

# Module 9

## "WPF Testing and Debugging"



**TEKNOLOGISK**  
**INSTITUT**

# 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

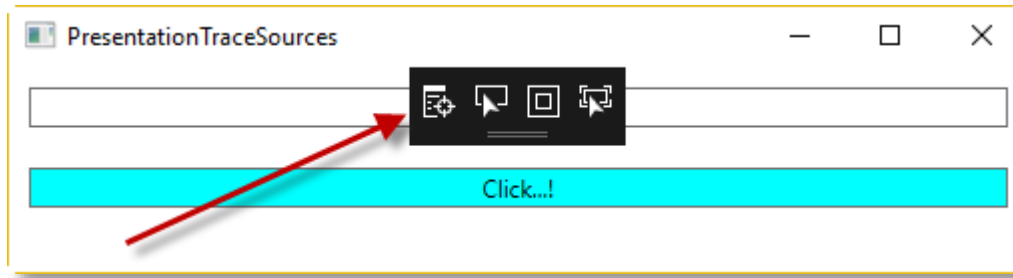
## ► Visual Tree

- 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

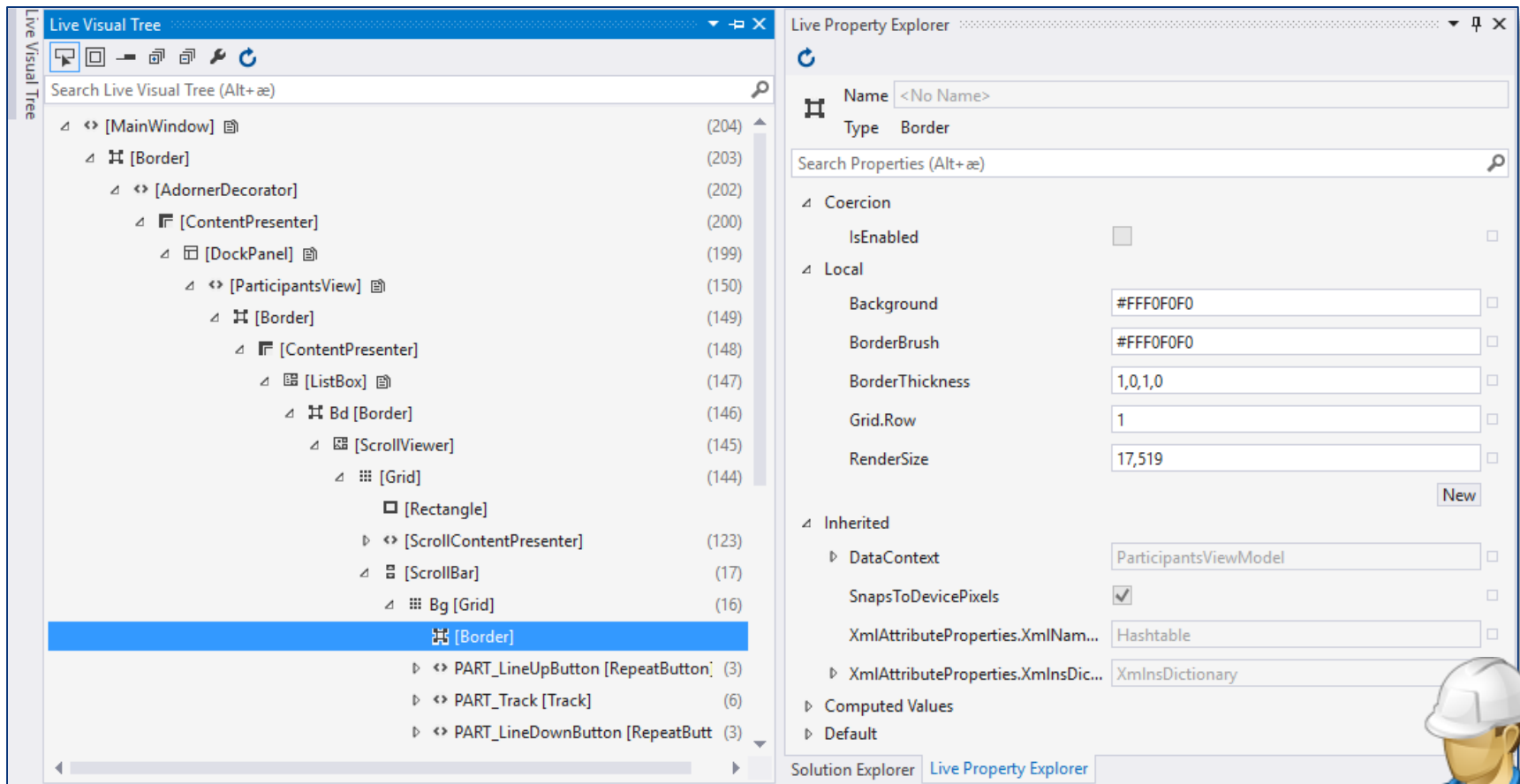
- ▶ Visual Studio 2015 added UI Debugging Tools



