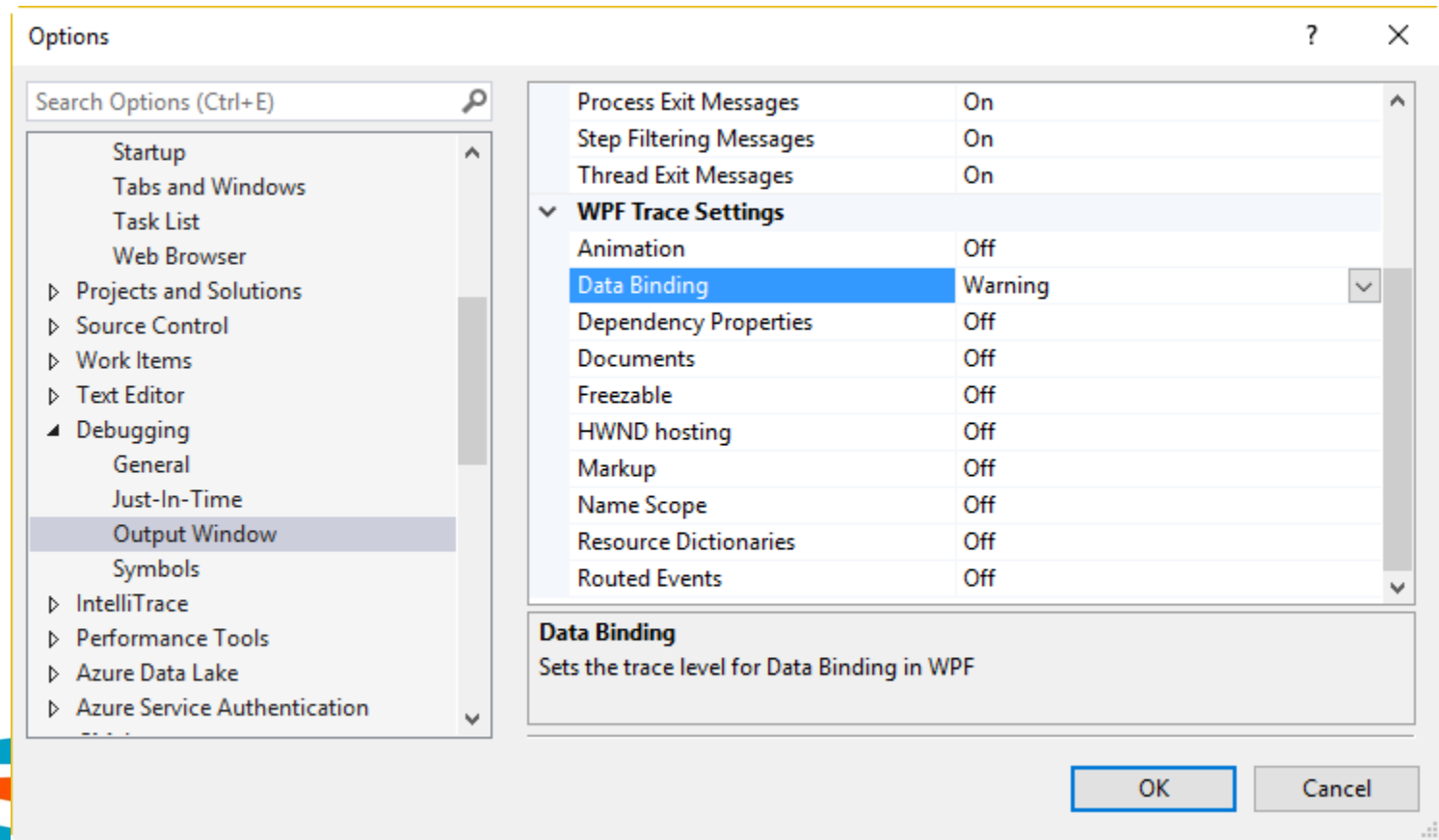
The **PresentationTraceSources** Class

- ▶ The **PresentationTraceSources** class holds all the **TraceSource** objects from WPF
 - "System.Windows.Data"
 - "System.Windows.DependencyProperty"
 - "System.Windows.RoutedEvent"
 - "System.Windows.Media.Animation"
 - "System.Windows.ResourceDictionary"
 - "System.Windows.Markup"
 - "System.Windows.Documents"
 - ...
- ▶ Initialize **PresentationTraceSources**
 - Programmatically via **PresentationTraceSources.Refresh()**



