Module 8

"WPF Testing and Debugging"

TEKNOLOGISK
INSTITUT

Agenda

- Testing
- Debugging
- Performance Measuring

TEKNOLOGISK INSTITUT

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
- ***Xxx*`AutomationPeer`** 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

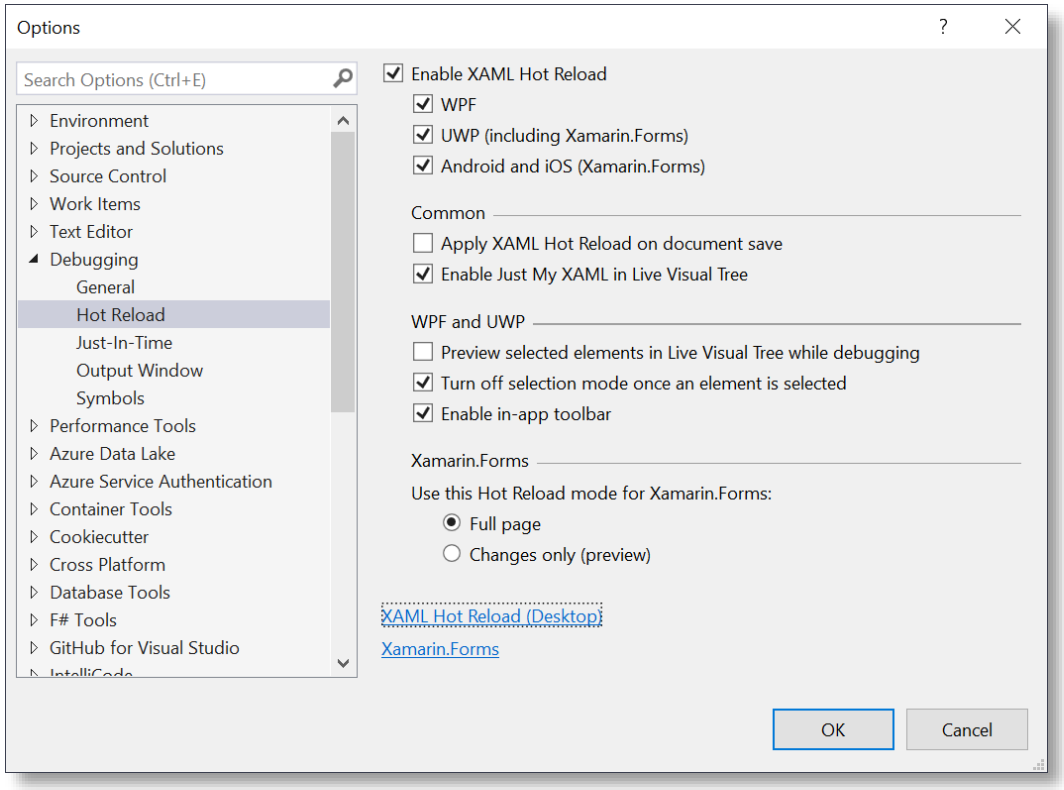
- Testing
- **Debugging**
- Performance Measuring

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

XAML Hot Reload

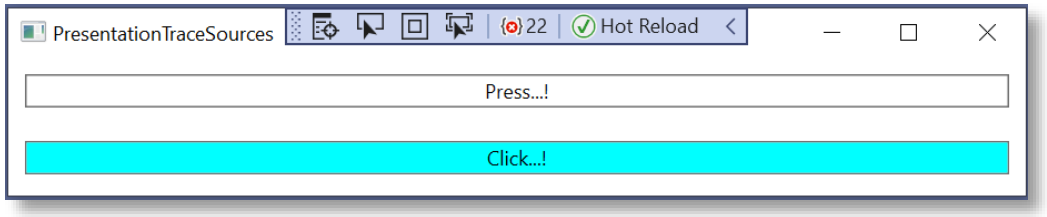
- `Tools > Options > Debugging > Hot Reload`



