

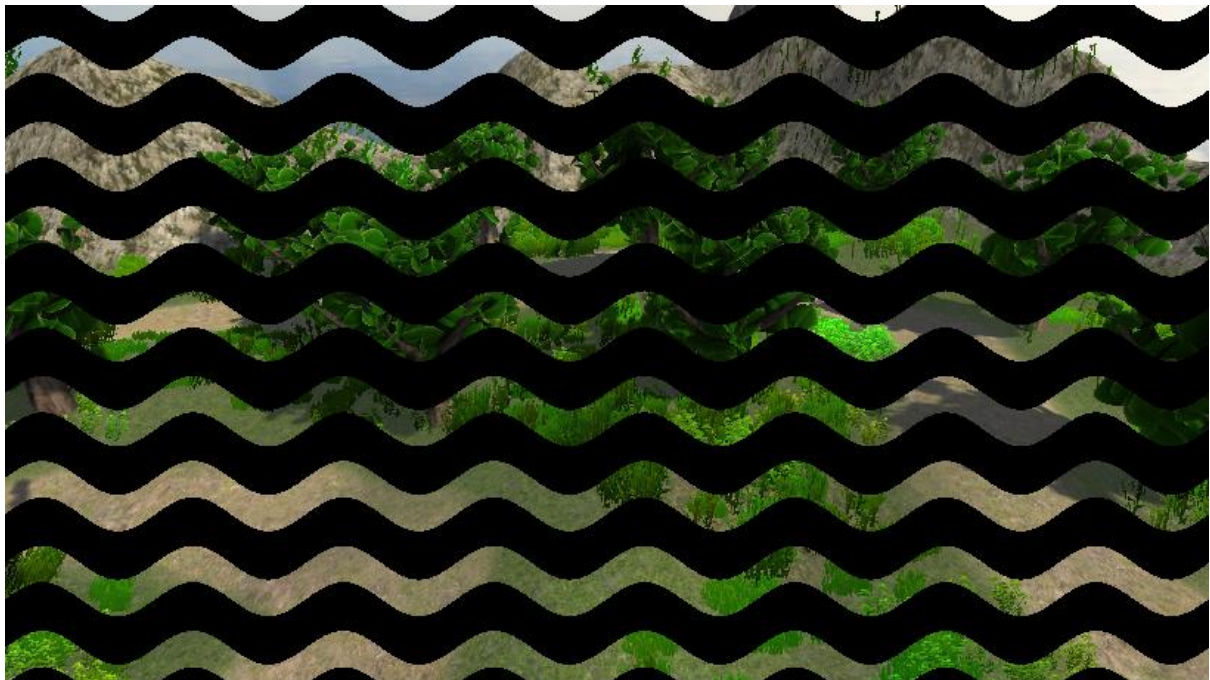
Camera Fade Pack (Version 1.2.1)

Camera Fade Pack is a set of 125 full-screen camera shaders, all of which support Shader Model 2.0, which provide a number of creative and professional ways of fading a screen in or out. A transition component complete with Unity Editor Extension is included meaning transitions can be set up in the Unity Editor and started with a single line of code at runtime. An example scene is included to illustrate how this works. The transition component allows up to three shaders to be merged together to create unique transitions.

Alternatively the shaders can be set up manually – each shader has a **value** property that is set between 0 and 1, with 0 representing the unaffected screen view before a transition and 1 representing the full transition. This gives you full control over how you want the transitions to work, and even allows you to fix a transition at a certain value to get a permanent screen effect. For example one of the **Pixelate** shaders can be set with a fixed value to give a pixelated screen view.

All shaders have optional parameters that allow fading to a background colour. Both the background colour itself and whether the fading should take place over the entire transition, over the second half of the transition or not at all, are configurable. Full source code for the shaders is provided.

Many of the shaders have additional parameters – for example any shader that uses a wave shape, such as the “Blinds Vertical Wave” shader, shown in action below...

