Gradient Shader Pack v3.0

Introduction

This is a pack of over 200 shaders. Each shader outputs a colorful gradient. The gradient might be combined with one or two palette textures, and/or a sprite texture, depending on the shader. All material properties can be adjusted in realtime via script or animation clip.

Requirements

Minimum version of the Unity Editor for all shaders is 2019.2. This means the shaders no longer support ANY prior versions of Unity.

All shaders require and are built with the Unity Shader Graph editor (required package). The shader graph editor in earlier versions (e.g. 2019.1) do not support Sprite shaders.

There is a folder named “Internal - Required for ALL shaders”, containing “sub graphs”. These are small pieces of reusable shader code and are necessary in order for the shaders to run. All shaders use at least one of the subgraphs. Whichever shader you use in your project, you’ll **need to include this folder in your project also**, otherwise the shader’s will be broken.

All gradients are generated based on input **UV coordinates**. Non-sprite textures use UV0 channel to generate the texture shape. Sprite (textured) gradients use UV0 for the sprite texture itself, so UV1 is used for the gradient. Sprites and other Unity geometry objects usually include UV0 and UV1 configured in a 0..1 range as normal. Geometry must contain UV coordinates!

Shader Graph shaders **only work with the Unity Scriptable Render Pipelines** such as the Universal/Lightweight Render Pipeline. If you use the Lit Sprite shaders, you’ll need to set up a 2D lighting environment including 2D lights and appropriate render-pipeline settings.

Color Space has an influence on the generated gradients because it affects the renderer’s interpretation of color. Gamma space may generally produce better blends of gradients on some shaders, but all shaders work in Linear or Gamma space. The output may look visually a little different on each.

What’s New in 3.0

A COMPLETE REWRITE: Version 3 is a complete rewrite of the entire pack from scratch. The old pack, version 2.2, was built for Unity 3.0 using hand-written OpenGL shaders. This was several years ago. Newer Unity technologies, faster mobile devices and new graphics API’s called for a complete rewrite.

MORE SHADERS: The old pack featured 85 shaders, the new pack features over 200.

MODERN TECH: All shaders are based on the Unity Scriptable Render Pipeline in order to support modern platforms and Unity features. They will not work without it.

FULLY EDITABLE: All shaders are fully editable in the Unity Shader Graph editor, and are stored as Shader Graphs. Every shader also uses some Sub Graphs which must be included in the project in addition to the main shader file. See the “Internal - Required for ALL Shaders” folder. All shaders can be modified to add or remove or adjust features. Note that shaders based on Gradient Shader Pack cannot be re-released as a derivative work as asset-store assets.

SPRITE AND UI SUPPORT: A new set of 40 sprite shaders has been added including 20 unlit and 20 lit shaders. The lit shaders are currently experimental. Sprite shaders work as normal Unity sprites and can be also used on Unity UI elements such as buttons. Note that sprite shaders are automatically alpha-blended. Sprite shaders replace old “textured” versions and are now the main way to combine a texture with a gradient. (Alternative is to render a textured object first, then render a Multiply Blend gradient on top of it.)

CLASSIC GRADIENT TYPES: Standard gradients include a more useable angular gradient, diamond gradient, the vertical gradient now doubles as horizontal (set angle = 90). Existing shaders such as Box4, Box6 and Box9 are included. Box3 is obsolete (use box4 with two colors the same).

NEW NOISE GRADIENTS: Three types of noise shaders have been added - one noise field, 3 noise fields, and 3 fields independently layered. There is also a noise shader which combines with a Box4 to transition between Box4 colors and noise colors.

NEW BLOB GRADIENT: A new meta-ball/blob-based gradient has been added featuring 4 individually controllable blobs of color which combine to produce smoothly curved gradients. These allow you to individually move the location of multiple colored blobs.

MORE BLEND MODES: All shaders except sprite shaders (which are automatically alpha blended) are now available in solid, alpha blended, additive and multiply blend modes. This means the output from the shader can be overlaid on top of the background or other objects as multiple layers. Additive blend is useful for lighting effects, while multiply is useful for shading effects or for “colorizing” parts of the scene.

SUPPORT FOR LIGHTING: PRB versions of the shaders now support lighting and shadows per Unity’s Scriptable Rendering Pipelines. This means the shaders can be more effectively used in 3D environments, or with new 2D lighting. These shaders support light, specular highlights, and shadows. The lit sprite shaders also support normal mapping.

MORE FLEXIBILITY: All shaders can be adjusted. The UV coordinates can be translated, rotated, and scaled which “positions” and scales the gradient effect. Colors can of course be adjusted and any transparency in the color will mix with any transparent from a texture (if any). Palettes can be animated by adjusting their horizontal or vertical offset within the palette texture. 2D palette textures are also now supported to allow another dimension of animation. UV coordinates can also be reversed in the material properties instead of a separate shader.