TEKNOLOGISK INSTITUT

UI Debugging Tools in Visual Studio

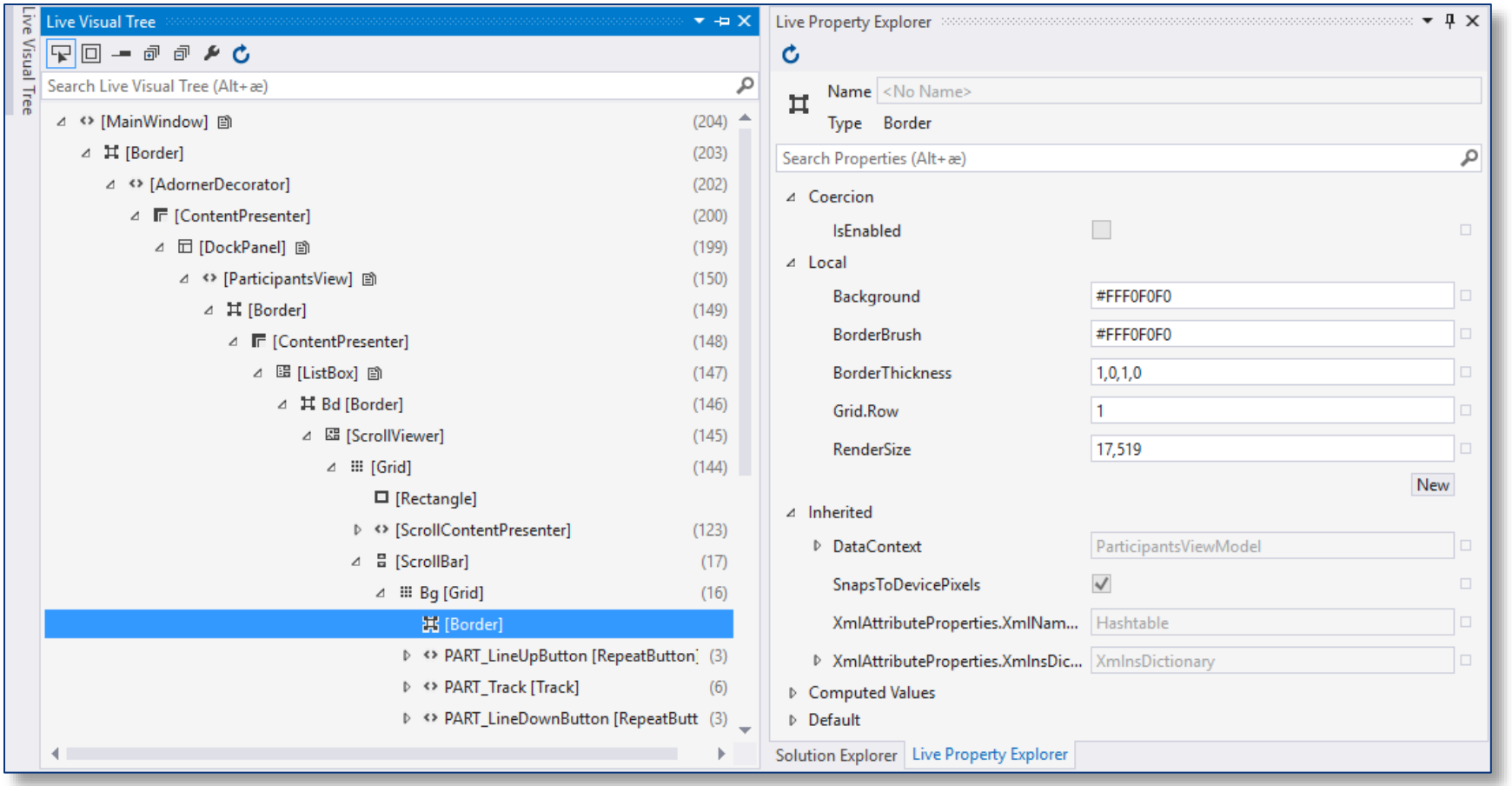
- Visual Studio 2015 added UI Debugging Tools



