Module 2

”Data Binding Properties”

TEKNOLOGISK
INSTITUT

Agenda

- Data Binding
- Data Conversion
- Validation
- Change Notification

The **Binding** Class

- Binding
  - ElementName, Source, RelativeSource
  - Path
  - XPath — XML sources
  - Mode
  - Delay
  - NotifyOnSourceUpdated — SourceUpdated event
  - NotifyOnTargetUpdated — TargetUpdated event
  - FallbackValue + TargetNullValue
- Binding can be set
  - Declaratively in XAML
    - Full syntax or markup extension syntax "**{}**"
  - Programmatically in code
- Bindings can be cleared (in code) via BindingOperations.ClearBinding()

RelativeSource

- You can bind to another element in the visual tree relative to the element being bound
  - use the **RelativeSource** property

- RelativeSource
  - Mode
    - FindAncestor
      - AncestorType
      - AncestorLevel
    - Self
    - PreviousData
    - TemplatedParent

Data Context

- It is possible to bind to CLR objects by
  - Referencing the namespace and assembly
  - Setting the **`Binding.Source`** property

- The can be shared among all child elements by setting the **`FrameworkElement.DataContext`** property
  - Can be set in XAML
  - Can be set in code
  - Bindings then implicitly refer to the data in **`DataContext`**
    - **`ElementName`** or **`Source`** can then be omitted

Binding Mode

- The binding mode specifies the direction of the binding

- `Binding`
  - `Mode`
    - `Default` determined by the target property
    - `OneTime`
    - `OneWay`
    - `OneWayToSource`
    - `TwoWay`
  - `UpdateSourceTrigger`
    - `Default` determined by the target property
    - `Explicit` when `BindingExpression.UpdateSource()` is called
    - `LostFocus`
    - `PropertyChanged`

Small, Important Features... ☺

- Bindings can be delayed
  - New feature in .NET 4.5

```
<Slider Name="slider"
        Value="{Binding ElementName=textbox,
                Path=Text,
                Mode=TwoWay,
                Delay=500}" />
```

- Applies for updates from target to source in two-way bindings
  - The delay does not apply for both directions!

- Data binding can be set up ”visually” through Visual Studio

Agenda

- Data Binding
- **Data Conversion**
- Validation
- Change Notification

IValueConverter

- Value converters
  - Convert, format, localize, combine data
    - Conditionally or unconditionally
  - Usually applied with Data Binding

- `IValueConverter`

• `Convert()`	Source -> Target
• `ConvertBack()`	Target -> Source

- Localization proceeds through the **`culture`** parameter
  - Defaults to **`xml:lang`**

- `ValueConversionAttribute`
  - Only conveys the intended meaning to tools
  - Has no semantic meaning!

IMultiValueConverter

- Multi-value converters
  - Performs similar conversion, but combines multiple values
- **IMultiValueConverter**
  - **Convert()**
    - Accepts **object[]** instead of just **object**
  - **ConvertBack()**
    - Returns **object[]** instead of just **object**
- Must use **MultiBinding** to apply multi-value converter!
- Side note:
  - There is also a **PriorityBinding** for use with asynchronous data binding. We will investigate that in Module 13

Data Binding Special Cases

- Converters can return special values when conversions are not possible (or not wanted)

- `Binding.`
  - `FallbackValue`
  - `DoNothing` Ignore update

- `DependencyProperty.`
  - `UnsetValue` "Errorneous" update

Agenda

- Data Binding
- Data Conversion
- Validation
- Change Notification

Validation Rules

- Data bindings are subjected to a set of validation rules
  - Exceptions are by default not thrown

- `ValidationRule`
  - `ExceptionValidationRule` built-in

- Setting `Binding.ValidatesOnExceptions` to true
  - is equivalent to adding `ExceptionValidationRule`

- Custom validation rules
  - Override `ValidationRule.Validate()`
  - `ValidationResult` signals success or failure

Handling Validation Errors

- `Validation.`
  - `HasErrors`
  - `Errors` `ValidationError` collection
  - `Error` event
    - Raised if `Binding.NotifyOnValidationError == true`

- `ValidationErrorEventArgs`
  - `Action`
    - `Added`
    - `Removed`
  - `Error`
    - `ErrorContent`
    - `Exception`
    - `RuleInError`
    - `BindingInError`

Validation Error Templates

- Controls can set attached `Validation.ErrorTemplate` property

- Note:
  - This is a `ControlTemplate` (not a `DataTemplate`)
  - `AdornedElementPlaceholder` refers back to control being validated!