BETTER PLATFORM SUPPORT: Since the new shaders are built with Shader Graph, Unity can automatically build optimized shaders for multiple platforms and graphics APIs. They can work with various rendering pipelines. The shaders are still performant on mobile devices, especially since modern mobile devices have better hardware, but also via the Lightweight/Universal Render Pipeline. Most shaders are cheap but blob/noise are more expensive.

SMOOTHER SKY PALETTES: All included “sky palette” palette textures have been smoothed to remove banding and produce very smooth gradients.

NEW EXAMPLES: Includes 6 new example scenes. Hit play to see some animation.

Palletized Shaders

Palletized versions of shaders pull from a **“palette texture”**. The texture can be any size but will be smoother if it’s larger. For a one-dimensional palette, e.g. a 2048x1 texture is ideal. If you need more color control, use a larger texture. If you need less color control, you can use a smaller texture with bilinear filtering to automatically interpolate the colors.

You can also use a **two-dimensional palette texture**. This is a texture (any size/shape) which has multiple “rows” of palettes. Palette rows can be selected in the material properties using the “Palette Offset Y” texture coordinate. And since this value is exposed, it can also be animated/modified by animation clips or scripts. This allows an entire texture-based gradient to shift towards an entirely different palette or palettes over time. It also allows you to select between multiple palettes stored in a single texture. All palletized shaders allow you to adjust the horizontal UV coordinate and the vertical UV coordinate.

For shaders with **two palette textures**, you can either use the same texture twice (with or without different UV coords) or you can use two separate textures. Gradients will be generated interpolating “between” the two palettes.

Colors from the palette can include **an alpha channel**, if your palette texture is imported in an alpha-channel-supporting format e.g. RGBA.

Texture sampling also affects the smoothness of the palette. In the texture import settings for the palette texture, consider also that “Point” sampling will produce more banding (especially for smaller textures) but also give more “accurate” intended colors. While “Bilinear” sampling will produce more interpolated/smoothed gradients (in between each color). To produce nice smooth palette-based gradients in general you should apply some kind of blur to the pixels in the palette image file.

The noise shaders will first generate the noise and then reinterpret the noise values as horizontal offsets in the palette texture. Brighter samples will convert to higher UV coordinates (later palette colors) and darker samples to lower UV coordinates (earlier palette colors).

Note that **the Blob shader** cannot use color inputs while also trying to apply a palette, because of the way meta-balls are calculated and combined. Instead, colors will be converted to grayscale in the shader before being converted to a palletized version.

Materials

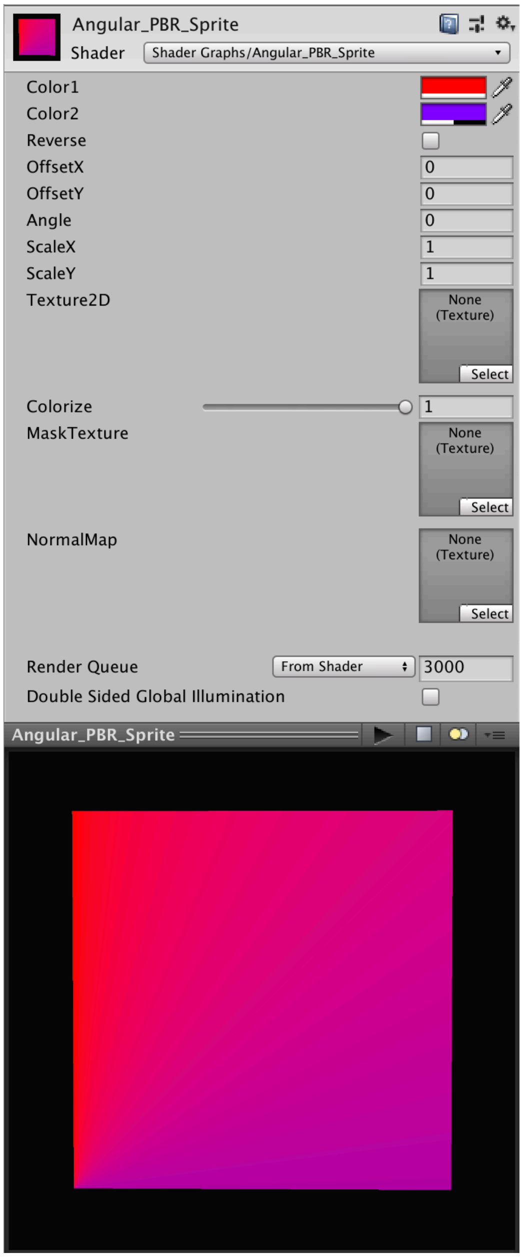
A typical material a shader looks like the image below (example for a lit sprite):

Color fields are for selecting start/end or intermediary colors. Alpha values can be adjusted for blended shaders.