- Go to Live Visual Tree
- Select Element
- Display Layout Adorners
- Track Focused Element
- Data Binding Failures
- Hot Reload

XAML Live Visual Tree

- Live Visual Tree + Live Property Explorer



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

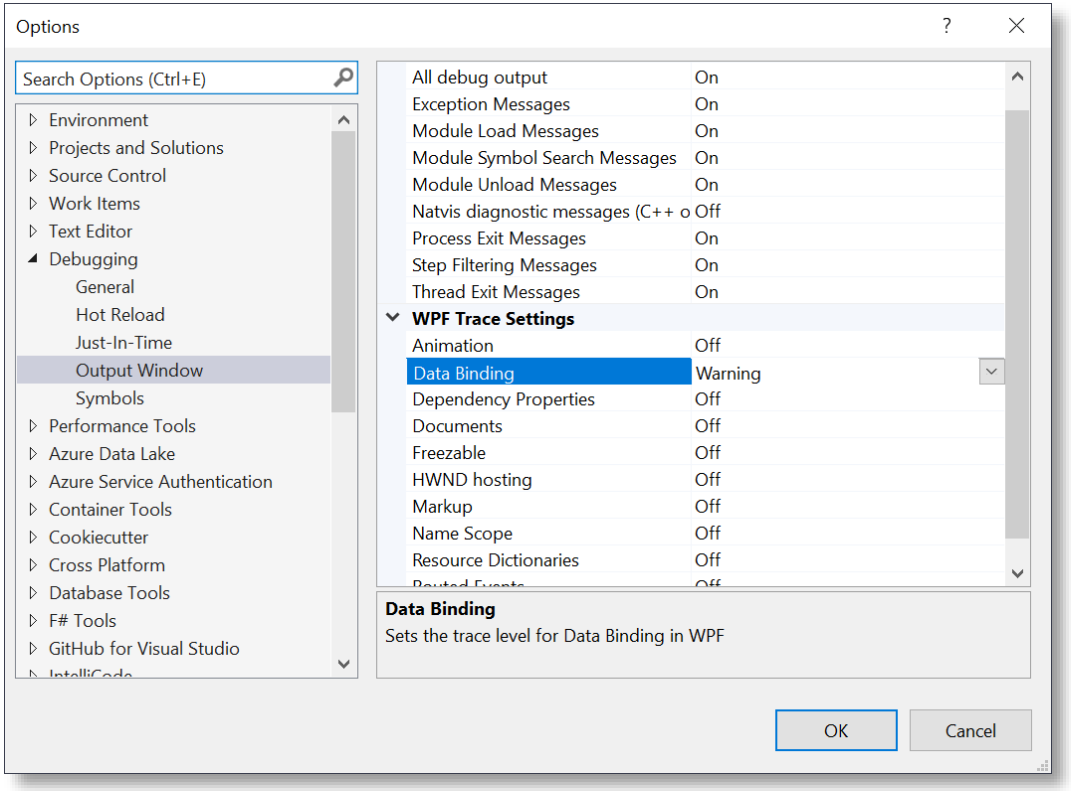
The `PresentationTraceSources` Class

