E07: Color Flash

Cocos2D's default shader allows you to tint a sprite by it's color, but there are other things you might want to do with the color. One thing that you often see in old 2D games is to make a sprite flash a solid color. This is a very easy effect to do in a shader.

To make it a little more interesting, let's say you want to have a separate flash and tint color. We'll have to pass the flash color into the shader in a different way.

```
uniform vec4 u_ColorFlash;

void main(){
  vec4 color = texture2D(cc_MainTexture, cc_FragTexCoord1);

  // Composite the flash color over the texture color.
  // mix() is a builtin GLSL function that does linear interpolatio
n.
  color.rgb = mix(color.rgb, u_ColorFlash.rgb*color.a, u_ColorFlash.
a);

  gl_FragColor = cc_FragColor*color;
}
```

Simple enough, but let's break it down:

- The `uniform` keyword on the `u_FlashColor` tells GLSL that the variable is set from your program. More on that below.
- GLSL's builtin `mix()` function implements run of the mill alpha blending.
- Since the flash color is being masked by the texture's alpha, we need to premultiply it by the texture's alpha.
- Finally tint the composited color.

Uniform Variables:

Uniform variables are variables that you can set from your Objective-C code. They are "uniform" because the value will be the same for every pixel in a sprite. You can use them to pass colors, positions, offsets, textures and matrices to a shader.

You've actually used a number of uniform variables already. Many of Cocos2D's builtin shader variables are uniforms. `cc_MainTexture` and `cc_Time` are both uniforms. [1]

There are two ways to set a uniform variable in Cocos2D, both using dictionaries to store the values. The fastest way is to use the `CCDirector.globalShaderUniforms` dictionary. These global uniforms are shared by all shaders on every object. You can also set uniforms for specific sprites. The reason why global uniforms are faster is Cocos2D cannot automatically batch the rendering of sprites that have custom uniform values.

```
// Pass colors using CCColor objects.
sprite.shaderUniforms[@"MyColor"] = [CCColor redColor];

// Pass floats using NSNumber objects.
sprite.shaderUniforms[@"MyFloat"] = @(123.456);

// Pass positions, offsets, or sizes using NSValue objects.
sprite.shaderUniforms[@"MyPoint"] = [NSValue valueWithCGPoint:
point];
sprite.shaderUniforms[@"MySize"] = [NSValue valueWithCGSize:si
ze];

// Global uniforms work the same way, but are set on the CCDir
ector.
[CCDirector sharedDirector].globalShaderUniforms[@"FortyTwo"]
= @(42);
```

For a full list of shader uniform types, see the Cocos2D wiki or peek at the CCShader.m code.

Exercises:

- Try rearranging the shader to apply the tinting before applying the flash while still preserving the sprite's alpha.
- Try modifying the shadow shader to use a uniform for the shadow offset.

---

1. Later you'll also learn about GLSL's `attribute` and `varying` variable types.↩