Custom Live Tile with Formatted Text

This article explains how to format text on a Windows Phone *Live Tile* so that longer strings are not truncated.

Introduction

Live Tiles are the entry point to a Windows Phone app, and can provide dynamic information to the user though text and images. By default apps get a Flip Tile which has a front (usually displaying an icon) and back (usually displaying text) and periodically flips between them. Tiles come in three sizes - the user determines their preferred size at runtime.

One common problem is that text that will fit nicely on the back of the wide tile size will often be truncated when displayed on the medium size. Of course it is possible to have shorter text for the smaller tile-size, but sometimes we *really* need to display all the text, and can do so acceptably at a smaller font.

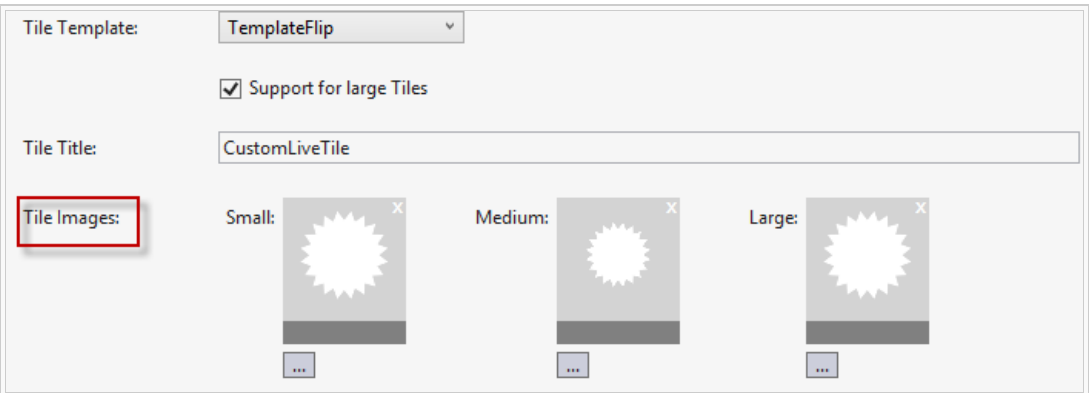
Custom text truncated

Custom text formatted to fit tile

This article shows how you can format the text on the back of a secondary tile by saving and displaying it as an image.

Creating a Live Tile

Apps automatically get a Flip Tile by default. Set the images for the front of the tiles (in the three different tile sizes) in the **WMAppManifest.xml** file from Assets/Tiles folder:



Provide Tile images in the WMAppManifest.xml

The three tile sizes with their default "front" icons are shown below. The user can change the size on the device start screen to small, medium or large:

Live Tiles Sizes

The text on the back of the tile is changed programmatically using code or by manually editing the **WMAppManifest.xml** file. To edit the application manifest file, Right-click **WMAppManifest.xml** and

- select **View Code** menu item or
- use **Open With…** menu item and select **XML (Text) Editor**

You can find changed data in next xml as part of the manifest file for `BackContent`, `BackTitle` and `LargeBackContent` parameters:

(C) Copyright Nokia 2014. All rights reserved.

```
<Title>CustomLiveTile</Title>
<BackContent>My Back Content</BackContent>
<BackBackgroundImageURI>
</BackBackgroundImageURI>
<BackTitle>My Back Title</BackTitle>
<LargeBackgroundImageURI IsRelative="true"
IsResource="false">Assets\Tiles\FlipCycleTileLarge.png</LargeBackgroundImageURI>
<LargeBackContent>My Large Back Content</LargeBackContent>
```

The final result of the text that we put for medium and large tiles are:

Back Content Default

What we can do from our application is to update data for default, Primary Tile and change every parameters in it or to create Secondary Tile (Secondary Tile can be created from application by pinning some functionality that are unique, for instance weather for specific city, contact for specific person or something similar).

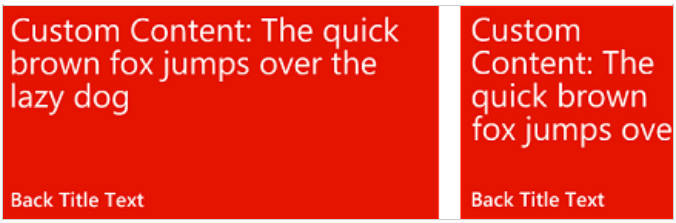
Update Live Tile with default parameters