- 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()`

TEKNOLOGISK INSTITUT

Setting Tracing in Visual Studio

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



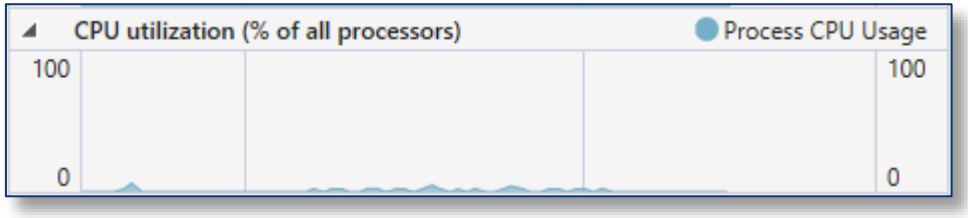
TEKNOLOGISK
INSTITUT

Agenda

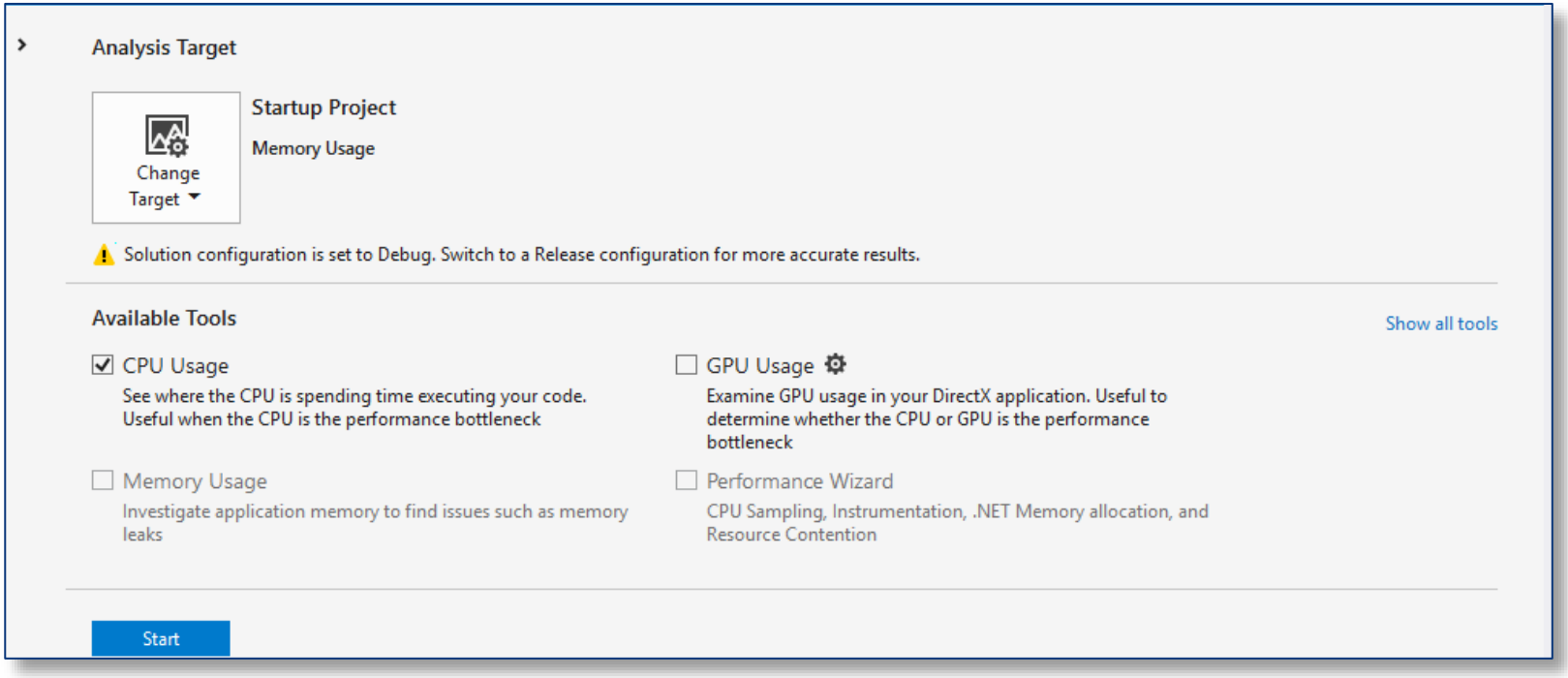
- Testing
- Debugging
- **Performance Measuring**

