HueShift Documentation

With the HueShift shader you can easily change the color of a material to add addtional variety to your assets. Compared with the basic approach of multiplying a color to a texture, you can limit the color change to a specific hue range. These exclusions allow you to change the hue of a texture partially.

Requirements

Unity 4.1 required for the material inspector

Beside of the custom inspector, the shader should work on older versions as well as it does not rely on any other fancy stuff.

Shader Model 2.0

Took a while, but it compiles with less than 64 instructions making it usable in SM 2.0

Does not work in Flash builds

I'd need to save one additional register to make it compilable with Flash. Since this might result in exceeding the 64 instruction limit and thus forcing Shader Model 3.0, I do not plan to adapt the shader to Flash builds (especially since Unity will drop Flash support either).

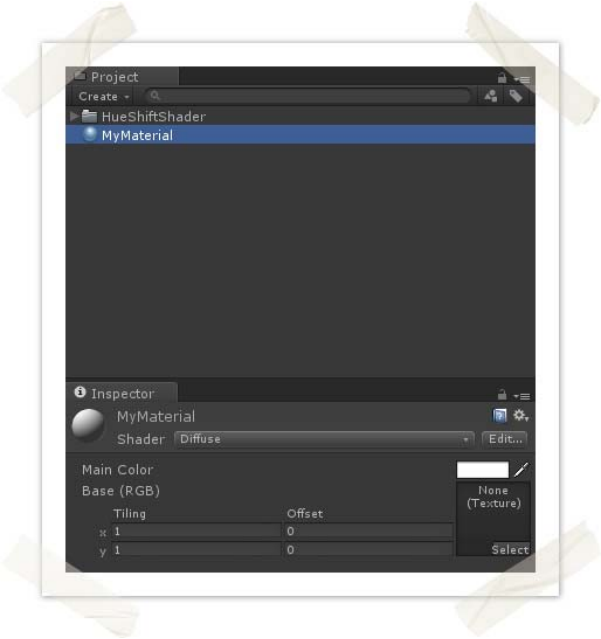
Platforms

Works on WebPlayer, Windows, Mac OS and Linux builds.

Mobiles and Consoles are not tested yet. Any feedback - especially for consoles - is highly appreciated

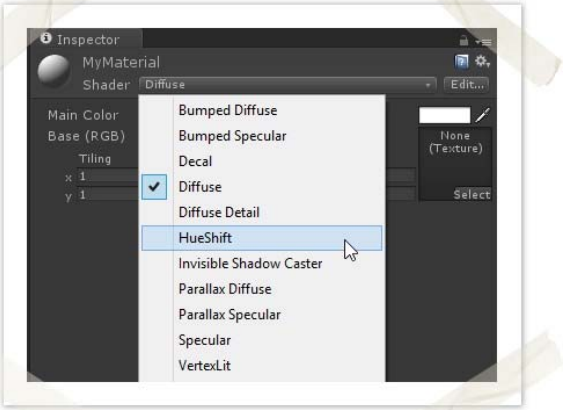
The basics

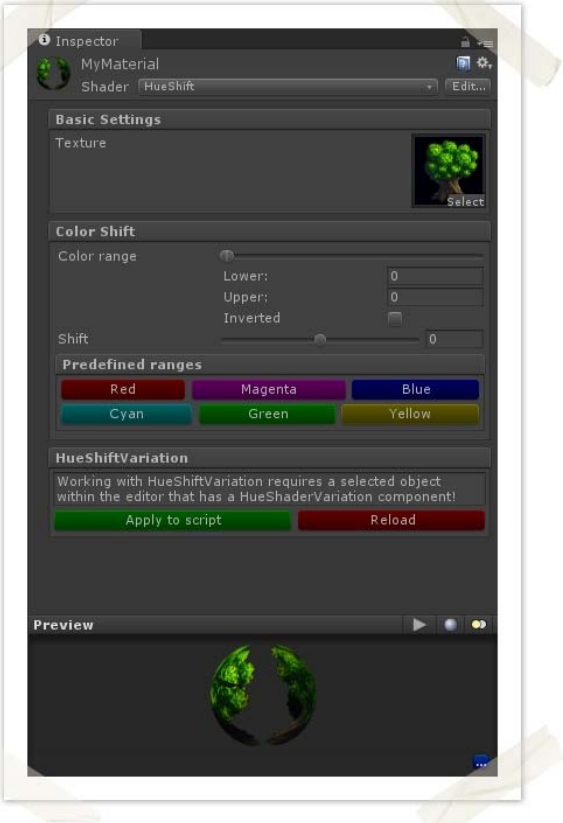
Create a new material for the texture you want to use.



