

# Module 5

## "Control Templates and User-Defined Controls"



**TEKNOLOGISK**  
**INSTITUT**

# Agenda

- ▶ **Control Templates**
- ▶ Creating User and Custom Controls in WPF

# WPF Trees

## ► Logical Tree

- View in Visual Studio with
  - **View → Other Windows → Document Outline**
  - Bottom-left corner 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

# Introducing Control Templates

- ▶ A control template replaces the visual tree of the control
  - **Control.Template** property
- ▶ **ControlTemplate**
  - **VisualTree**
  - **TargetType**
  - **Triggers**

Content property  
Restricts use
- ▶ Can be applied
  - Directly
  - As a resource
  - Through a style
- ▶ Even though they're similar Control Templates are not the same as Data Templates!

# Triggers, Presenters, and Template Bindings

## ▶ **ControlTemplate**

- **Triggers**

## ▶ **Presenters**

- **ContentPresenter**                      Light-weight, specialized **ContentControl**
- **ItemsPresenter**                      Light-weight, specialized **ItemsControl**

## ▶ **TemplateBinding**                      Light-weight, specialized **Binding**

- Binds properties to properties of the *Templated Parent*
  - **Content** property is implicitly bound
    - Note: **ControlTemplate.TargetType** must be set
  - Used for “respecting” Templated Parent’s properties

# Tips and Tricks

- ▶ **TemplateBinding** vs. **Binding**
  - **TemplateBinding**
    - Only works within template's visual tree: not in triggers!
  - Can use data binding with **RelativeSource = TemplatedParent**
    - Conventional data binding can always be used
- ▶ Viewing built-in templates
  - Use **XamlWriter** to save template property value
- ▶ Predefined **PART\_Xxx** names in templates
  - E.g. TextBox: **PART\_ContentHost**
  - E.g. ComboBox: **PART\_Popup**
  - ...

# Visual State Manager

- ▶ You should consider all the visual states of a control when templating it!
  - You could craft all this by hand. But... ☹
- ▶ Visual State Manager (VSM)
  - Easy control over parts and states e.g. Expression Blend
  - Better than to use triggers!
- ▶ WPF controls exhibit a number of States in State Groups
  - e.g. for **Button**:
    - **CommonStates**    **Normal, MouseOver, Pressed, Disabled**
    - **FocusStates**     **Unfocused, Focused**
  - Always in one State from each State Group

# Animations and the Visual State Manager

- ▶ **VisualStateManager.VisualStateGroups** attached property on visual tree
  - **VisualStateGroup** collection
    - **VisualState** collection
      - Storyboard
    - Transitions
- ▶ **Storyboard** contains animations (See Module 1)
- ▶ **VisualStateGroup.Transitions**
  - **VisualTransition** collection
    - To optional state (matches all)
    - From optional state (matches all)
- ▶ **ViewBox** is often very helpful!



# Agenda

- ▶ Control Templates
- ▶ Creating User and Custom Controls in WPF

# Templating or Control Authoring?

- ▶ Three possibilities for custom elements
  - Templates
    - When appearance needs to be changed
  - User Controls
    - When combining existing into composite functionality
  - Custom Controls
    - When functionality is otherwise not possible
      - Derive from existing control, or
      - Derive from Control or ContentControl directly
  
- ▶ "Choose wisely" 😊

# Creating User Controls

- ▶ Use "**WPF User Control Library**" project template in Visual Studio
  - Design control surface
  - Add functionality
  - Add dependency properties
  - ...
  
- ▶ Add Reference til control library
  - **clr-namespace**
  - Use as any other control 😊

# Creating Custom Controls

- ▶ Use "**WPF Custom Control Library**" project template in Visual Studio
  - Add markup to your control
  - Add functionality
  - Add dependency properties
  - ...
  - Create **Generic.xaml** control template
  
- ▶ Add Reference til control library
  - **clr-namespace**
  - Use as any other control 😊

# Themes

## ► Themes

- Rely on OS's visual characteristics
- **Generic.xaml** must be defined!
- Optionally; *ThemeName.ThemeColor.xaml* files, e.g.
  - **Aero2.NormalColor.xaml** Windows 8
  - **Aero.NormalColor.xaml** Windows 7 + Windows Vista
  - **Luna.Metallic.xaml** Windows XP
  - **Classic.xaml** Windows Classic
- **ThemeInfoAttribute**
  - **ResourceDictionaryLocation**
    - **None**
    - **SourceAssembly**
    - **ExternalAssembly**
      - *ControlAssemblyName.ThemeName.dll*

# Summary

- ▶ Control Templates
- ▶ Creating User and Custom Controls in WPF



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>

Ringgårdsvej 4A

8270 Højbjerg

Denmark