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

# XAML Live Visual Tree

## ► Live Visual Tree + Live Property Explorer



The screenshot displays the Visual Studio IDE with the **Live Visual Tree** and **Live Property Explorer** windows open.

**Live Visual Tree:** This window shows the hierarchical structure of the XAML UI. The tree is rooted at **[MainWindow]** (204) and includes the following elements:

- [Border]** (203)
  - [AdornerDecorator]** (202)
    - [ContentPresenter]** (200)
      - [DockPanel]** (199)
        - [ParticipantsView]** (150)
          - [Border]** (149)
            - [ContentPresenter]** (148)
              - [ListBox]** (147)
                - [Bd [Border]]** (146)
                  - [ScrollViewer]** (145)
                    - [Grid]** (144)
                      - [Rectangle]** (123)
                        - [ScrollContentPresenter]** (17)
                          - [ScrollBar]** (16)
                            - [Bg [Grid]]** (16)
                              - [Border]** (3)
                                - PART\_LineUpButton [RepeatButton]** (3)
                                  - PART\_Track [Track]** (6)
                                    - PART\_LineDownButton [RepeatButton]** (3)

**Live Property Explorer:** This window displays the properties of the selected element, **[Border]**. The properties are organized into sections:

          - Name:** <No Name>
          - Type:** Border
          - Search Properties (Alt+æ):** (Search bar)
          - Coercion:**
            - IsEnabled:** ☐
          - Local:**
            - Background:** #FFF0F0F0
            - BorderBrush:** #FFF0F0F0
            - BorderThickness:** 1,0,1,0
            - Grid.Row:** 1
            - RenderSize:** 17,519
          - Inherited:**
            - DataContext:** ParticipantsViewModel
            - SnapsToDevicePixels:** ☒
            - XmlAttributeProperties.XmlNam...:** Hashtable
            - XmlAttributeProperties.XmlInsDic...:** XmlInsDictionary
            - Computed Values:**
            - Default:**

The **Solution Explorer** at the bottom shows the **Live Property Explorer** tab selected.





