Rajzolás és egyéb UI elemek

Csorba Kristóf

Color, Brush

```
Fill = new SolidColorBrush(Colors.Red)
```

```
<ImageBrush ImageSource="forest.jpeg"/>
```

```
private static Color CreateColor(float r, float g, float b)
{
    // EVIP: creating a Color from ARGB values.
    //  Static factory method inside Color.
    return Color.FromArgb(255, (byte)(r * 255.0),
        (byte)(g * 255.0), (byte)(b * 255.0));
}
```

Mandelbrot\MandelbrotCommon\HsvRgbConverter.cs

Brush újrahasznosítás

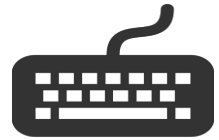
```
public class IsSelected2BrushConverter : IValueConverter
{
    // EVIP: reusing brushes, named constants
    readonly private static SolidColorBrush yellow = new SolidColorBrush(Colors.Yellow);
    readonly private static SolidColorBrush gray = new SolidColorBrush(Colors.Gray);

    public object Convert(object value, Type targetType, object parameter, string language)
    {
        return ((bool)value) ? yellow : gray;
    }
```

AttaxxPlus\AttaxxPlus\View\IsSelected2BrushConverter.cs

Shape és leszármazottjai

- Online: https://docs.microsoft.com/en-us/windows/uwp/design/controls-and-patterns/shapes
- Rectangle, Ellipse
- Polygon, Polyline
- Path

DrawingAndShapesDemo



- Rectangle, Ellipse, Polygon
- StackPanel és Canvas
- XAML és Code behind
- Attached properties

Path

```
<Path Stroke="black" StrokeThickness="1"
      Data="{x:Bind Path=Geometry, Mode=OneWay}"/>
```

```
public PathGeometry CreateGeometry(SPoint[] points)
{
    var pathFigure = CreatePathFigure(points);
    var geometry = new PathGeometry();
    geometry.Figures.Add(pathFigure);
    return geometry;
}

public PathFigure CreatePathFigure(SPoint[] points)
{
    var figure = new PathFigure() { IsFilled = false, IsClosed = true };
    if (points.Length > 0)
    {
        figure.StartPoint = PointFromSPoint(points[0]);
        for (int i = 1; i < points.Length; i++)
        {
            var segment = new LineSegment()
            {
                Point = PointFromSPoint(points[i])
            };
            figure.Segments.Add(segment);
        }
    }
    return figure;
}
```

SpirographLab\SpirographViewer\Views\SpirographView.xaml
SpirographLab\SpirographViewer\Helpers\StrokeGeometryBuilder.cs

BitmapImage osztály (AttaxxPlus boosters)

```
protected void LoadImage(Uri imageFileUri)
{
    // EVIP: loading an image from Uri
    Image = new BitmapImage
    {
        UriSource = imageFileUri
    };
}
```

```
<Button Command="{x:Bind }" BorderBrush="Gray" BorderThickness="2"
        Background="LightGray">
    <!--EVIP: StackPabel inside Button -->
    <StackPanel Orientation="Horizontal" Spacing="5">
        <Image Width="30" Height="30" Source="{x:Bind Image, Mode=OneWay}"/>
        <TextBlock Width="100" VerticalAlignment="Center"
                   Text="{x:Bind Title, Mode=OneWay}" Foreground="Black"/>
    </StackPanel>
</Button>
```

WritableBitmap

```
public static void SetPixels(
    this IEnumerable<(Point, Color)> values,
    WriteableBitmap image)
{
    // EVIP: using context of WritableBitmap.
    //  (Otherwise, it will be extremely slow.)
    // EVIP: using and IDisposable
    using (image.GetBitmapContext())
    {
        // EVIP: Tuple and deconstruction
        foreach ((Point p, Color c) in values)
            image.SetPixel((int)p.X, (int)p.Y, c);
    }
}
```

Mandelbrot\MandelbrotCommon\ImageIndexingExtensions.cs