Camera Lens apps on Windows Phone

This index provides links to Camera lens resources for Windows Phone.

Introduction

A lens is a specialised camera or imaging app that can be opened from the built-in camera app (in addition to the application list). Lenses offer a consistent experience for users to find apps and tools which can work with the camera and images.

Lenses are well documented on Dev Center in the following articles and examples:

- Lens extensibility for Windows Phone 8 (MSDN)
- Lenses for Windows Phone 8 (MSDN)
- Lens Design Guidelines (MSDN)
- Basic Lens Sample (MSDN)

The wiki also has a number of practical examples addressing specific use-cases:

- Handwriting overlay lens app for Windows Phone
- Creating a Lens application that uses HLSL effects for filters

The main value in this article is the index of lens resources above. The remainder of the article provides a brief overview of the "minimal lens app".

Minimal lens requirements

Lens picker icons

Lens apps require extra icons for the lens picker.

You must add three different icons in your Assets folder: **Lens.Screen-WVGA.png** should be a 173x173 pixel image, **Lens.Screen-720p.png** should be a 259x259 pixel image, and **Lens.Screen-WXGA.png** should be a 277x277 pixel image.

Lens registration

Lens apps must be registered. This is done by editing the **WMAppManifest.xml** file located in the Properties folder of your app.

Add the following lines to the XML file after the closing `</token>` tag (via MSDN):

```
<Extensions>
 <Extension ExtensionName="Camera_Capture_App"
  ConsumerID="{5B04B775-356B-4AA0-AAF8-6491FFEA5631}"
  TaskID="_default" />
</Extensions>
```

Capabilities

The lens "framework" does not itself require any capabilities. However to work with the camera and images in the gallery, most lenses will need capabilities `ID_CAP_ISV_CAMERA` and `ID_CAP_MEDIALIB_PHOTO` (respectively).

Handling a Lens Request

When an app is launched as lens, it will launch the default page unless specified otherwise. If a separate page is required for the lens viewfinder, create a new page which does contain a viewfinder and forward the app to that page if the app is opened from the Camera app.

Open the **App.xaml.cs** file, then add the following class to it (via MSDN):

```
class LensExampleUriMapper : UriMapperBase
{
```

```
    private string tempUri;
    public override Uri MapUri(Uri uri)
    {
        tempUri = uri.ToString();
        // Look for a URI from the lens picker.
        if (tempUri.Contains("ViewfinderLaunch"))
        {
            // Launch as a lens, launch viewfinder screen.
            return new Uri("/viewfinderExperience.xaml", UriKind.Relative);
        }
        // Otherwise perform normal launch.
        return uri;
    }
}
```

Finally, add the following to your `InitializePhoneApplication` function (you might need to show the "Phone application initialization" region) in **App.xaml.cs**:

```
//Assign the lens example URI-mapper class to the application frame.
RootFrame.UriMapper = new LensExampleUriMapper();
```

The above example will send the app to the **viewfinderExperience.xaml** page if it's being launched as a lens, and to the app's default start page otherwise.

Developing the Viewfinder Page

Getting the MediaViewer

The easiest way to go about getting the viewfinder page set up would be to download Microsoft's `MediaView` control and use it in your app. Download the Basic Lens Sample on MSDN, and then import the "MediaView" project into your own application. Build the solution (Build -> Build solution), and add a reference in your own project (right-click references in the Solution Explorer and click Add) to `MediaView`, then go back to (or create) your **viewfinderExperience.xaml** page and add the following line to the `<phone:PhoneApplicationPage>` element:

```
xmlns:controls="clr-namespace:Microsoft.Phone.Controls;assembly=MediaViewer"
```

Frontend/XAML

At the very least, add a

```
<controls:MediaViewer>
```

element with

```
Items="{Binding CameraRoll}"
```

in the root grid. A complete `MediaViewer` example is available as part of the Basic Lens Sample download on MSDN.

Backend/C#

You should download the code from the sdkBasicLensWP8CS project (again, part of the Basic Lens Sample download on MSDN), and then copy the ViewModels folder into your project (changing sdkBasicLensWP8CS to the name of your project). You can then modify the code in the **MainPage.xaml.cs** file found in the sdkBasicLensWP8CS project of the Basic Lens Sample

download on MSDN to fit your needs.

How to test a Lens

The Windows Phone 8 Emulator has a functioning camera which will display a multi-coloured block moving around the screen. While this is satisfactory to ensure that you've set up the lens correctly, it's highly recommended that you test your app on an actual Windows Phone 8 device before completion.

Credits

Most of the code samples on this page were taken from the Basic Lens Sample on MSDN and Lens Extensibility on MSDN.