If we want to change some data for Flip Live Tile we can do it from code. Firstly, we need to find Shell Tile and if it exists change `Title`, `BackTitle`, `BackContent` and `WideBackContent` parameters. If we don't want to use `BackBackgroundImage` we need to specify an empty URI.

```
ShellTile tile = ShellTile.ActiveTiles.FirstOrDefault();
if (tile != null)
{
   FlipTileData flipTile = new FlipTileData();

   flipTile.Title = "Title Text";
   flipTile.BackTitle = "Back Title Text";

   //Medium size Tile 336x336 px
   flipTile.BackContent = "Custom Content: The quick brown fox jumps over the lazy dog";
   flipTile.BackgroundImage = new Uri("/Assets/Tiles/FlipCycleTileMedium.png",
UriKind.Relative);
   flipTile.BackBackgroundImage = new Uri("", UriKind.Relative);

   //Wide size Tile 691x336 px
   flipTile.WideBackgroundImage = new Uri("/Assets/Tiles/FlipCycleTileLarge.png",
UriKind.Relative);
   flipTile.WideBackContent = "Custom Content: The quick brown fox jumps over the lazy
dog";
   flipTile.WideBackBackgroundImage = new Uri("", UriKind.Relative);

   //Update Live Tile
   tile.Update(flipTile);
}
```

This results in the tile below. As you can see, the text in the medium Tile is truncated. We can't change font size by default, so we will use image as Back content.

(C) Copyright Nokia 2014. All rights reserved.



Back Content Default using Update

Update Live Tile with rendered image

We can instead use an image of the text sized to fit the area (not that care needs to be taken to ensure that the text is still readable). The final result of the code in this section is shown below.

Back Content Custom using Update

We will use two methods that are in `Helpers` class (if we want to use it from more then one location in our app it is appropriate to be in some sort of `Util` class):

- `UpdateLiveTile`
- `RenderText`

Firstly, we need to find Shell Tile, and if it exists change `Title`, `BackTitle`, `BackContent` and `WideBackContent` parameters. The catch is to use `BackBackgroundImage` and `WideBackBackgroundImage` from Isolated Storage that contain our custom back content text. For medium Live Tile size, if text is greater than 135 characters, we use only first 135 characters and an ellipse in order to ensure the font is still readable.

```
public static void UpdateLiveTile(string info)
{
 ShellTile tile = ShellTile.ActiveTiles.FirstOrDefault();
 if (tile != null)
 {
  FlipTileData flipTile = new FlipTileData();
  flipTile.Title = "Title Text";
  flipTile.BackTitle = "Back Title Text";

  flipTile.BackContent = " ";
  flipTile.WideBackContent = " ";

  //Medium size Tile 336x336 px
  //Crete image for BackBackgroundImage in IsoStore
  if (info.Length >= 135)
  {
   RenderText(info.Substring(0, 135) + "...", 336, 336, 30, "BackBackgroundImage");
  }
  else
  {
   RenderText(info, 336, 336, 28, "BackBackgroundImage");
  }

  flipTile.BackBackgroundImage = new
```

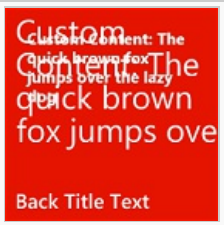

(C) Copyright Nokia 2014. All rights reserved.

```
Uri(@"isostore:/Shared/ShellContent/BackBackgroundImage.jpg", UriKind.Absolute);
//Generated image for Back Background 336x336
  flipTile.BackgroundImage = new Uri("/Assets/Tiles/FlipCycleTileMedium.png",
UriKind.Relative); //Default image for Background Image Medium Tile 336x336 px
  //End Medium size Tile 336x336 px

  //Wide size Tile 691x336 px
  flipTile.WideBackgroundImage = new Uri("/Assets/Tiles/FlipCycleTileLarge.png",
UriKind.Relative); ////Default image for Background Image Wide Tile 691x336 px

  //Create image for WideBackBackgroundImage in IsoStore
  RenderText(info, 691, 336, 40, "WideBackBackgroundImage");
  flipTile.WideBackBackgroundImage = new
Uri(@"isostore:/Shared/ShellContent/WideBackBackgroundImage.jpg", UriKind.Absolute);
  //End Wide size Tile 691x336 px

  //Update Live Tile
  tile.Update(flipTile);
 }
}
```

We need to set the image as and empty string first and then use image from Isolated Storage.

```
flipTile.BackContent = " ";
flipTile.WideBackContent = " ";
```

Warning: If we don't set `BackContent` and `WideBackContent` with and empty string we'll get default back content written over our new content as shown below:



Reset Back Content

With `RenderText` method we generate image that contains our custom text with custom parameters such as font size, font weight, text alignment etc.

```
private static void RenderText(string text, int width, int height, int fontsize, string
imagename)
{
 WriteableBitmap b = new WriteableBitmap(width, height);

 var canvas = new Grid();
 canvas.Width = b.PixelWidth;
 canvas.Height = b.PixelHeight;

 var background = new Canvas();
 background.Height = b.PixelHeight;
 background.Width = b.PixelWidth;

 //Created background color as Accent color
 SolidColorBrush backColor = new
```


(C) Copyright Nokia 2014. All rights reserved.

