Tabbed interface with Pivot animation for Windows Phone

This article explains how to create a tabbed interface with a pivot-style animation. This sort of UI has been used in the official Twitter app and 6tag Instagram client on Windows Phone.

22 Sep 2013

Introduction

The official Twitter app on Windows Phone uses an uncommon interface for Windows Phone - effectively a "tabbed" control but with pivot-like animation for displaying the selected content.

The behaviour of the Twitter app suggests this is a normal pivot control, but 2 things are different from the standard implementation. Firstly, all the icons stay in view, even while switching through each pivot content (normally the headers will flow outside the boundary of the normal page). Secondly, the use of such icons as headers is also not common.

This article explains how to create this sort of user interface using path controls for its headers. These path controls contain XAML data to represent the "tab icon" images. The screenshots below show how this looks (there is also a video below).

pivot 1

pivot 3

Implementation

Create a new windows phone app and in the **MainPage.xaml** add a `ListBox` called `ImageBar`. This `ImageBar` will host our 4 icons which will act as our pivot headers. Each icon is actually a custom `PathControl` (more on this below) that contains XAML data representing the image.

We are using a `ListBox` for this because it has all the necessary event handling and actions to select an item and automatically deselecting all other items. This will also help in setting a selected colour.

Note: To get nice looking icons, just download Metro Studio from Syncfusion! Open the program, search for a nice icon and copy paste the generated XAML data into your windows phone project.

```
<ListBox x:Name="ImageBar"
            Grid.Row="0"
            ScrollViewer.VerticalScrollBarVisibility="Disabled"
            ItemContainerStyle="{StaticResource ListBoxItemStyle}"
            SelectedIndex="{Binding SelectedIndex, ElementName=ContentPivot, Mode=TwoWay}">
    <ListBox.ItemsPanel>
        <ItemsPanelTemplate>
            <customcontrols:SplitPanel />
        </ItemsPanelTemplate>
    </ListBox.ItemsPanel>
    <ListBoxItem>
        <customcontrols:PathControl Content="M37.786324,45.467003C39.394405,45.467003 40.702843,46.774002 40.702843,48.383999 40.702
    </ListBoxItem>
    <ListBoxItem>
        <customcontrols:PathControl Content="M44.032538,19.436991L46.035559,20.664995C46.766642,21.111948 46.996969,22.06896 46.5463
    </ListBoxItem>
    <ListBoxItem>
        <customcontrols:PathControl Content="M19.589941,21.452004L29.499502,21.452004 37.799032,35.562995 29.729471,35.562995 37.109
    </ListBoxItem>
    <ListBoxItem>
```


(C) Copyright Nokia 2013. All rights reserved.

```
        <customcontrols:PathControl Content="M31.348,48.494C32.685356,48.494 33.770001,49.577489 33.770001,50.913598 33.770001,52.24
    </ListBoxItem>
</ListBox>
```

After that, also add a `Pivotl` control to the page, but don't add a `HeaderTemplate`! We don't need one, because our `ImageBar` list box will act as one. We need as many `PivotItems` as `ImageBar` images, because each image represents one `PivotItem`.

```
<controls:Pivot x:Name="ContentPivot"
                Grid.Row="1"
                Margin="0,-12,0,0">
    <controls:PivotItem>
        <StackPanel>
            <TextBlock Text="RAIN" Style="{StaticResource PhoneTextTitle2Style}" FontWeight="Bold"/>
            <RichTextBox Margin="0,12,0,0">
                <Paragraph>
                    <Run>
                        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc sit amet interdum dolor. In ornare vulputate n
                    </Run>
                </Paragraph>
            </RichTextBox>
        </StackPanel>
    </controls:PivotItem>
    <controls:PivotItem>
        <StackPanel>
            <TextBlock Text="CLOUDY" Style="{StaticResource PhoneTextTitle2Style}" FontWeight="Bold"/>
            <RichTextBox Margin="0,12,0,0">
                <Paragraph>
                    <Run>
                        Sed bibendum vehicula quam. Mauris id massa non erat vehicula feugiat id eget nibh. Donec quis neque in lore
                    </Run>
                </Paragraph>
            </RichTextBox>
        </StackPanel>
    </controls:PivotItem>
    <controls:PivotItem>
        <StackPanel>
            <TextBlock Text="STORM" Style="{StaticResource PhoneTextTitle2Style}" FontWeight="Bold"/>
            <RichTextBox Margin="0,12,0,0">
                <Paragraph>
                    <Run>
                        Integer eget molestie lacus. Nam facilisis purus sed vehicula rutrum. Integer auctor imperdiet lectus a ultr
                    </Run>
                </Paragraph>
            </RichTextBox>
        </StackPanel>
    </controls:PivotItem>
    <controls:PivotItem>
        <StackPanel>
            <TextBlock Text="DRIZZLE" Style="{StaticResource PhoneTextTitle2Style}" FontWeight="Bold"/>
            <RichTextBox Margin="0,12,0,0">
                <Paragraph>
                    <Run>
                        Phasellus molestie volutpat ante pulvinar convallis. Suspendisse tincidunt ante quis arcu dignissim, nec con
                    </Run>
                </Paragraph>
            </RichTextBox>
        </StackPanel>
    </controls:PivotItem>
</controls:Pivot>
```

Note: Important thing to note is the `SelectedIndex` property on the `ImageBar` control! It's bound to the `SelectedIndex` of the `Pivot` control in 2 way mode. By doing this, these 2 controls their `SelectedIndex` will stay in sync!

