Drag & Drop in Windows Phone

This article explains how to use Interaction Behaviors API to Drag & Drop objects in Windows Phone

25 Nov 2012

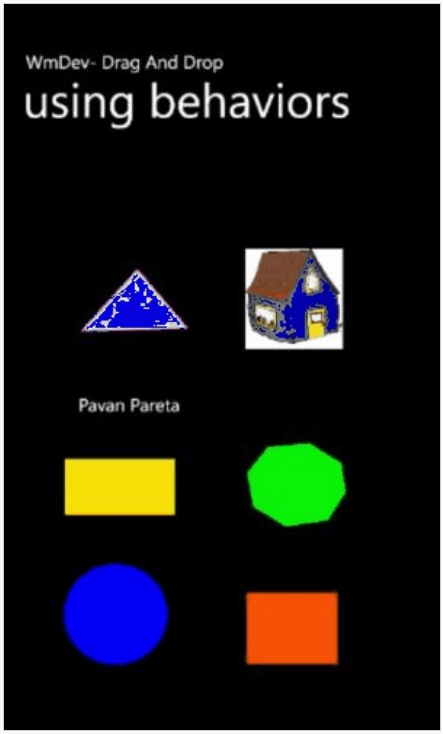
Introduction

There are three core sequences to be implemented in drag & drop using traditional process:

1. Tap (Press) & hold down – MouseDown Event.
2. Drag the mouse to place the object to the desired location – MouseMove Event.
3. Drop the object by releasing the mouse button - MouseUp Event.

In this example I have implemented the above drag'n'drop sequences using the Interaction Behaviors API. I have targeted both UI elements (drawing objects like Rectangle, Ellipse and Path etc) and UI components (TextBlock, Image). Note that you can also use other components.

The animated gif below shows the running example.



Output Screen

Implementation

Follow the steps mentioned to test the example, first create a new Windows Phone application. To do this start Microsoft Visual Studio then create a new Project and select Windows Phone Application Template. I have named the application "WmDev_DragAndDrop".

Step-1: Add two references to **Microsoft.Expression.Interactions** and **System.Windows.Interactivity** in your solution.

1. Go to solution explorer and right click on Project **->** References **->** click on Add References...
2. In the **Add Reference** dialog box, select the .NET tab indicating the type of component you want to reference.
3. Select the **Microsoft.Expression.Interactions** and **System.Windows.Interactivity** component to reference, and then click OK.

Step-2: Add the following namespace at the top of your XAML file:

```
xmlns:el="clr-namespace:Microsoft.Expression.Interactivity.Layout;assembly=Microsoft.Expression.Interactions"
xmlns:i="clr-namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity"
```

Step-3: Define the following objects in XAML.

1. Rectangle
2. Ellipse
3. Path
4. TextBlock
5. Image

(C) Copyright Nokia 2013. All rights reserved.

```
<Rectangle x:Name="rectangle1" Fill="#FFF35206" HorizontalAlignment="Left" Height="74" Margin="172,478,0,0" Stroke="Black" VerticalA
<Rectangle x:Name="rectangle2" Fill="#FFF3DF06" HorizontalAlignment="Right" Height="64" Margin="0,333,279,0" Stroke="Black" Vertical
<Ellipse Margin="54,450,288,72" Fill="Blue"/>
<Path Data="M78,318 L41,383 L111,380 z" Fill="#FFF306DF" HorizontalAlignment="Left" Height="71.5" Margin="70,0,0,405.5" Stretch="Fil
<Path Data="M275,309 L322,305 L357,327 L362,360 L342,391 L295,398 L259,376 L252,338 z" Fill="#FF06F30A" HorizontalAlignment="Right"
<TextBlock Height="30" Margin="70,50,250,133" Name="textBlock1" Text="Pavan Pareta" />
<Image Height="110" Name="image1" Stretch="Fill" Source="/WmDev_DragAndDrop;component/Images/house.png" Margin="254,90,95,407"/>
```