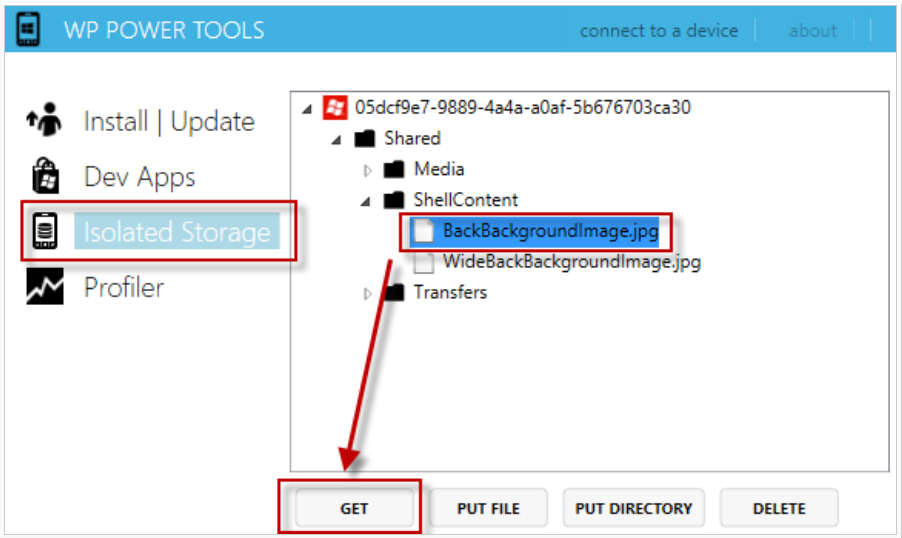
```
SolidColorBrush((Color)Application.Current.Resources["PhoneAccentColor"]);
 background.Background = backColor;

 var textBlock = new TextBlock();
 textBlock.Text = text;
 textBlock.FontWeight = FontWeights.Bold;
 textBlock.TextAlignment = TextAlignment.Left;
 textBlock.HorizontalAlignment = HorizontalAlignment.Center;
 textBlock.VerticalAlignment = VerticalAlignment.Stretch;
 textBlock.Margin = new Thickness(35);
 textBlock.Width = b.PixelWidth - textBlock.Margin.Left * 2;
 textBlock.TextWrapping = TextWrapping.Wrap;
 textBlock.Foreground = new SolidColorBrush(Colors.White); //color of the text on the
Tile
 textBlock.FontSize = fontsize;

 canvas.Children.Add(textBlock);

 b.Render(background, null);
 b.Render(canvas, null);
 b.Invalidate(); //Draw bitmap

 //Save bitmap as jpeg file in Isolated Storage
 using (IsolatedStorageFile isf = IsolatedStorageFile.GetUserStoreForApplication())
 {
  using (IsolatedStorageFileStream imageStream = new
IsolatedStorageFileStream("/Shared/ShellContent/" + imagename + ".jpg",
System.IO.FileMode.Create, isf))
  {
   b.SaveJpeg(imageStream, b.PixelWidth, b.PixelHeight, 0, 100);
  }
 }
}
```

You can play with formatting text that can appear on the Tile using size, foreground color, alignment, wrapping, weight when you create `TextBlock` control in `RenderText` method. But you need to be careful with colours because user can change current accent colour. We suggest you white text for back content.

How to get image from Isolated Storage

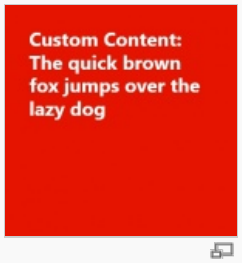
If you want to get image that is generated in Isolated Storage you can install Windows Phone Power Tools for WP 8 from CodePlex (http://wptools.codeplex.com/). First, we need to connect to Emulator (if we install app on the Emulator) and then Get right file:

(C) Copyright Nokia 2014. All rights reserved.



WP Power Tools

So, the Back Content is image with accent color as background with white text as Back Content:



BackBackgroundImage

Summary

This article showed how to provide custom title text to a Flip Tile, and more specifically, how to render title text to an image so that it can be displayed (in a smaller font) without being truncated.

References

- Creating Custom Live Tiles for Windows Phone (Spaso Lazarevic Blog - Original source)

(C) Copyright Nokia 2014. All rights reserved.