Validation

Validation of user input
when using data binding
in WPF
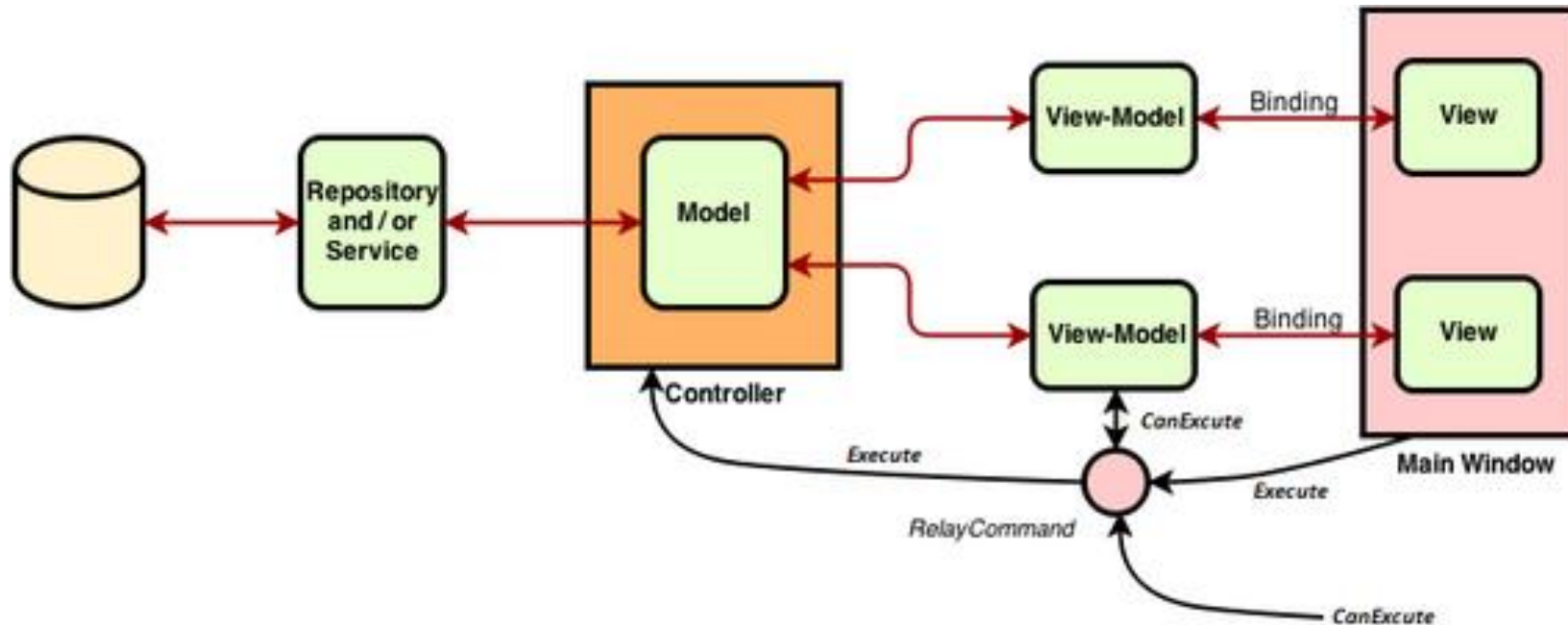
What is Validation?

- A key ingredient in any data binding scenario is ***validation***
    - *logic that catches incorrect values and refuses them*

- Some will differ between "validation rules" and what is considered "business rules"
    - But users do not make such a distinction
    - Both are just "rules" to them, some simple, some complex

- WPF provides a validation feature that works closely with the data binding system

AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

WPF's validation features

- You can raise errors in your data object (ViewModel):
  - Simply throw an exception from a property set procedure
  - Or implement the INotifyDataErrorInfo or IDataErrorInfo interface in your data class (ViewModel)

- You can define validation at the binding level:
  - If you define your validation in a distinct class, you can easily reuse it with multiple bindings that store similar types of data

AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

A typical WPF application Architecture

Use of Exceptions

- Validation in the Data Object with Exceptions

```csharp
public class Person
{
  private int _age;
  public int Age
  {
    get { return _age; }
    set
    {
      if (value < 0)
      {
        throw new ApplicationException("Age cannot be negative."
        );
      }
      _age = value;
    }
  }
…
```

Use of Exceptions

- WPF quietly ignores data binding errors that occur when setting or getting properties!
  - So **you must explicit tell WPF to show the error**
- If the **ValidatesOnExceptions** binding attribute is set to True, the framework will check for exceptions thrown during the update of the underlying data in the view-model
- If an exception is thrown, the framework will add a system ValidationError to the ValidationErrors collection associated with that control
- And if the **NotifyOnValidationError** binding attribute is also set to True, the framework will raise a routed ErrorEvent

Reacting to Validation Errors

- To provide more information, you can handle the Error event, which fires whenever an error is stored or cleared

- Show a simple MessageBox:

```
<Grid Validation.Error="ValidationError">
```

```csharp
private void ValidationError(object sender, ValidationErrorEventArgs e)
{
  // Check that the error is being added (not cleared).
  if (e.Action == ValidationErrorEventAction.Added)
  {
    MessageBox.Show(e.Error.ErrorContent.ToString());
  }
}
```

AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

Showing a ToolTip

- You might want to show the error message in a tooltip for the Border or TextBox itself so that the error message appears when the user moves the mouse over any portion of the control

```xml
<Style TargetType="{x:Type TextBox}">
...
  <Style.Triggers>
    <Trigger Property="Validation.HasError" Value="True">
      <Setter Property="ToolTip"
        Value="{Binding RelativeSource={RelativeSource Self},
        Path=(Validation.Errors)[0].ErrorContent}" />
    </Trigger>
  </Style.Triggers>
</Style>
```

AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

Showing a DifferenError Indicator

- Create your own error template that flags errors the way you want
- Error templates use the *adorner layer*, which is a drawing layer that exists just above ordinary window content

```xml
<Style TargetType="{x:Type TextBox}">
  <Setter Property="Validation.ErrorTemplate">
    <Setter.Value>
      <ControlTemplate>
        <DockPanel LastChildFill="True">
          <TextBlock DockPanel.Dock="Right" Foreground="Red" FontSize="14"
            FontWeight="Bold" ToolTip="{Binding ElementName=adornerPlaceholder,
            Path=AdornedElement.(Validation.Errors)[0].ErrorContent}">*</TextBlock>
          <Border BorderBrush="Red" BorderThickness="2">
            <AdornedElementPlaceholder Name="adornerPlaceholder">
            </AdornedElementPlaceholder>
          </Border>
        </DockPanel>
      </ControlTemplate>
    </Setter.Value>
  </Setter>
```

References & Links

- How to: Implement Binding Validation
  https://msdn.microsoft.com/en-us/library/ms753962(v=vs.110).aspx

- **Data validation in WPF**
  https://blog.magnusmontin.net/2013/08/26/data-validation-in-wpf/