# 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

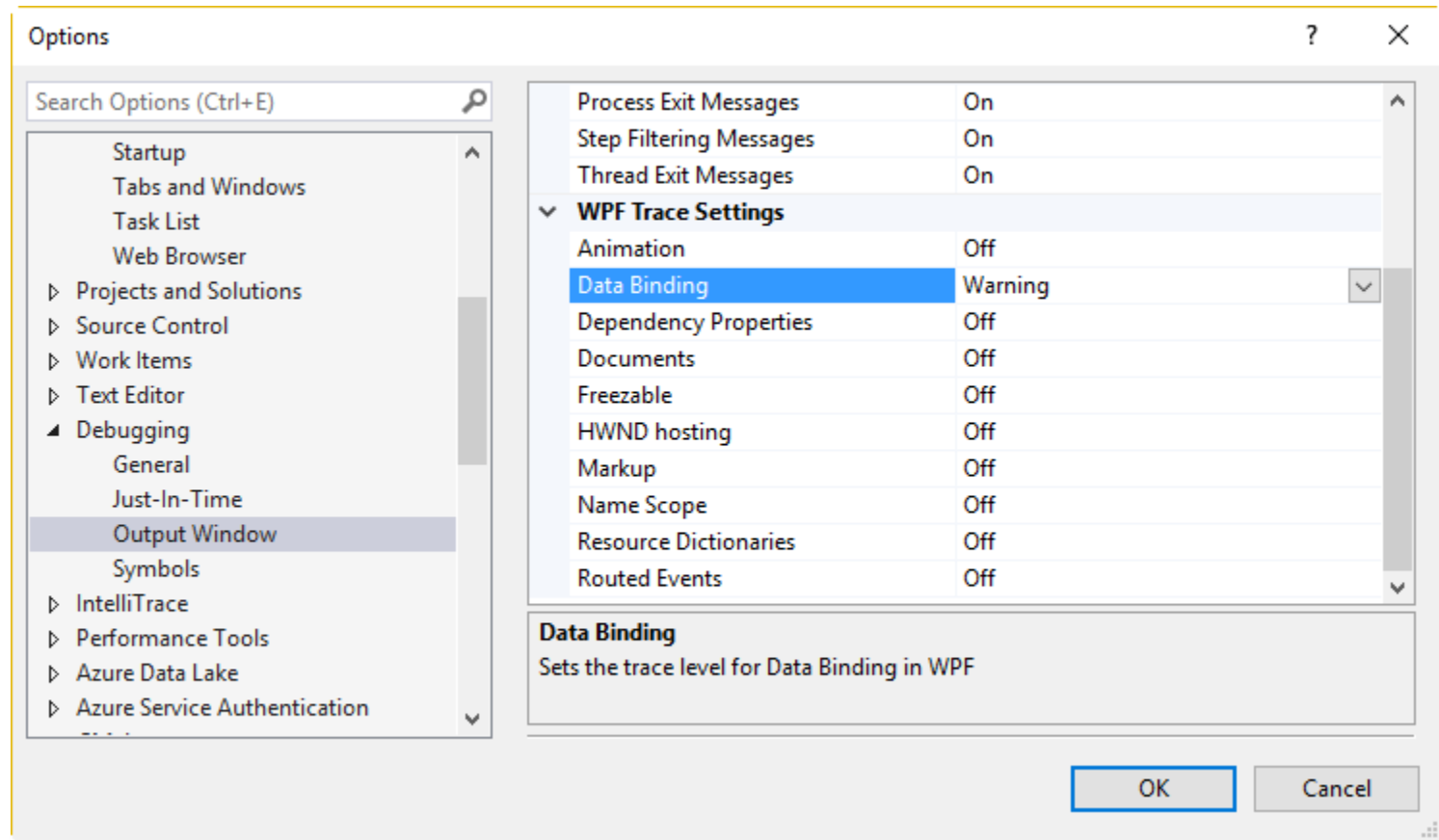
# The `PresentationTraceSources` Class

- ▶ 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



