Windows Phone

Live Tiles Cheat Sheet

SDK v7.1 "Mango"

Beta v0.2

Prepared by Kevin Ashley, Microsoft Architect Evangelist

http://kevinashley.com

@kashleytwit

1 Application Tile

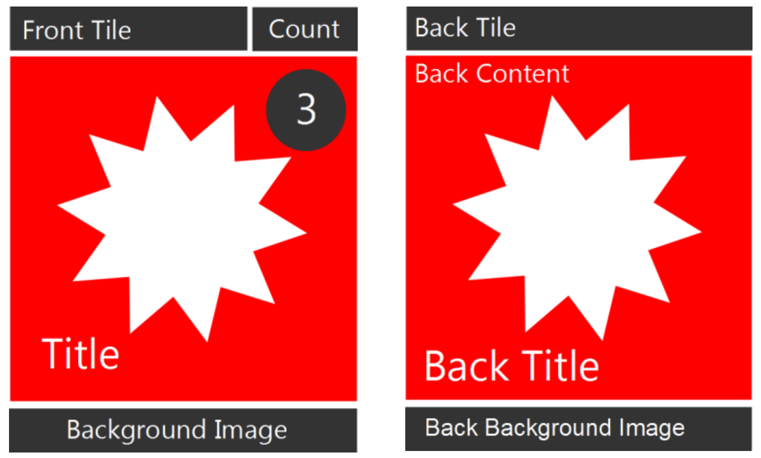
Can be created by the user only when the user taps and holds the application name in the Application List and then selects pin to start. Properties are initially set in the Application Manifest, **WPAppManifest.xml**:

```
<PrimaryToken TokenID="WPAppToken" TaskName="_default">
<TemplateType5>
  <BackgroundImageURI IsRelative="true"
   IsResource="false">AppTile.png</BackgroundImageURI>
  <Count>0</Count>
  <Title>WP7SampleApp52</Title>
  </TemplateType5>
</PrimaryToken>
```

Important:

Application Tile CANNOT be deleted programmatically.
Can be updated by:
ShellTile APIs. The Application Tile is always the first item in the ActiveTiles collection, even if the Tile is not pinned to Start.
Push Notifications for Windows Phone
ShellTileSchedule APIs

2 Secondary Tile



Can be created only as the result of user input in an application. The application then uses the **Create(Uri, ShellTileData)** method to create a Tile on Start. Because the UI will navigate to Start when a new secondary Tile is created, only one secondary Tile can be created at a time.

Can be deleted by:
The user unpinning the Tile from Start.
Uninstalling the application.
Calling Delete.
Can be updated by:
ShellTile APIs,
Push Notifications for Windows Phone,
ShellTileSchedule APIs.

3 Tile Properties

Title - A string indicating the title of the application. The Title must fit on a single line of text and should not be wider than the actual Tile. Approximately 15 characters will fit in the title before being truncated.

BackgroundImage. An image displayed on the front of the Tile. We recommend that you always have a background image on the front of the Tile.

Count (also known as Badge). An integer value from 1 to 99. If the value of Count is not set or it is set to 0, the circle image and value will not display in the Tile.

Tile images can be either a .jpg or .png file.
Use local resources for Tile images.
Use a .png with transparent portions to the image
Tiles are 173 x 173 pixels. Different size will stretch the image
https is not supported for remote images.
Remote images are limited to 80 KB or less
Remote images must download in 30 seconds or less
If the BackGround or BackBackGround images fail to load for any reason, none of the other properties set in the update will be changed either.

Important design considerations: tiles should conform to the look and feel of the Start screen. Avoid using round corners and chrome, and make tiles consistent with Metro UI clean design concepts.

4 Code Example – Creating Tiles

In this example we'll create a tile, and provide some code to handle user navigation, when the user clicks on the tile in the Start screen.

4.1 Create Secondary Tile

Check if the tile already exists, create a tile, making it a deep link with the url containing some id that will be later used to handle navigation events, when the user clicks on the tile.

```
ShellTile tile = ShellTile.ActiveTiles.FirstOrDefault(x =>
x.NavigationUri.ToString().Contains("someId=" + someId));

if (tile == null)
{
  StandardTileData NewTileData = new StandardTileData
  {
        BackgroundImage = new Uri(item.PostImage,
                      UriKind.Relative),
        Title = item.Title,
        Count = 0,
        BackContent = item.Title,
        BackBackgroundImage = new Uri(@"/TileImage.png",
                    UriKind.Relative)
   };

   ShellTile.Create(
         new Uri("/Details.xaml?someId="+someId,
                 UriKind.Relative),
                 NewTileData
         );
}
```

4.2 Handle Navigation to the App

Once the tile is created, we need to provide some code to handle the navigation back to the app, when the user clicks on the tile.

