Gallery Screenshot (iOS / Android)

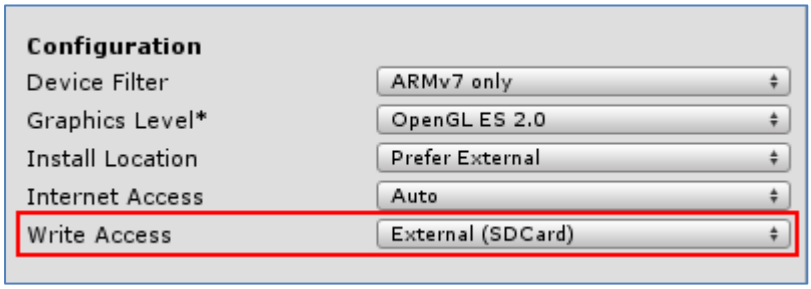
This plugin will take a screenshot of your app, and register the image file so that it appears in your picture gallery (camera roll) on both iOS and Android. It uses native extensions so you will need Unity iOS or Unity Android (which both come as standard in version 4.0 onwards). Tested and working with Unity 3.5 & 4.2.

The plugin can also take an existing image and register it to the picture gallery.

Android Setup

The asset package should import to "***Assets/Plugins/***" - please ensure this is the location of the files otherwise you will run into errors.

If you are using Android, you will need to go into ***Build Settings*** and access the ***Player Settings*** panel. Change ***Write Access*** to ***External (SDCard)*** to allow the screenshots to save.



Using the Plugin to Save a Screenshot

This plugin is initialised through a Coroutine. To take a screenshot write the following code:

StartCoroutine(ScreenshotManager.Save("screenshot_name", "album_name"));

Where ***screenshot_name*** is the filename you would like the screenshot to be saved as, and ***album_name*** is the folder that the screenshot will be put into (and will appear as an album name on Android). A number and timestamp will be automatically appended to each image to prevent overwrites, i.e. MyScreenshot_1_12-03-13. This can easily be modified in the ScreenshotManager.cs source.

```
void OnGUI ()
{
    if(GUI.Button (new Rect (10,10,150,100), "Take Screenshot"))
    {
        StartCoroutine(ScreenshotManager.Save("MyScreenshot", "MyApp"));
    }
}
```

N.B. On older devices (iPhone 3 / Galaxy S) it can take a couple of seconds to write the picture file to the SD card. If the user exits before this finishes, the photo won't be registered to the gallery. Therefore it is recommended to have a "saving photo" graphic on screen for these older devices.

Using the Plugin to Register an Existing Image

Once again this is initialised through a Coroutine. To register the image to your gallery write:

StartCoroutine(ScreenshotManager.SaveExisting("filepath"));

Where ***filepath*** is the location of your saved file (as a string variable). Remember you must ensure your source texture is set to **Read/Write Enabled** in the Unity editor for this to work. There is a good example on my blog detailing how to save an image to disk, and then register it to appear in the gallery.

http://ryanwebb.com/blog/gallery-screenshot-and-saving-existing-images

```
IEnumerator SaveTexture ()
{
    yield return new WaitForEndOfFrame();

    byte[] bytes = texture.EncodeToPNG();
    string path = Application.persistentDataPath + "/" + fileName + ".png";
    File.WriteAllBytes(path, bytes);

    yield return new WaitForEndOfFrame();

    StartCoroutine(ScreenshotManager.SaveExisting(path));
}
```

If you run into problems, you can email me at ryan@ryanwebb.com, but please make sure you have thoroughly read this manual beforehand.

Thanks for purchasing the plugin!