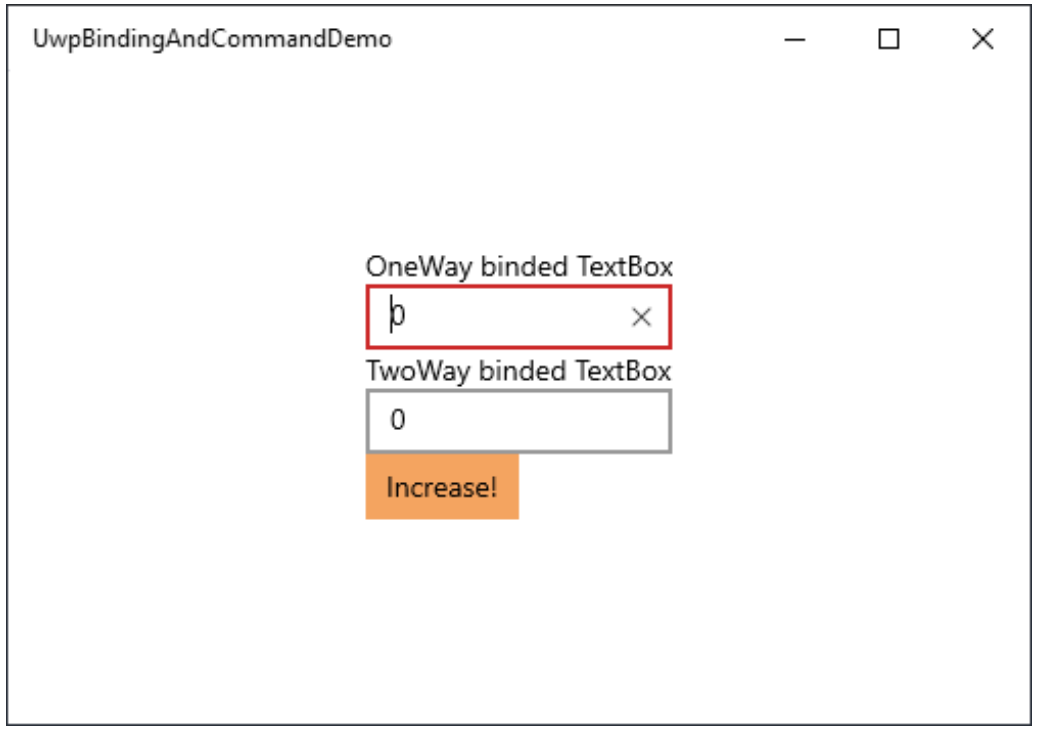
Adatkötés (data binding)

Csorba Kristóf

INotifyPropertyChanged (ism.)