This can be done in **OnNavigatedTo** method. Note that in the code below, we're parsing someId that was used to create the tile Uri in the code above.

```
protected override void OnNavigatedTo(NavigationEventArgs e)
{

if (NavigationContext.QueryString.ContainsKey("someId"))
{
  _someId = NavigationContext.QueryString["someId"];
…
```

5 Tile Notifications

There're three types of push notifications: **Toast**, **Tile** and **Raw**. Let's take a look at the **Tile** notification mechanism. Properties that can be updated are:

Title. A string indicating the title of the application. The Title must fit on a single line of text and should not be wider than the actual Tile. Approximately 15 characters will fit in the title before being truncated.
BackgroundImage. An image displayed on the front of the Tile. We recommend that you always have a background image on the front of the Tile.
Count (also known as Badge). An integer value from 1 to 99. If the value of Count is not set or it is set to 0, the circle image and value will not display in the Tile.
BackTitle. A string displayed at the bottom of the back of a Tile. The BackTitle must fit on a single line of text and should not be wider than the actual Tile. Approximately 15 characters will fit in the title before being truncated.
BackBackgroundImage. An image displayed on the back of the Tile.
BackContent. A string displayed in the body of the back of a Tile. Approximately 40 characters will fit in the Tile before being truncated.

Note: When the app uses Push Notifications, please, check application policies on MSDN. For notifications other than tile updates, you may be required to provide opt out mechanisms in the app.

6 Code Example – Tile Notifications

To update tiles using a push notification service, you need to create a push notification channel first, then a handler to handle the notification. You will also need a server side script that sends notification messages in XML format.

6.1 Create Notification Channel

This code example creates a push notification channel.

```
using Microsoft.Phone.Notification;

HttpNotificationChannel pushChannel;

    // The name of our push channel.
    string channelName = "TileSampleChannel";
```

```
        InitializeComponent();

        // Try to find the push channel.
        pushChannel = HttpNotificationChannel.Find(channelName);

        // If the channel was not found, then create a new connect
ion to the push service.
        if (pushChannel == null)
    {
        pushChannel = new HttpNotificationChannel(channelName);

        // Register for all the events before attempting to open t
he channel.
        pushChannel.ChannelUriUpdated += new EventHandler<Notificat
ionChannelUriEventArgs>(PushChannel_ChannelUriUpdated);
        pushChannel.ErrorOccurred += new EventHandler<NotificationC
hannelErrorEventArgs>(PushChannel_ErrorOccurred);

        pushChannel.Open();

        // Bind this new channel for Tile events.
        pushChannel.BindToShellTile();


    }else{
        // The channel was already open, so just register for al
l the events.
        pushChannel.ChannelUriUpdated += new EventHandler<Notifi
cationChannelUriEventArgs>(PushChannel_ChannelUriUpdated);
        pushChannel.ErrorOccurred += new EventHandler<Notificati
onChannelErrorEventArgs>(PushChannel_ErrorOccurred);

        // Display the URI for testing purposes. Normally, the U
RI would be passed back to your web service at this point.
        System.Diagnostics.Debug.WriteLine(pushChannel.ChannelUr
i.ToString());
        MessageBox.Show(String.Format("Channel Uri is {0}",
        pushChannel.ChannelUri.ToString()));

    }
```

6.2 Handle Push Notifications

Next, we need a method to handle notification events:

```
void PushChannel_ChannelUriUpdated(object sender,
 NotificationChannelUriEventArgs e)
{

   Dispatcher.BeginInvoke(() =>
   {
     // Display the new URI for testing purposes. Normally, the U
RI would be passed back to your web service at this point.
```

```
     System.Diagnostics.Debug.WriteLine(e.ChannelUri.ToString());
     MessageBox.Show(String.Format("Channel Uri is {0}",
     e.ChannelUri.ToString()));

   });
}
```

6.3 Notification Message

The notification XML message looks similar to the following; you can use any server side script to generate it. You can provide any updateable properties in the notification message.

```
<?xml version="1.0" encoding="utf-8"?>
<wp:Notification xmlns:wp="WPNotification">
 <wp:Tile>
<wp:BackgroundImage>/Images/Sunny.png</wp:BackgroundImage>
   <wp:Count>68</wp:Count>
   <wp:Title>Mountain View</wp:Title>
 </wp:Tile>
</wp:Notification>
```

7 Reference and Resources

Windows Phone Tile resources on MSDN:

http://msdn.microsoft.com/en-us/library/hh202948(v=VS.92).aspx

Push Notifications for Windows Phone Tiles on MSDN:

http://msdn.microsoft.com/en-us/library/hh202970(v=VS.92).aspx

Windows Phone Training

http://create.msdn.com/en-US/education/catalog

Windows Phone Offline Training Kit

http://msdn.microsoft.com/en-us/WP7MangoTrainingCourse

Register for App Hub

http://create.msdn.com/en-us/home/membership

My Blog

http://kevinashley.com