Setting Tracing in Visual Studio

- ▶ Tools -> Options -> Debugging -> Output Window -> WPF Trace Settings



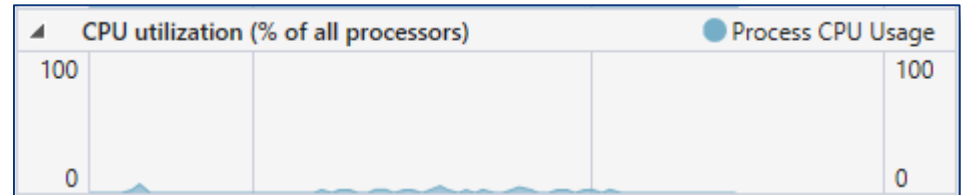
Agenda

- ▶ Testing
- ▶ Debugging
- ▶ **Performance Measuring**

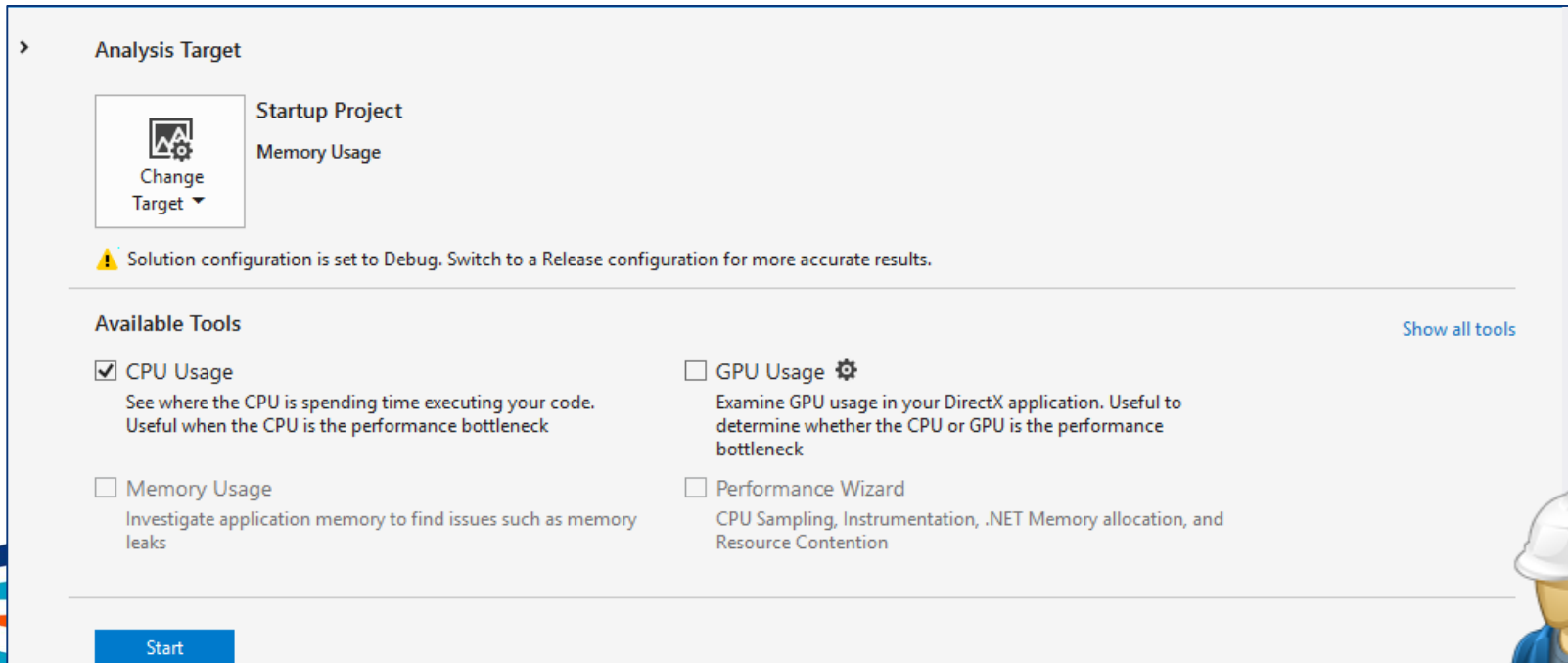


CPU Utilization