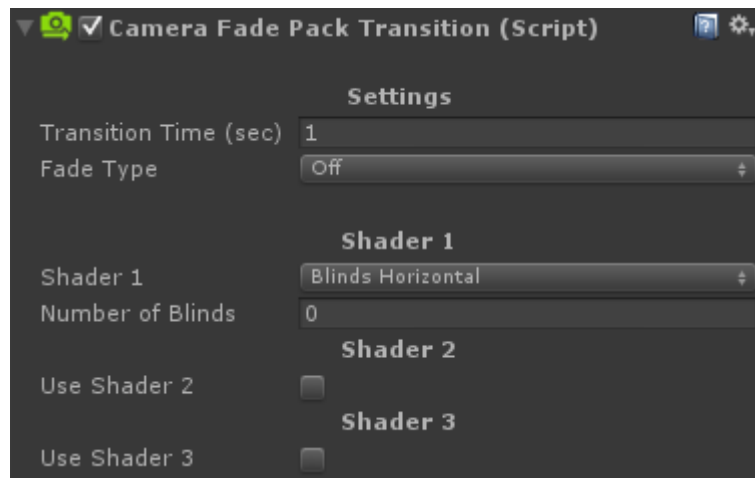


...will always allow the number of waves, the wavelength and the amplitude of the waves to be set via parameters, thus allowing any shape of wave you like.

Using the Transition Component

The transition component provides an easy way to configure a combination of transitions and all their associated parameters, and to execute them with a single line of code. To use the transition component, simply add a **Camera Fade Pack Transition** component to your main scene

camera. The component requires a camera component – if there isn't one on the game object, one will be added. Once you've added the component, it will be shown in the inspector like this:



From here you can decide on the shaders you want to be part of the transition, and set any associated variables. The configurable properties are as follows:

- **Transition Time** – This is the amount of time the screen takes to apply the transition and fade in or out. Defaults to one second, and doesn't have to be a whole number.
- **Fade Type** – Here you can select how you want the screen to fade, either not at all, to the background colour over the entire transition or to the background colour over the second half of the transition.
- **Fade Colour** – If the fade type is anything other than **Off**, the fade colour must be selected, which is the colour that the screen will fade to (e.g. black).
- **Shader 1** – The first shader used in the transition. At least one shader must be selected, and depending on the selection, a number of properties for that shader can be set. In the example above, the **Blinds Horizontal** shader has a single **Number of Blinds** property.
- **Shaders 2 and 3** – These shaders are optional and can be used to combine transition effects. Check the **Use Shader** checkboxes to enable them, after which they can be configured the same way as the first shader.

Once the component has been set up, a single line of code can be used to start the transition fading in or out. Simply call the **fadeIn()** or **fadeOut()** method on the component:

```
GetComponent<CameraFadePackTransition>().fadeIn();
```

The fade in and out methods are also overloaded to allow another method to be called when the transition finishes:

```
GetComponent<CameraFadePackTransition>().fadeOut(fadeOutFinished);
```

```
...
```

```
public void fadeOutFinished() {  
    Debug.Log("Fade out has finished!");  
}
```

This is useful if, for example, you want to change the scene after a fade out has completed.

Not Using the Transition Component

If you do not wish to use the transition component, you can still use the shaders on a full screen camera by setting the shader properties yourself. To do this, using a script on a game object that contains a camera, add the following method...

```
void OnRenderImage(RenderTexture sourceRenderTexture, RenderTexture destinationRenderTexture) {  
    material.SetFloat("value", 0.5f);  
    Graphics.Blit(sourceRenderTexture, destinationRenderTexture, material);  
}
```

...where **material** is a material that points to a Camera Fade Pack shader. The **value** property must be between 0 and 1 and specifies the amount to apply the fade, so you could transition this value yourself. For some shaders, additional properties must be set – all shaders have the **value** property and properties for fading to a background colour but many shaders have additional properties. You will have to look at the code for the shader you wish to use to determine what these properties are.