Custom Live Tile Layout with UserControl as image source

This article explains how to create a custom layout tile for you application by rendering a User Control to a bitmap.

07 Jul 2013

Introduction

Windows Phone 8 has some restrictions on what can be changed in a tile update. For example, on the primary tile you can only specify Title, Background Image, Count (can be only int data type), Icon.

Image Courtesy Microsoft

Other use cases are not supported - for example having more than one counter on your tile, having a different format of counter instead on int data type, or to have custom position placement of your tile icon, etc.

This article shows how you can render virtually any custom tile layout.

Solution: Using a custom/user control

The solution is to create a User Control and stream it to a bitmap image - then use this image as the background source for our tile (a User Control is a UI control made up of existing UI elements but with its own behaviour - you can use it just like any other default controls available)

So basically our tile update functionality will not use the update attributes like counter, content or so. Instead all the required logic will be in the custom control and then we will take a snapshot of the control and then apply it as our background image.

Step 1: Create a user control

We will create a simple user control with some custom layout and functionally it will show us the current day and time.

```
<UserControl x:Class="CustomTile.CustomTileControl"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    d:DesignHeight="336" Width="336">

    <Grid x:Name="LayoutRoot" Background="{StaticResource PhoneChromeBrush}">
        <TextBlock HorizontalAlignment="Left" Margin="-2,104,0,0" TextWrapping="Wrap"
           VerticalAlignment="Top" Height="77" Width="337" FontSize="50"
Text="CustomTile"
           TextAlignment="Center"/>
        <Canvas HorizontalAlignment="Left" Height="97" VerticalAlignment="Top"
Width="89"
           Background="#FFA00808"/>
        <Canvas HorizontalAlignment="Left" Height="97" VerticalAlignment="Top"
Width="89"
           Background="#FFF0E507" Margin="247,0,0,0"/>
        <Canvas HorizontalAlignment="Left" Height="89" VerticalAlignment="Top"
```


(C) Copyright Nokia 2014. All rights reserved.

```
Width="88"
           Background="#FF077CF0" Margin="247,245,0,0"/>
        <TextBlock x:Name="tbDate" HorizontalAlignment="Left" Margin="0,181,0,0"
           TextWrapping="Wrap" VerticalAlignment="Top" Width="335" Height="65"
           TextAlignment="Center"  FontSize="30"/>
        <Canvas HorizontalAlignment="Left" Height="89" VerticalAlignment="Top"
Width="88"
           Background="#FF44C115" Margin="1,247,0,0"/>
    </Grid>
</UserControl>
```

This is how the control will look:

Step 2: Convert the user control to image source

The below code will user the custom control and save it as a .jpg file in the Isolated Storage. This image file will be later used as the background image of our tile.

```
var customTile = new CustomTileControl();
customTile.Measure(new Size(336, 336));
customTile.Arrange(new Rect(0, 0, 336, 336));

var bmp = new WriteableBitmap(336, 336);
bmp.Render(customTile, null);
bmp.Invalidate();

const string filename = "/Shared/ShellContent/CustomTile.jpg";

using (var isf = IsolatedStorageFile.GetUserStoreForApplication())
{
   if (!isf.DirectoryExists("/CustomLiveTiles"))
  {
          isf.CreateDirectory("/CustomLiveTiles");
   }
   using (var stream = isf.OpenFile(filename, System.IO.FileMode.OpenOrCreate))
   {
           bmp.SaveJpeg(stream, 336, 366, 0, 100);
   }
}
```

Step3: Create/Update Tile with the recently saved image

This is the final step where we will create a FlipTile. And then assign the `backgroundimage` attribute of this tile with the source of

(C) Copyright Nokia 2014. All rights reserved.

user control's image. Make sure you have a unique name of tile to have a unique URI. In the example we have kept the `Title` blank so that it doesn't appear in our final tile output.

```
FlipTileData tileData = new FlipTileData
{
     //Title = "CustomSecondaryTile",
     WideBackgroundImage = new Uri("isostore:" + filename, UriKind.Absolute),
     BackgroundImage = new Uri("isostore:" + filename, UriKind.Absolute),
};

string tileUri = string.Concat("/MainPage.xaml?", "");
ShellTile.Create(new Uri(tileUri, UriKind.Relative), tileData,true);
```

Code Attachment

Working example: File:CustomTile.zip

References

Live Apps: Creating Custom Tile and Lock Screen Images (blogs.msdn.com)

(C) Copyright Nokia 2014. All rights reserved.