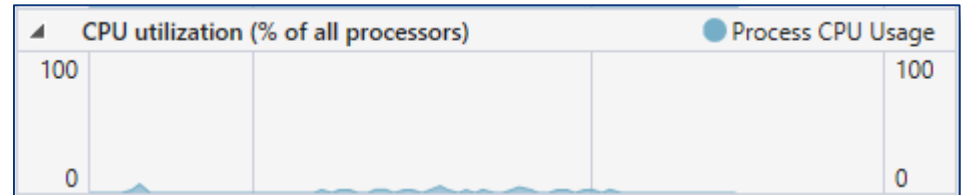


# Agenda

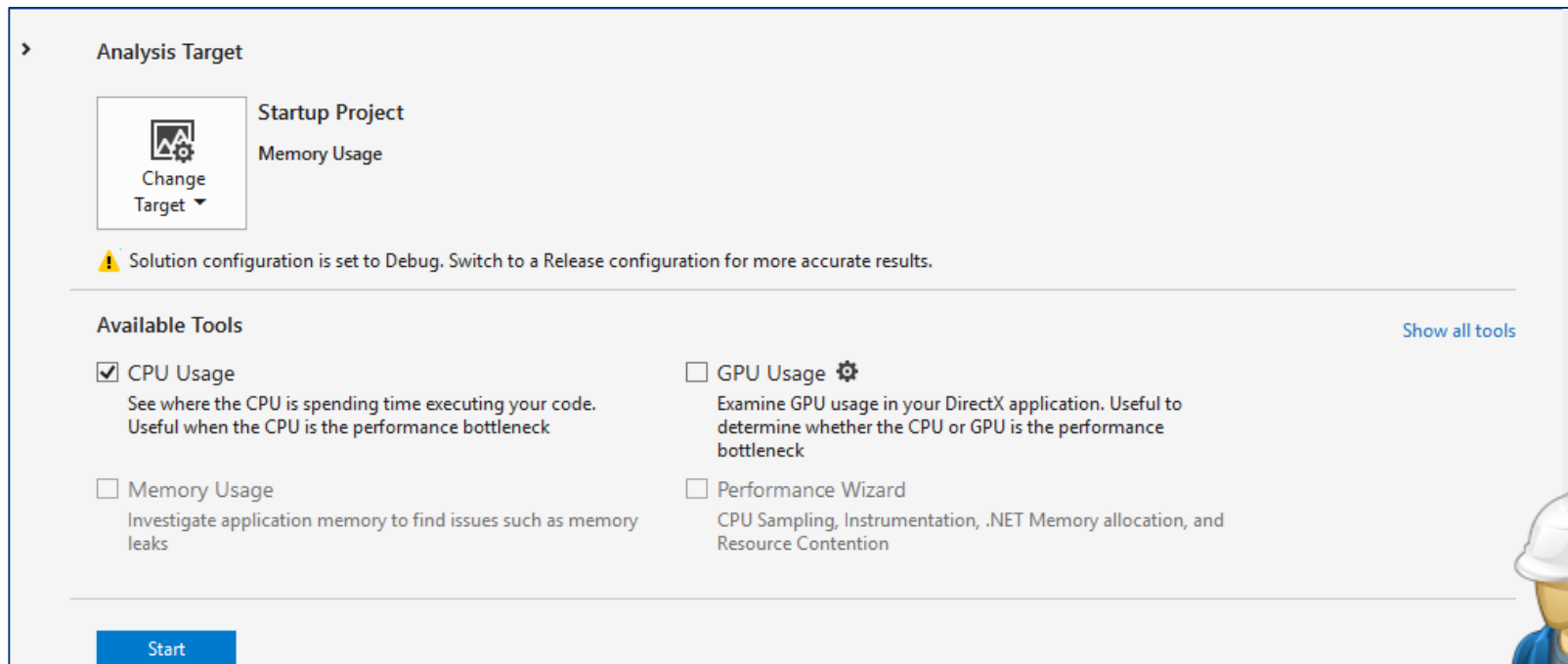
- ▶ Testing
- ▶ Debugging
- ▶ **Performance Measuring**

