SHINE HIGHLIGHT SHADER

VERSION 1.0 - USER MANUAL

Shine Highlight Shader is an asset for Unity 3d to quickly and easily add an animated shine to your sprites and UI images. This can make elements of your game pop to call attention to them.

Features

- Different textures create different shiny effects
- Easily Change speed and pause between blinks!
- All blinks with the same config are synchronized to avoid distractions
- Multi-compile shader for improved performance
- Included script to manually control the animation
- Example scene to show the different configuration options

Table of Contents

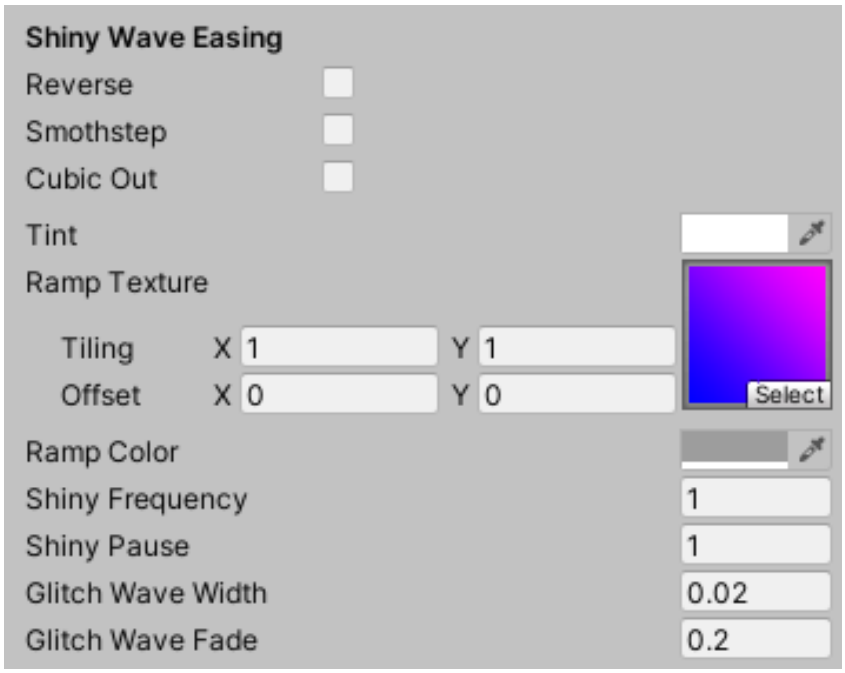
SHARKBOMB Studios

Getting Started

Check out the example scene to see the various included visual effects as well as explore the configuration options.

Once you are comfortable enough, take a look at the materials and create copies of those you plan to use in your project. That way you can safely change them without fear of an update to the asset overwriting your changes.

You may want to experiment with different **Ramp Textures** to get the effect you want. Be aware that you may want to change the compression on your Ramp Textures to high quality to prevent clearly visible bands in the gradient.



Also note that some of the materials have a slot for a ramp texture, while others (called standalone shaders) use the texture applied to the sprite or UI image instead.

Basic Usage

To use this asset you simply apply the appropriate material to the sprite or UI image in question. However since there are a number of shaders and materials included it may be a bit overwhelming.

Included Shaders

The shaders, and by extension the materials, all work similar at their core but when and how they are used differs. Firstly they are divided based on the object they are used on:

- **SpriteDefault** shaders for unlighted sprites
- **SpriteDiffuse** shaders for sprites that react to light
- **UI** shaders for UI image objects

These groups are then subdivided by way the shine is applied to the objects. The options are:

- **BlendAdditive** shaders put the shine on top of the texture used in the sprite or image by adding the color to the color of the source object. This will always create a lighter shine.
- **BlendAlpha** shaders will put the shine on top of the texture used in the sprite or image by replacing the color of the source object with the color of the shine.
- **Emission** shaders will put the shine into the emissive channel of the light-responsive source sprite and ensure that it is visible, even in darkness.
- **Standalone** shaders will directly show the shine as the UI image or sprite and source the gradient used from the sprite or image and not from the material.

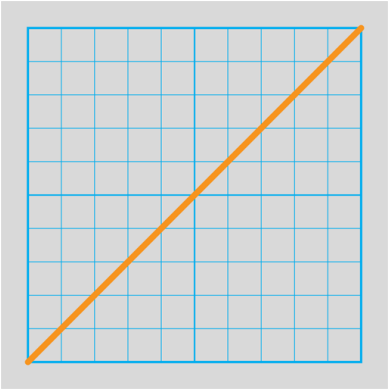
Lastly there are **basic** versions of these shaders, that constantly loop the animation, and the **manual** version that can be controlled via a script.