CPU Utilization

- Identify spikes
- Evaluate parallelization potential

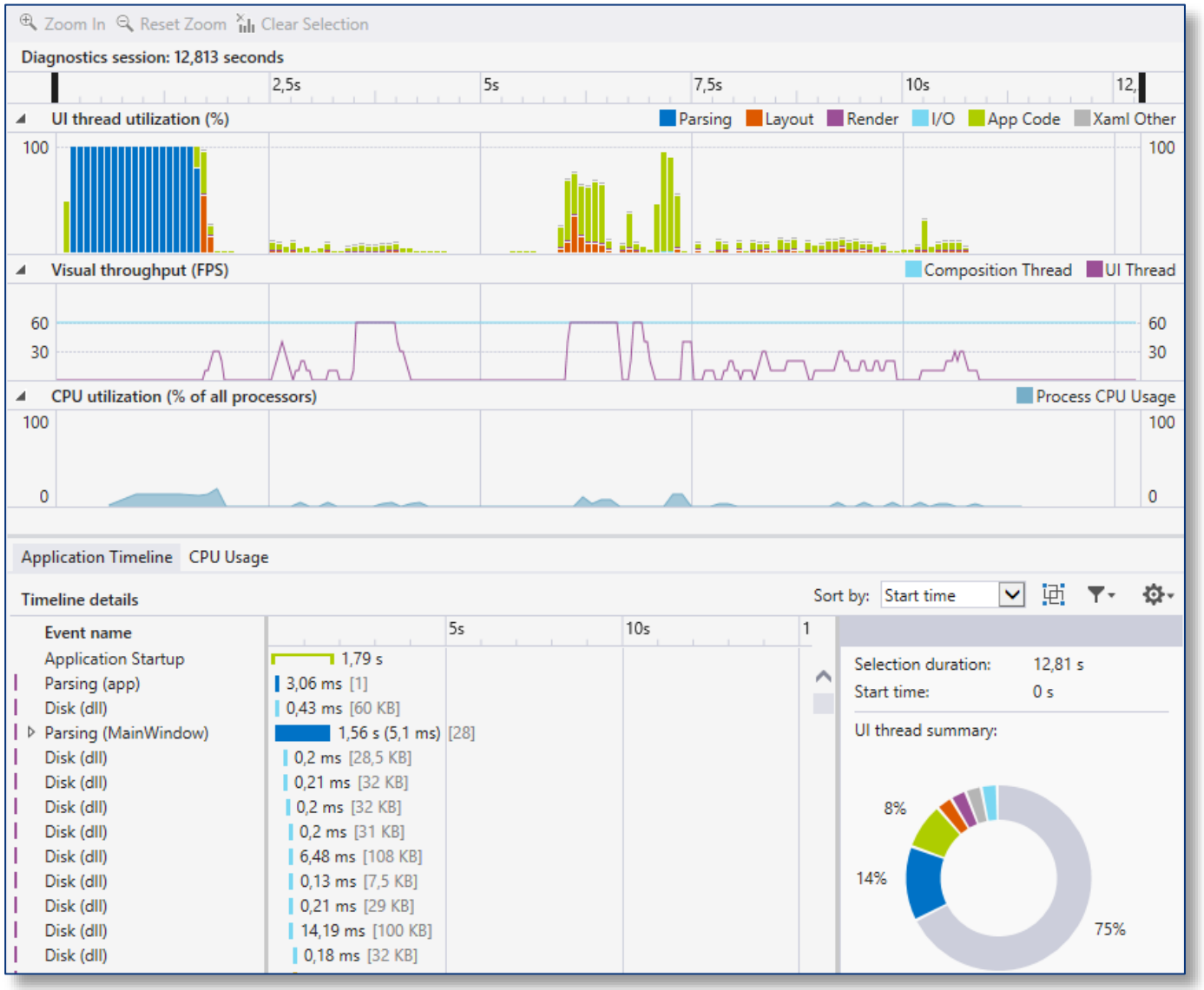


- `Start Diagnostic Tools Without Debugging…`
  `~ ALT+F2`



TEKNOLOGISK INSTITUT

XAML Application Timeline Tool



Summary

- Testing
- Debugging
- Performance Measuring


WINCUBATE
Jesper Gulmann Henriksen
PhD, MCT, MCSD, MCPD
Phone : +45 22 12 36 31
Email : jgh@wincubate.net
WWW : http://www.wincubate.net
Ringgårdsvej 4A
8270 Højbjerg
Denmark