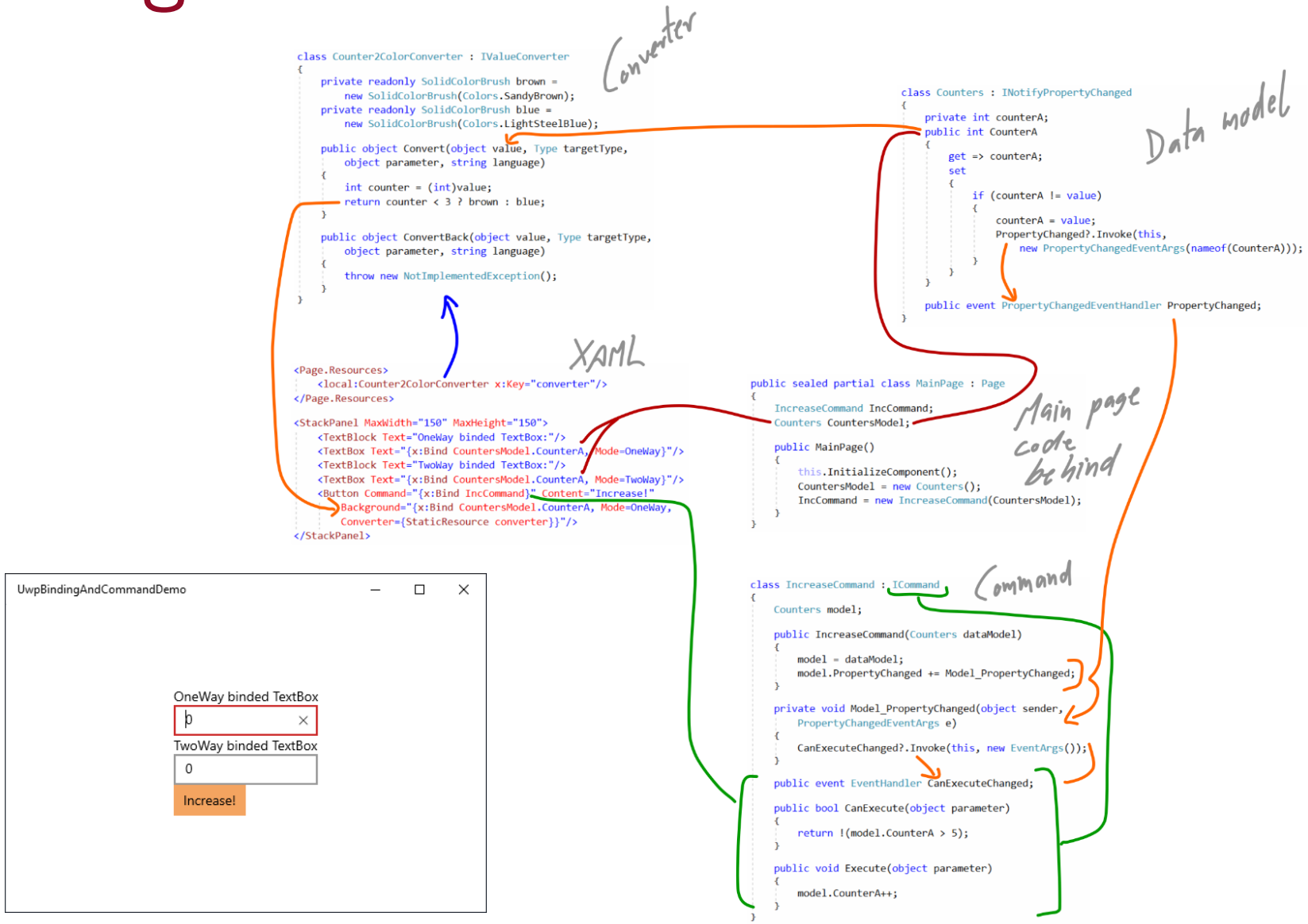
A view és a model

- View:
  a megjelenítés
- Model:
  amit megjelenítünk



Honnan tudja a view, hogy újra rá kell néznie a model értékére, mert megváltozott?

UwpBindingAndCommandDemo



Sima OneWay data binding

```
// EVIP: image for data binding
public WriteableBitmap Thumbnail { get; set; }
```

```
<ListView.ItemTemplate>
    <DataTemplate x:DataType="vm:AreaViewModel">
        <StackPanel Orientation="Horizontal" BorderBrush="DarkGray"
                    BorderThickness="1" Spacing="10">
            <Button Content="Show" VerticalAlignment="Stretch"
                    Command="{x:Bind ShowCommand}"/>
            <Image Source="{x:Bind Thumbnail, Mode=OneWay}"
                   Width="50" Height="50"/>
            <TextBlock Text="{x:Bind AsString, Mode=OneWay}"
                       VerticalAlignment="Center"/>
        </StackPanel>
    </DataTemplate>
</ListView.ItemTemplate>
```

Mandelbrot\FavoriteMandelbrots\ViewModel\AreaViewModel.cs
Mandelbrot\FavoriteMandelbrots\View\MainViewer.xaml

IValueConverter

```
// EVIP: IValueConverter
public class ColorToBrushConverter : IValueConverter
{
    public object Convert(object colorValue, Type targetType,
        object parameter, string language)
    {
        return new SolidColorBrush((Color)colorValue);
    }

    public object ConvertBack(object brushValue, Type targetType,
        object parameter, string language)
    {
        return ((SolidColorBrush)brushValue).Color;
    }
}
```

```
shape.SetBinding(Shape.FillProperty,
    new Binding() { Path = new PropertyPath("ColorPickerToBindTo.Color"),
        Converter = converter });
```

IValueConverter

```
public class OwnerIndex2BrushConverter : IValueConverter
{
    // Note: reusing brushes for all calls,
    //  using array instead of switch in Convert.
    readonly private static SolidColorBrush red = new SolidColorBrush(Colors.Coral);
    readonly private static SolidColorBrush blue = new SolidColorBrush(Colors.CornflowerBlue);
    readonly private static SolidColorBrush gray = new SolidColorBrush(Colors.LightGray);
    readonly private static SolidColorBrush[] burshes = new [] { gray, red, blue };

    public object Convert(object value, Type targetType, object parameter, string language)
    {
        // EVIP: avoiding switch with an array: field owner (int) -> Brush
        return burshes[(int)value];
    }

    public object ConvertBack(object value, Type targetType, object parameter, string language)
    {
        throw new NotImplementedException();
    }
}
```

AttaxxPlus\AttaxxPlus\View\OwnerIndex2BrushConverter.cs

Adatkötés Geometryhez

```
<ItemsControl Background="White" ItemsSource="{x:Bind SpirographVMs}" >
    <ItemsControl.ItemsPanel>
        <ItemsPanelTemplate>
            <Canvas />
        </ItemsPanelTemplate>
    </ItemsControl.ItemsPanel>
    <ItemsControl.ItemTemplate>
        <DataTemplate x:DataType="vm:SpirographVM">
            <Path Stroke="black" StrokeThickness="1"
                  Data="{x:Bind Path=Geometry, Mode=OneWay}"/>
        </DataTemplate>
    </ItemsControl.ItemTemplate>
</ItemsControl>
```