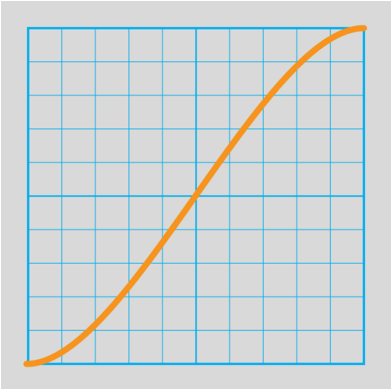
Material Options

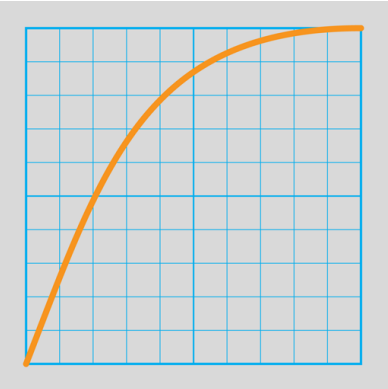
To use a shader it has to first be applied to a material. The asset comes with a set of materials but if you want to configure the shine further you may want to create your own materials to use.

Easing Options

All basic shaders can set the easing with which the gradient is animated. By default this is set to linear. But you have access to Smoothstep and Cubic Out easings as well and you can also reverse the direction.







Manual shaders do not have easing options.

Timing Options

To change the speed and timing you have two floats:

- **Shine Frequency** defines how many shine animations are shown in one second.
- **Shine Pause** defines the break between shine animations. It is measured in multiples of the duration of a single shine animation.

Visual Options

These two floats change the display of the shine itself:

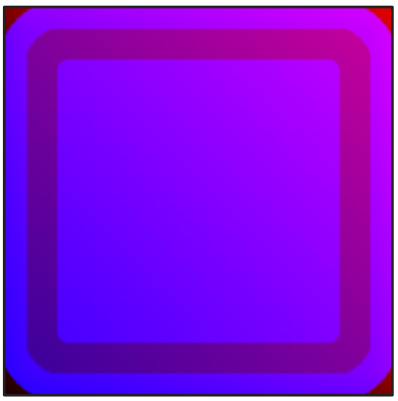
- **Shine Width** defines how wide the shine highlight is.
- **Shine Fade** defines how long the falloff of the shine highlight.

The Ramp Texture

A large part of the visuals of a shine effect is determined by the ramp texture. This defines how the shine animates and how strong it is.

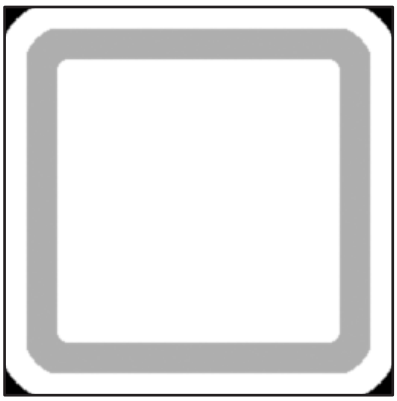
Color Channels

To enable this functionality two channels of the texture's RGB information are used. The red value of all pixels is used to define the timing and animation, the blue value of the texture are used to define the strength of the shine.



RGB values together

Red values only



Blue values only

Red Values: Shine Animation

The shine animation starts by showing the low red values. Then it moves over showing the middle values to eventually showing only the high red values. The timing of the animation combined with Shine Width and Shine Fade define which pixels are visible and how much.

Blue Values: Shine Strength

The strength of the maximum shine is defined by the blue values. Pixels with 0 blue will not get any shine at all, while areas with maximum blue value will get the full shine. Pixels in between will fulled accordingly.

If you have a consistent shape underlying the shine, you can have it react to that accordingly, with some parts more shiny than others.

Manual Triggering

Contrary to the basic or automatic shaders, the timing of the manual shine shaders is not generated by the shader itself but is instead fed into it from the outside. This is done by setting the **_TimeOffset** property on the respective material.

You may want to use the included **ShineControl** script. It allows you to trigger your shine or synchronize other effects to the animation:

Properties

- **Target Sprite / Target Image** defines the sprite or UI image whose material will be adjusted by this script. If both are set, only the sprite will be used.
- **Anim Curve** defines the easing of the animation with a range for time and value from 0 to 1.
- **Num Loops On Enable** sets the number of loops to run when the component is enabled. At 0 no animations will be triggered. A value below 0 will trigger an endless loop. A value greater than 0 it will loop that many times.

Triggering via Script

The ShineControl has a method that allows you to trigger the animation at will:

- **TriggerShine(int loops)** will immediately trigger an animation with the given number of loops. 0 loops will do nothing, a number less than 0 will create an infinite loop.

Animation Events

The control script exposes three UnityEvents that you can use to drive other behaviors based on the animation of the shine. These are:

- **OnShineShow** fires when a shine animation starts. It supplies the number of the loop being started (int) and the duration of the animation (float).
- **OnShinePause** fires when a shine animation ends and the pause begins. It supplies the number of the loop being done (int) and the duration of the pause (float).
- **OnLoopsEnded** fires when all planned loops have ended successfully.

Miscellaneous

Thank you for purchasing this asset and supporting future development.

Support

For additional support please turn to one of these channels:

- **Twitter**: @sharkbombs
- **Discord**: discord.sharkbombs.com
- **E-Mail**: support@sharkbombs.com

Credits

The following people were involved in creating this asset:

- **Martin Nerurkar** - Shine Shader asset
- **Caro Asercion** - Example icons (via game-icons.net)
- **Unity Technologies** - Basic UI and sprite shader