IDataErrorInfo

- IDataErrorInfo in System.ComponentModel

```
public interface IDataErrorInfo
{
   string Error { get; }
   string this[ string columnName ] { get; }
}
```

- DataErrorValidationRule
  - Checks for errors raised by classes implementing **IDataErrorInfo**
- Setting **Binding.ValidatesOnDataErrors** to true
  - is equivalent to adding **DataErrorValidationRule**

- WPF 4.5 adds **INotify**DataErrorInfo for more advanced uses
  - There is a similar **Binding.ValidatesOnNotify**DataErrors

Agenda

- Data Binding
- Data Conversion
- Validation
- **Change Notification**

INotifyPropertyChanged

- Implement **`INotifyPropertyChanged`** to propagate modifications to a single element through data binding
  - **`PropertyChanged`** event
    - Raise event with CLR property name whenever it is changed

Caller Info Attributes

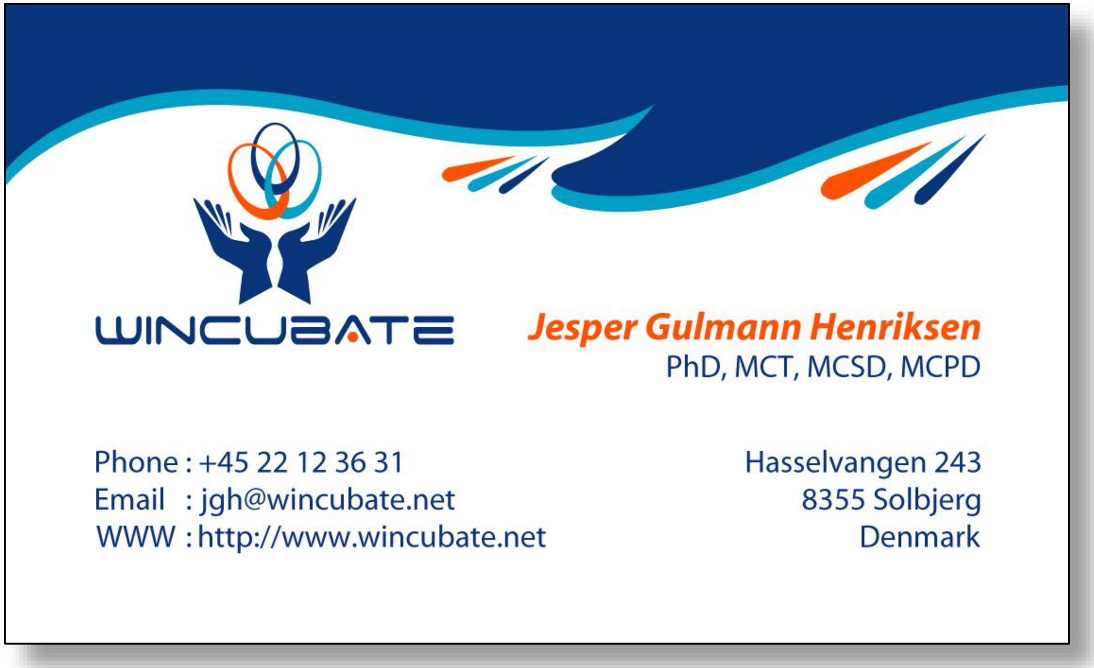
- C# 5.0 introduced three types of caller info attributes
  - `[CallerMemberName]`
  - `[CallerFilePath]`
  - `[CallerLineNumber]`

```
static void Log(
         [CallerMemberName] string callerName = null,
         [CallerFilePath] string callerFilePath = null,
         [CallerLineNumber] int callerLine = -1 )
{
   ...
}
```

- Applicable to default parameters
  - Compiler replaces values at compilation time
  - In `System.Runtime.CompilerServices`
- Perfect for **`INotifyPropertyChanged`**...! ☺

Summary

- Data Binding
- Data Conversion
- Validation
- Change Notification



WINCUBATE

Jesper Gulmann Henriksen
PhD, MCT, MCSD, MCPD

Phone : +45 22 12 36 31
Email : jgh@wincubate.net
WWW : http://www.wincubate.net

Hasselvangen 243
8355 Solbjerg
Denmark