# Camera and Image Configuration

## Plugins

Install the plugins from NUGET into the Android and Shared projects only: Xam.Plugin.Media

Android Manifest:

Under properties folder in solution explorer for the Android project, open Android Manifest xml file and add the following between the <application> tags, you will need to change the :

<provider android:name="android.support.v4.content.FileProvider"

android:authorities="${applicationId}.fileprovider"

android:exported="false"

android:grantUriPermissions="true">

<meta-data android:name="android.support.FILE\_PROVIDER\_PATHS"

android:resource="@xml/file\_paths"></meta-data>

</provider>

Add a new folder called xml to the Resources folder of the Android project then add a new xml file called file\_paths.xml. Inside the file add:

<?xml version="1.0" encoding="utf-8"?>

<paths xmlns:android="http://schemas.android.com/apk/res/android">

<external-files-path name="my\_images" path="Pictures" />

<external-files-path name="my\_movies" path="Movies" />

</paths>

## Assembly Info

Open the Assembly Info C# file under properties in the solution explorer for Android project and add:

[assembly: UsesFeature("android.hardware.camera", Required = false)]

[assembly: UsesFeature("android.hardware.camera.autofocus", Required = false)]

# Calling the Camera App

var photo = await Plugin.Media.CrossMedia.Current.TakePhotoAsync(new Plugin.Media.Abstractions.StoreCameraMediaOptions() { Name = filename });

If the photo object is not null, you will have access to the Path property and also you can use ImageSource.FromStream to set the source of an Image view:

if (photo != null)

{

lblPath.Text = photo.Path;

photoImage.Source = ImageSource.FromStream(() => { return photo.GetStream(); });

}

};

You can also use a Path string to bind to an image source property inside of a view cell:

img.SetBinding(Image.SourceProperty,"Path");