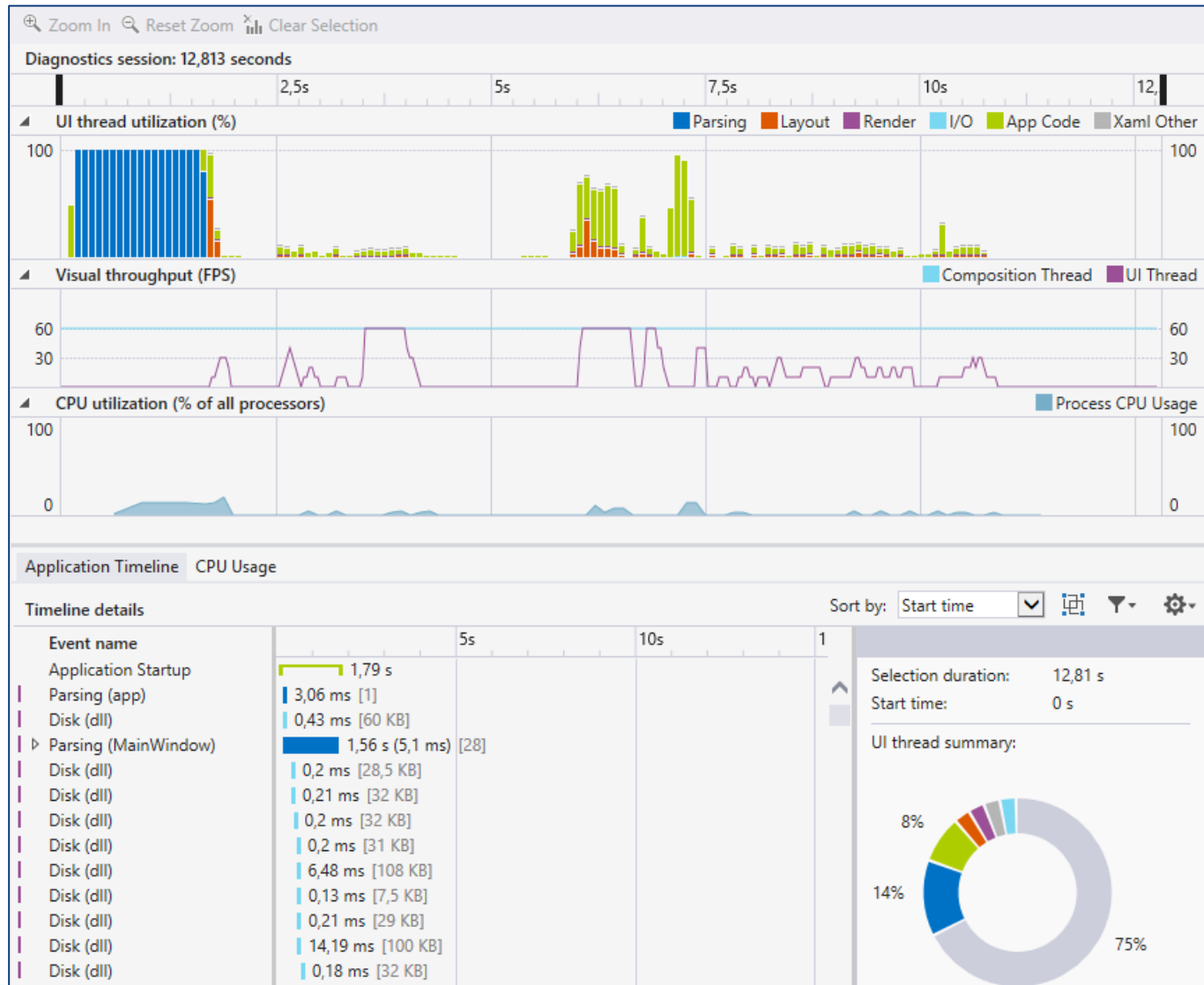
- ▶ Identify spikes
- ▶ Evaluate parallelization potential



- ▶ Start Diagnostic Tools Without Debugging...
~ ALT+F2



XAML Application Timeline Tool



Summary

- ▶ Testing
- ▶ Debugging
- ▶ Performance Measuring