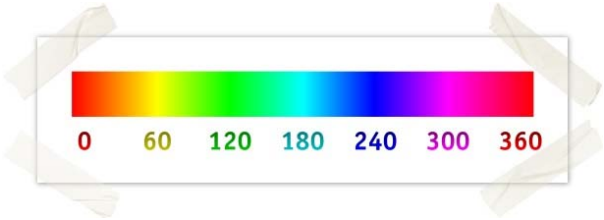
Change the default *Diffuse* shader to *HueShift* and drag the texture of your choice into the Texture field

If you are using Unity 4.1 or newer, the HueShift shader will use a custom material inspector. On older versions you will probably get errors because of the missing MaterialEditor-API. You need to delete the custom editor, the shader itself does not rely on any 4.x features (although working without the custom inspector might be a hassle. If you need to provide values without the custom inspector, be sure to divide them by 360 and regard the limits)





In our example, we want to change the color of a tree texture. First, we need to select a color range to be used for shifting. For those who are not familiar with the HSV color space: http://en.wikipedia.org/wiki/HSL_and_HSV



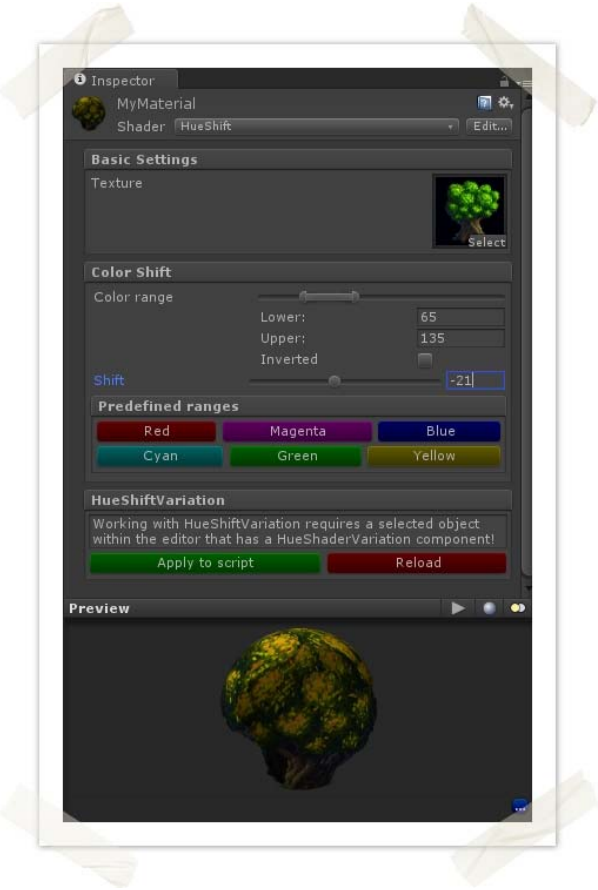
We need to give a lower and an upper limit for our shifting. The problem is that red is split between 0..20 and 340..360 (more or less). To use red as color range, we select all other colors (e.g. 21 .. 339) and select *Inverted*. This will advise the shader to use all other colors except the selected range. Some color ranges are predefined. Just click on one of the buttons below and the shader will be adjustet to the corresponding range:

Now the final step: The *Shift* slider determines the amount of color shift. It works in both directions from -180 to +180 allowing you to map any input range to any output range.

In this example, we want to give our green summer tree an autumnal look. Since the leafs in a green range, we can use the predefined green color range (75 - 135). A slight shift towards red of -24 turns should be enough.

Finetuning

The predefined color ranges are a for starters, but you need to adjust both limits to match the desired range for your texture. In our case, we still have some green parts in our leafs left (in case you don't see them, just make an extreme shift to -180 and you will see that not all leafs are red). For our example tree a lower range of 65 instead of 75 does the job.



Material Instancing

If you use several trees within your scene you will notice that all trees are shifted the same. This is because they share the same material. If you want to have trees with different shiftings you need to create multiple materials.

> Each new Material results in one additional draw call. Try to reuse as many of them as possible.

To simplify the task of material instancing, a simple helper script called *HueShiftVariation* is included. Attach it to a game object, provide the desired settings for color range and shifting and when running your game, the necessary material is instantiated automatically. Currently, it does not provide any caching or reusing of materials.

In the material inspector of the shader, you will notice an additonal section called *HueShiftVariation*. It's a small helper when working with the *HueShiftVariation* script. Click on *Apply to script* copies the current color range and shifting value of the shader into the HueShiftVariation script. Click on *Reload* to copy the values from the script into the shader.

Invisible Shadow caster

The HueShiftShader does not provide any cutout features and therefore does not allow objects to cast shadows. As a quick solution an additional shader is included in the package: *Invisible Shadow Caster*. It works just like the usual cutout shader but does not draw the object itself, i.e. it creates an invisible object that casts a shadow.

Support

You can contact me by mail at support@tastenhacker.de