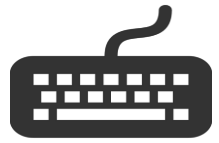
SpirographLab\SpirographViewer\Views\SpirographView.xaml

Egyebek: binding kódból

```
// EVIP: data binding from code, DependencyProperties
shape.DataContext = this;
shape.SetBinding(Shape.FillProperty,
    new Binding() { Path = new PropertyPath("ColorPickerToBindTo.Color"),
        Converter = converter });
shape.SetBinding(Shape.WidthProperty,
    new Binding() { Path = new PropertyPath("WidthSliderToBindTo.Value") });
shape.SetBinding(Shape.HeightProperty,
    new Binding() { Path = new PropertyPath("HeightSliderToBindTo.Value") });
```

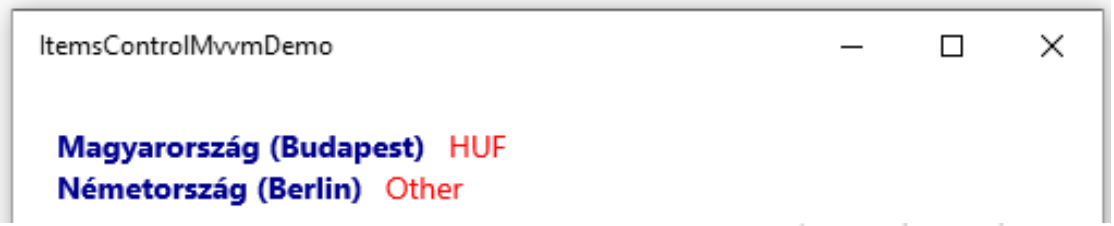
UwpShapeDataBinding\ShapeDataBinding\View\ShapeConfigurator.xaml.cs

ItemsControlMvvmDemo



- ItemsControl
- DataTemplate

```
<ItemsControl ItemsSource="{x:Bind Lands}">
    <ItemsControl.ItemTemplate>
        <DataTemplate x:DataType="vm:LandViewModel">
            <StackPanel Orientation="Horizontal" Spacing="10">
                <TextBlock Text="{x:Bind LandNameAndCapital}"
                           FontWeight="Bold" Foreground="DarkBlue"/>
                <TextBlock Text="{x:Bind HufOrOther}"
                           Foreground="Red"/>
            </StackPanel>
        </DataTemplate>
    </ItemsControl.ItemTemplate>
</ItemsControl>
```




```
public string LandNameAndCapital =>
    $"{land.Name} ({land.Cities.Single(c=>c.IsCapital).Name})";
public string HufOrOther =>
    (land.Currency == "HUF") ? "HUF" : "Other";
```

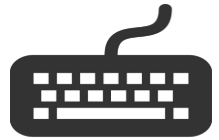
ItemsControl, DataTemplate

```
// EVIP: need to bind to a generic container, using wrapper class
//  without additional content. We cannot bind an ItemControl element
//  to List<FieldViewModel> so we create a wrapper class.
public class FieldViewModelList : ObservableCollection<FieldViewModel>
```

```
<ItemsControl.ItemTemplate>
    <DataTemplate x:DataType="vm:FieldViewModelList">
        <!-- EVIP: Now ItemsSource will refer to the elements
            ObservableCollection<FieldViewModel>, so no path r
        <ItemsControl ItemsSource="{x:Bind Mode=OneWay}">
            <ItemsControl.ItemTemplate>
                <DataTemplate x:DataType="vm:FieldViewModel">
                    <!-- EVIP: Button without content. Margin
                    <Button Command="{x:Bind FieldCommand, Moc
                            MinHeight="40" MinWidth="40" Margi
                            Background="{x:Bind Owner, Mode=Or
                            BorderBrush="{x:Bind IsSelected, M
                </DataTemplate>
            </ItemsControl.ItemTemplate>
        </ItemsControl>
    </DataTemplate>
</ItemsControl.ItemTemplate>
```

AttaxxPlus\AttaxxPlus\ViewModel\FieldViewModelList.cs

TemplateSelector @SpirographLab



- SpirographLab\SpirographViewer\
  TemplateSelector\
  TransformationTemplateSelector.cs

Egyebek: bindolás a user controlon kívülre

```
// EVIP: Exposing private controls for data binding
//  from outside this class. The wrapped controls are private
//  which does not allow data binding from other controls.
public ColorPicker ColorPickerToBindTo => this.ColorPicker;
public Slider WidthSliderToBindTo => this.WidthSlider;
public Slider HeightSliderToBindTo => this.HeightSlider;
```

```
<Slider x:Name="WidthSlider"
        Margin="0,0,0,0" Header="Width"
        Value="50" Minimum="10" Maximum="100"
        RelativePanel.AlignLeftWithPanel="True"
        RelativePanel.AlignRightWithPanel="True"
        RelativePanel.AlignTopWithPanel="True"/>
```

UwpShapeDataBinding\ShapeDataBinding\View\ShapeConfigurator.xaml.cs