What we now need to do, is making the icons change colour when you select one from the `ImageBar`. This is done by giving the `ListBox` an `ItemContainerStyle` in which we are going to use the visual states of the `ListBox` to set the colour when an item is selected. In our example we use the `PhoneAccentColor` for the selected state and `PhoneInactiveColor` for the unselected state.

```
<Style x:Key="ListBoxItemStyle" TargetType="ListBoxItem">
    <Setter Property="Background" Value="Transparent"/>
    <Setter Property="BorderThickness" Value="0"/>
    <Setter Property="BorderBrush" Value="Transparent"/>
    <Setter Property="Padding" Value="0"/>
    <Setter Property="HorizontalContentAlignment" Value="Center"/>
    <Setter Property="VerticalContentAlignment" Value="Top"/>
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="ListBoxItem">
                <Border x:Name="LayoutRoot" BorderBrush="{TemplateBinding BorderBrush}" BorderThickness="{TemplateBinding BorderThic
                    <VisualStateManager.VisualStateGroups>
                        <VisualStateGroup x:Name="CommonStates">
                            <VisualState x:Name="Normal"/>
                            <VisualState x:Name="MouseOver"/>
                        </VisualStateGroup>
                        <VisualStateGroup x:Name="SelectionStates">
                            <VisualState x:Name="Unselected">
                                <Storyboard>
                                    <ColorAnimation Duration="0" To="{StaticResource PhoneInactiveColor}" Storyboard.TargetProperty=
                                    <ColorAnimation Duration="0" To="{StaticResource PhoneInactiveColor}" Storyboard.TargetProperty=
                                </Storyboard>
                            </VisualState>
                            <VisualState x:Name="Selected">
                                <Storyboard>
                                    <ColorAnimation Duration="0" To="{StaticResource PhoneAccentColor}" Storyboard.TargetProperty="(
                                </Storyboard>
                            </VisualState>
                            <VisualState x:Name="SelectedUnfocused">
                                <Storyboard>
                                    <ColorAnimation Duration="0" To="{StaticResource PhoneAccentColor}" Storyboard.TargetProperty="(
                                </Storyboard>
                            </VisualState>
                        </VisualStateGroup>
                    </VisualStateManager.VisualStateGroups>
                    <ContentControl x:Name="ContentContainer" ContentTemplate="{TemplateBinding ContentTemplate}" Content="{Template
                </Border>
            </ControlTemplate>
        </Setter.Value>
    </Setter>
```


(C) Copyright Nokia 2013. All rights reserved.

```
</Style>
```

When you try this in your own application, you will notice that a specific element is missing in the XAML mentioned above. This is the custom control called `PathControl`. It's actually a `ContentTemplate` wrapper around a Path control, but this needed to allow the use of the `Foreground` property defined in the `ListBoxItemStyle`. Because by default the `Path` control doesn't have a `Foreground` property, but uses the `Fill` property to set it's colour. With this custom `PathControl` we transfer the value of the `Foreground` property to the `Fill` property so that all styling is working. We do a similar transfer for the `Content` property, this needs to be moved to the `Data` property of the actual `Path` control.

```
public class PathControl : ContentControl
{
    public PathControl()
    {
        DefaultStyleKey = typeof(PathControl);
    }
}
```

```
<Style TargetType="controls:PathControl">
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="controls:PathControl">
                <Grid>
                    <Path Data="{TemplateBinding Content}"
                          Fill="{TemplateBinding Foreground}"/>
                </Grid>
            </ControlTemplate>
        </Setter.Value>
    </Setter>
</Style>
```

But still, when you would run this, each icon will stick to the next, we need to divide them evenly! You could calculate this and add margins... But in our example we again use a special user control. This one is derived from Panel, called `SplitPanel` and we use this as `ItemsPanelTemplate` on the `ListBox`. It will do the width calculation and add the needed space so that each listbox item takes up equally space along the page.

```
public class SplitPanel : Panel
{
    protected override Size MeasureOverride(Size availableSize)
    {
        // the final measure size is the available size for the width, and the maximum
        // desired size of our children for the height
        Size finalSize = new Size { Width = availableSize.Width };

        if (this.Children.Count != 0)
            availableSize.Width /= (double)this.Children.Count;

        foreach (var current in this.Children)
        {
            current.Measure(availableSize);

            Size desiredSize = current.DesiredSize;
            finalSize.Height = Math.Max(finalSize.Height, desiredSize.Height);
        }

        // make sure it will works in design time mode
        if (double.IsPositiveInfinity(finalSize.Height) || double.IsPositiveInfinity(finalSize.Width))
            return Size.Empty;

        return finalSize;
    }

    protected override Size ArrangeOverride(Size arrangeSize)
    {
        Rect finalRect = new Rect(new Point(), arrangeSize);
        double width = arrangeSize.Width / this.Children.Count;

        foreach (var child in this.Children)
        {
            finalRect.Height = Math.Max(arrangeSize.Height, child.DesiredSize.Height);
            finalRect.Width = width;

            child.Arrange(finalRect);

            // move each child by the width increment
            finalRect.X += width;
        }

        return arrangeSize;
    }
}
```

With all this in place you should have a nice looking twitter mimicking app!

Video demo

The media player is loading...

Downloads

A complete working solution can be downloaded here: File:PivotWithPathControl.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.