# CPU Utilization

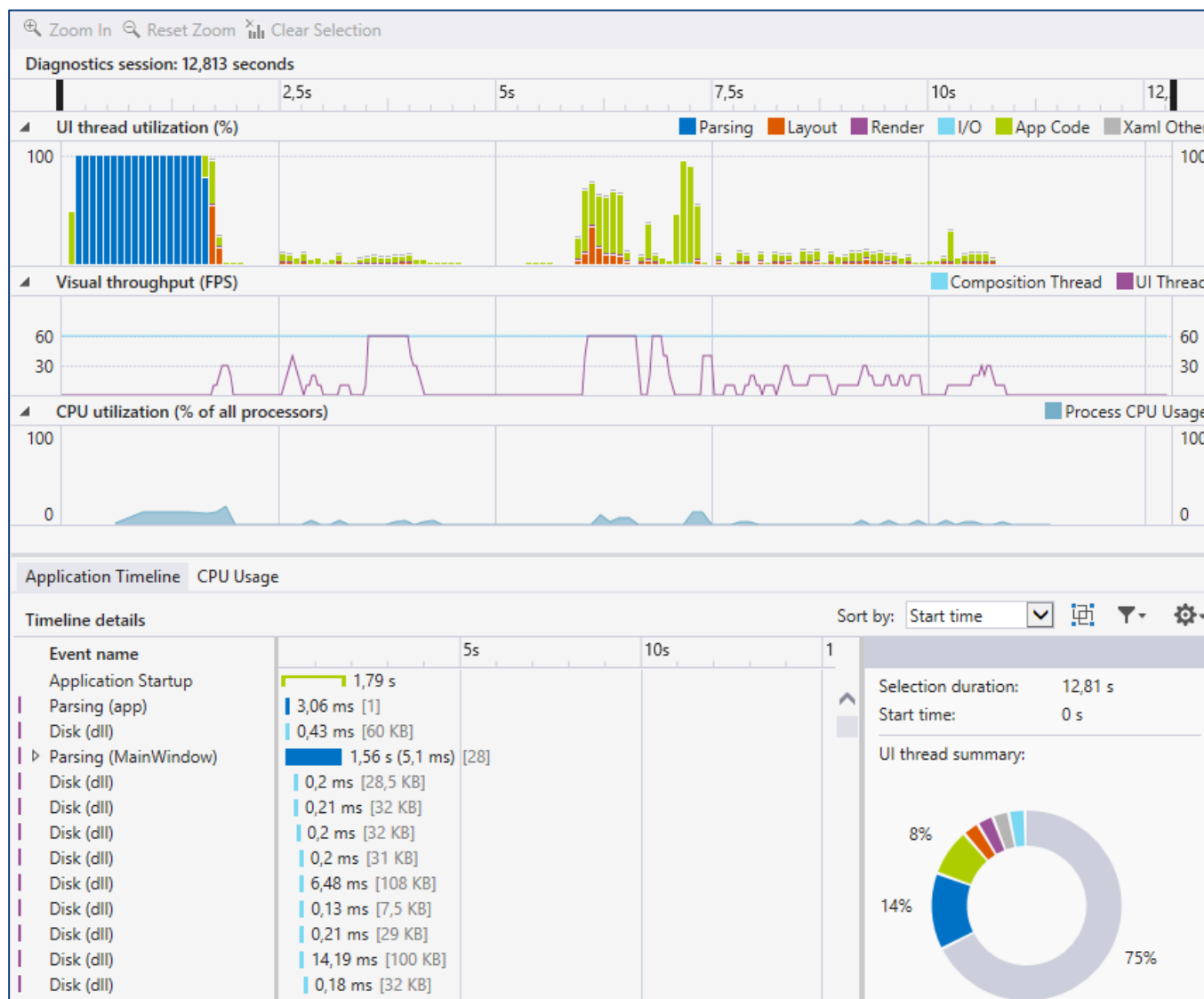
- ▶ Identify spikes
- ▶ Evaluate parallelization potential



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



# XAML Application Timeline Tool



# Summary

- ▶ Testing
- ▶ Debugging
- ▶ Performance Measuring





WINCUBATE

***Jesper Gulmann Henriksen***

PhD, MCT, MCSD, MCPD

Phone : +45 22 12 36 31

Email : [jgh@wincubate.net](mailto:jgh@wincubate.net)

WWW : <http://www.wincubate.net>

Hasselvangel 243

8355 Solbjerg

Denmark