PALETTE FX

Version 1.0

Thank you for buying PALETTE FX!

How to use

PaletteFX has two modes, 32bit and 8bit, each one requires a different setup.

- 32bit mode uses normal RGBA textures, simple and easy to work with.
- 8bit mode uses pseudo 8bit textures, to simulate a retro look based on palettes. The advantage of this mode is that you can change colors in a more controlled way, cycle colors in a palette, or even apply fullscreen effects just by altering the palette texture (see Tricks section).

32bit mode

32Bit mode let's you use normal textures, which you can then combine with a mask to change the color of different parts in the texture.
Using it is simple.

1. Create a new Material in Unity.
2. Change the Material shader to PaletteFX/32bit.
3. Add a Mask texture (in the same format as explained in the Mask section)
4. Configure the Hue and Shade shifts in the shader.
5. That's it!

8bit mode

8bit mode is a bit more complex, since textures need to be converted into a pseudo 8bit format. First you should create your textures with a certain palette in mind, as in, only using RGB values from that palette.
You will also need to have that palette imported into Unity (see next section).

After you have both, go to Tools → Palette Processor and select the texture you want to convert, along with the correct palette and press Convert.
A new texture will now appear in the path you specified (or if you put nothing as path, it will appear just in the project root).

When using the Palette Processor tool, note that while its possible to use any texture with any palette, results will usually be good only when the source image was created with that palette in mind, otherwise you can try different dither modes until you get something satisfatory.

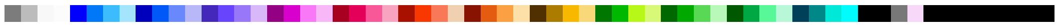
The converted texture will be a greyscale image, that depending on the palette used could look a bit weird. You need now to select that texture and in the inspector do the following changes to make the texture a "Pixel Perfect" texture:
"Filter Mode" → Point
"Compression" → None

After this, create a new material and do the following changes:

- Select PaletteFX/8bit as shader
- Set the converted texture as main texture
- Set a texture mask (See Mask section)
- Set a palette texture (The same that you used to convert the texture!)
- Set the correct number of shades and hues in the palette
- You can optionally configure the Hue and Shade shifts
- You can enable Dithered Alpha (see FAQ)

Creating Palette textures

A palette texture is just a texture that contains a sequence of colors. The height should always be 1, and the width is recommended to be a power of 2 (eg: 32, 64, 128, etc).
The colors should be ordered by hue, and each hue should be ordered by shade, from dark to light.
In case there not enough colors to make reach a “power of 2” width, you can add extra black colors to pad it to the pretended width, like in this example.



Creating Mask textures

A mask texture let's you separate specify different parts of a sprite, in order to apply different colors to them. By using a Mask, a texture can up to 3 separate parts.

A mask texture can only contain 3 colors:

- pure red, RGB(255, 0, 0)
- pure green, RGB(0, 255, 0)
- pure blue, RGB(0, 0, 255)
- you can use any color you want for transparent parts, it wont matter.

The colors map each part of a sprite to a different Hue/Shade channel, which are Vectors exposed in the PaletteFX/8bit shader.
Red maps to X component, Green to the Y component and Blue to the Z component.
The W component is unused for now.

The values in the Hue / Shift vectors will be floats that can be either positive or negative, and represent how much to add / subtract for the pixel hue / shade.
Both hue and shade will be properly limited in the shader, they can't go out of range even if you specify large numbers here.

IMPORTANT - In order to avoid color artifacts, the Mask texture should also be setup as Pixel Perfect, as explained in the 8bit section.

8bit mode Tricks

In 8bit mode textures colors will be represented as an index in the palette instead of an RGB color. Since PaletteFX expects palette textures with ordered hues/shades, this means that the shader can properly decompose an index into a (hue, shade) pair.

Note that by sharing a palette texture with all of your sprites, effectively altering this texture pixels will affect every 8bit sprite that uses that palette.
So, by doing that you can do advanced effects like glows, fades, hit effects and others.
In the next version of PaletteFX there will be some helper scripts for this.

Controlling Hues and Shades programmatically

It is possible and easy to adjust the Hue and Shade of a sprite at run-time.
In your own scripts, you just have to get hold of the Renderer component and manipulate its Material properties via C#.

Eg:

```
private Material spriteMaterial;
void Start() {
	var renderer = this.GetComponent<MeshRenderer>();
	this.spriteMaterial = renderer.material;
	this.spriteMaterial.SetVector("_HueShift", new Vector3(hue1, hue2, hue3));
	this.spriteMaterial.SetVector("_ShadeShift", new Vector3(shade1, shade2, shade3));
}
```

Of course, you can also change the properties in the Update() method, lerp them over time, etc, be creative!

FAQ

Q - **When using 8bit mode, the sprites colors appear all garbled!**
A – Check if all of the following is correctly done:

- The sprite must be using the same palette used when you converted it. Optionally an equivalent palette with exactly same size and color order is possible.
- The number of shades and hues is correct in the material
- Both the sprite texture and palette texture have Filter Mode as Point and Compression is disabled. Optionally check if the Mask too has the same settings.
- Also double check if the original image was created using only colors from the specified palette, otherwise artifacts might appear.

Q – **Why is the W component of the Hue/Shade shift unused?**
A – In theory, we could use this for an extra channel, letting the sprites having 4 different parts instead of 3.
However this would require mapping the Alpha channel of the Mask texture to this component, which would make masks more difficult to create. If you need this, contact me and I'll add this in a future update.

Q – **My sprites had black outlines but shifting the Hue in 8bit mode makes the outlines turn into different colors!**
A – This usually happens when the palettes darkest shades are not dark enough.
There's two solutions, either edit the palette to have the same color as the darkest shade in all hues, or put the outlines in a separate channel in the Mask.

Q – **What are the Dithered Alpha shaders?**
A – Many old consoles were unable to do real alpha blending.
By using the Dithered Alpha shaders, any sprite with transparency will not be Alpha Blended, but instead a dithering effect will be used, that does not introduce any new colors not present in the palette.
Use this for the ultimate retro look!

Support

If you have any questions, suggestions, comments or feature requests, please send me a mail to sergio.flores@lunarlabs.pt