“Reverse” reverses the direction of the UV coordinates, thus reversing the gradient.

OffsetX and OffsetY scroll the position of the gradient. For gradients such as radial or diamond which center around 0,0 you’ll want to scroll the gradient to center the shape.

Angle rotates the gradient around its origin.



Angular_PBR_Sprite

Shader: Shader Graphs/Angular_PBR_Sprite

Color1 🟥

Color2 🟪

Reverse □

OffsetX 0

OffsetY 0

Angle 0

ScaleX 1

ScaleY 1

Texture2D None (Texture)

Colorize 1

MaskTexture None (Texture)

NormalMap None (Texture)

Render Queue From Shader 3000

Double Sided Global Illumination □

ScaleX and ScaleY zoom the size of the gradient. Since they can be adjusted individually, you can e.g. warp a radial gradient to produce an ellipse, distort a diamond, stretch a gradient etc.

For a sprite shader you'll need a Texture2D sprite image which will be colorized (multiplied) with the gradient. There are also optional mask and normal maps.

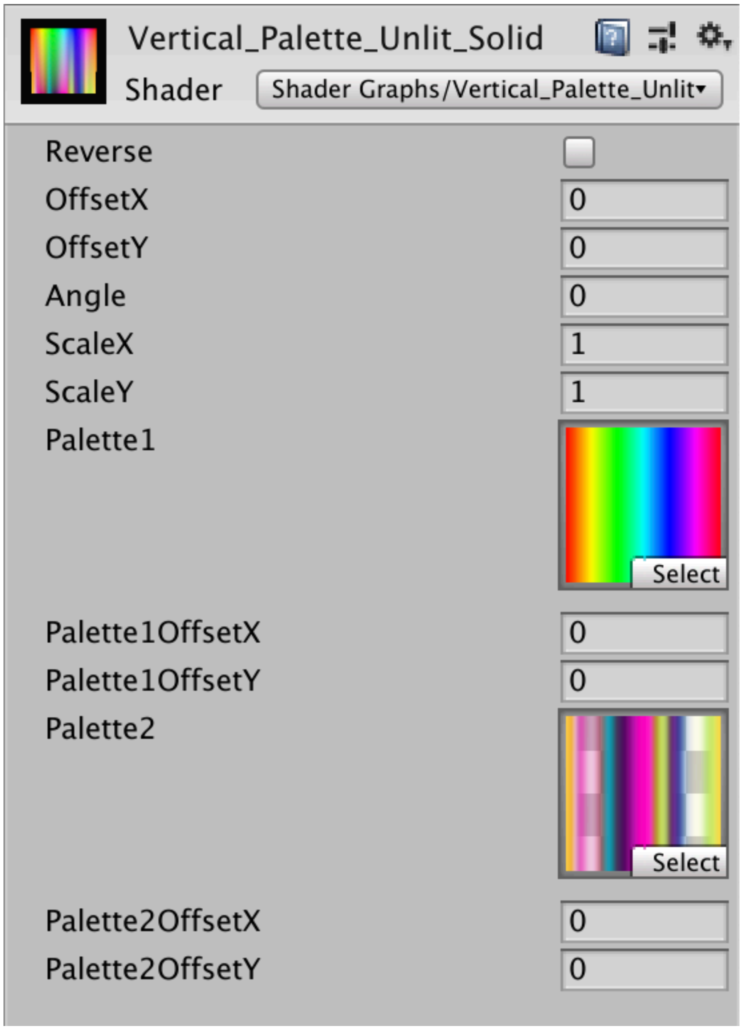
Adjust the Colorize slider to adjust how much the gradient is blended with the texture.

Palletized shaders will include further fields:

Palette1 (or Palette) is a palette texture, 2048x1 for example (2d is also allowed).

Palette1OffsetX and Palette1OffsetY adjust the UV coordinates of the palette row being used. To scroll/rotate the palette, adjust the Palette1OffsetX. To select a different row of colors in the texture, adjust the Palette1OffsetY.

Palette 2 is a second palette texture, 2048x1 for example (2d is also allowed). It also has its own X and Y offsets. You can use the same texture of a different texture.



Vertical_Palette_Unlit_Solid

Shader: Shader Graphs/Vertical_Palette_Unlit▾

Reverse	□
OffsetX	0
OffsetY	0
Angle	0
ScaleX	1
ScaleY	1
Palette1	
Palette1OffsetX	0
Palette1OffsetY	0
Palette2	
Palette2OffsetX	0
Palette2OffsetY	0

Questions

Any questions can be directed to support@venus12.com