Step-4: Attaching a `MouseDragElementBehavior` to the object and setting its `ConstrainToParentBounds` property to `true/false`, so it will respect the bounds of its parent object like a `Grid` or `Border` etc. In this example I have set the property `ConstrainToParentBounds` to `false`; now it only allows user to drag and drop within a parent's restricted area.

```
<Rectangle x:Name="rectangle1" Fill="#FFF35206" HorizontalAlignment="Left" Height="74" Margin="172,478,0,0" Stroke="Black" VerticalA
    <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="False"/>
    </i:Interaction.Behaviors>
</Rectangle>
```

Here is the complete XAML code:

```
<Grid x:Name="gd" Grid.Row="1" Margin="12,0,12,0">

    <Rectangle x:Name="rectangle1" Fill="#FFF35206" HorizontalAlignment="Left" Height="80" Margin="254,481,0,0" Stroke="Black" Verti
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="False"/>
        </i:Interaction.Behaviors>
    </Rectangle>

    <Rectangle x:Name="rectangle2" Fill="#FFF3DF06" HorizontalAlignment="Right" Height="64" Margin="0,333,279,0" Stroke="Black" Vert
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
        </i:Interaction.Behaviors>
    </Rectangle>

    <Ellipse Margin="54,450,288,72" MouseMove="MouseMoving" Fill="Blue">
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
        </i:Interaction.Behaviors>
    </Ellipse>

    <Path Data="M78,318 L41,383 L111,380 z" Fill="#FFF306DF" HorizontalAlignment="Left" Height="71.5" Margin="70,0,0,438" Stretch="F
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
        </i:Interaction.Behaviors>
    </Path>

    <Path Data="M275,309 L322,305 L357,327 L362,360 L342,391 L295,398 L259,376 L252,338 z" Fill="#FF06F30A" HorizontalAlignment="Rig
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
        </i:Interaction.Behaviors>
    </Path>

    <TextBlock Height="30" Margin="70,50,250,133"
        Name="textBlock1" Text="Pavan Pareta" MouseMove="MouseMoving">
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
                </i:Interaction.Behaviors>
    </TextBlock>

    <Image Height="110" Name="image1" Stretch="Fill" Source="/WmDev_DragAndDrop;component/Images/house.png" Margin="254,90,95,407" M
        <i:Interaction.Behaviors>
            <el:MouseDragElementBehavior ConstrainToParentBounds="True"/>
        </i:Interaction.Behaviors>
    </Image>

</Grid>
```


(C) Copyright Nokia 2013. All rights reserved.

UI: Design view output

Step-5: In addition i've attached an event **(`MouseMoving`)** to all objects. Handled and set ZIndex value to keep dragging object always on front of all objects.

> **Note:** While dragging and placing an object over another; the dragged object get sets behind the other one. To resolve this issue, a property `ZIndex` may be to the object being dragged.
> `ZIndex` is an attached property, which allows you to explicitly set the z-order of objects in the child collection of `Grid`, `Canvas` and `StackPanel` objects.

Code Snippet for the `MouseMoving` method.

```
private void MouseMoving(object sender, MouseEventArgs e)
{
            if (sender.GetType() == typeof(Rectangle))
            {
                Rectangle cx = (Rectangle)sender;
                Canvas.SetZIndex(cx, idx++);
            }
            if (sender.GetType() == typeof(Ellipse))
            {
                Ellipse cx = (Ellipse)sender;
                Canvas.SetZIndex(cx, idx++);
            }
            if (sender.GetType() == typeof(Path))
            {
                Path cx = (Path)sender;
                Canvas.SetZIndex(cx, idx++);
            }
            else if (sender.GetType() == typeof(TextBlock))
            {
                TextBlock realSender = (TextBlock)sender;
                Canvas.SetZIndex(realSender, idx++);
            }
            else if (sender.GetType() == typeof(Image))
            {
                Image realSender = (Image)sender;
                Canvas.SetZIndex(realSender, idx++);
            }
}
```

Step-6: Now run the application by pressing **F5** Key.

References

1. MouseDragElementBehavior
2. MouseDragElementBehavior.ConstrainToParentBounds Property
3. ZIndex

Sample Code

You can download sample code from Media:WmDev DragAndDrop.zip.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.