Aperture Super fast screen capture

Features

Write to the Camera Roll, disk, or a texture

Automatically save your screenshots to the users Camera Roll, or save them to disk for use with other tools. You can also write directly back into a Texture2D, to use the capture either in your game world or in your in-game UI.

Easily customizable screenshot preview

Includes a native iOS view with a preview of the screen capture, with callbacks for whether the user cancels or accepts the preview. The preview can easily be styled to match your existing game UI.

Speed!

Much faster than the built in screen capture functions.

Supports anti-aliasing

Don't sacrifice your graphics just to take screenshots. Aperture captures screenshots even with anti-aliasing enabled!

Compatable with Vuforia

Designed with augmented reality in mind, Aperture is compatible with Vuforia and many other AR libraries out of the box.

Requirements

- Unity 4.2 or higher
- iOS 6.0 or higher

Support

For support or to report any issues, please contact us at support@scarlet.io or @scarlet_io on Twitter.

If you're using Aperture in one of your projects, we'd love to hear about it!

Installation

Install the Unity package and drop the Aperture prefab into your scene, then you're ready to go.

To take a screenshot, all you need to do is send a `Photo()` message to the prefab!

Configuration

Save Mode

Aperture can save photos in several ways, which can be toggled on the prefab.

PhotoLibrary

Saves the photo directly to the users camera roll. Does not display a preview.

Disk

Saves the photo to disk in the applications Documents directory. Does not display a preview. Fires the `OnSavedToDisk` event.

DiskPreview

Saves the photo to disk, and displays it in the preview view. Fires the `OnSavedToDisk` event.

DiskPhotoLibraryPreview

Saves the photo to the camera roll and to disk, as well as displaying it in the preview. Fires the `OnSavedToDisk` event.

Texture

Saves the photo into a texture. The texture must already exist. Use the `targetTexture` property to set which texture the photo will save into. In this mode `Photo()` will do nothing if `targetTexture` is not set.

Preview Animation

This setting toggles how the capture preview should be animated in.

None

No animation, presents immediately.

CoverVertical

The standard iOS presentation method, the preview slides in from the bottom.

Crossfade

Fade in the preview. The fade animation works well as a camera flash effect, to alert the user to a screen capture.

Methods

Photo()

Starts the screen capture process.

SetSaveMode(SaveMode newMode)

Allows for changing the save method at runtime.

SetPreviewAnimation(PreviewAnimation style)

Allows for changing the preview animation method at runtime.

Events

If you connect a GameObject into the Event Receiver field on the Aperture prefab, it will these messages will be sent to it.

Do **not** put your event receiver script on the prefab.

OnPhoto()

Photo capture has begun. Use this to hide any UI you don't want visible in the photo.

OnCaptureComplete()

Photo capture has completed.

OnSavedToDisk(string path)

Photo capture has finished writing the image to disk, with the path to the file.

`OnAcceptedPreview`

User selected the positive action button on the photo preview.

`OnCancelledPreview`

User selected the negative action button on the photo preview.

Delegates

For slightly better performance, or if multiple listeners are necessary, the above events are also exposed as delegates through `Aperture.Events`.

Notes

- There may be a slight delay the first time `Photo()` is called, as the plugin needs to initialize.
- There is a one frame delay before a photo is captured.
- Photo capture is synchronous but saving is not. the preview will often be delayed a few frames. This is important if you have pausing Unity turned on, as you may see a few more frames of gameplay before the preview appears.
- Photos are **not** marked as Do-Not-Backup automatically, when writing to disk.
- Images written into a texture will appear vertically flipped. This can be corrected in the material UV settings, or by flipping the mesh.
- Output textures should be set with the `BGRA32` texture format. Other formats will work on iPhone but not iPad.