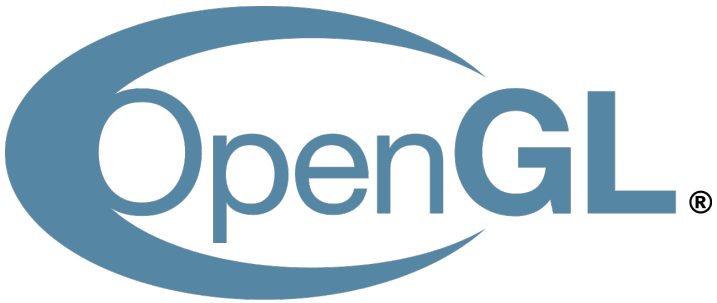
Intro to GLSL and Shaders

Dane Christensen, Brigham H. Keys, Esq.

Brigham Young University - Idaho

bkeys@bkeys.org

September 4, 2015



A little about shaders

Dane Christensen, Brigham H. Keys, Esq.

- There are multiple shading langauges, such as HLSL, GLSL, and Kronos is releasing SPIR-V later this year.
- Just like anything you write in C++, shaders have source code, files, and are compiled.
- GLSL's syntax is very similar to the C programming language, so hopefully the learning curve will be minimal.
- Shaders are stored in IDs, which are ints.

Types of Shaders

Dane Christensen, Brigham H. Keys, Esq.

There are two different kinds of shaders that together create a shader program, both are required for the shader to work. Chances are you are not going to need to write your own shader, as there are plenty of well written shaders online.

Vertex Shaders

- Changes the position of a vertex (Very similar to a GL translation)
- Capable of changing the color of a vertex

Vertex Shaders

- Uses lighting, normals, textures, etc.
- Sets the color of individual pixels

About making a shader program

Dane Christensen, Brigham H. Keys, Esq.

- Compiling a shader successfully gets us an ID for that shader.
- Requiring both a fragment shader, and a vertex shader.
- Compiler errors are possible, but checking for then is optional.
- A shader program is created by linking the fragment shader to the vertex shader.
- The shader program itself has an ID.
- Very rarely can a shader not be re-used

Examples of GLSL Program

Dane Christensen, Brigham H. Keys, Esq.

Vertex Shader

```
File Edit Options Buffers Tools Help
attribute vec2 coord2d;
attribute vec3 v_color;
varying vec3 f_color;

void main(void) {
  gl_Position = vec4(coord2d, 0.0, 1.0);
  f_color = v_color;
}
```

Fragment Shader

```
File Edit Options Buffers Tools Help
uniform float fade;
varying vec3 f_color;

void main(void) {
  gl_FragColor = vec4(f_color.x, f_color.y, f_color.z, fade);
}
```

Compiling into a shader

Dane Christensen, Brigham H. Keys, Esq.

Getting the program ID

The function needed to compile and get the ID for the shader is
GLuint glCreateShader(GL_VERTEX_SHADER);

Create the shader program

The function to bind the source code to the shader (this must happen before compilation)
void glShaderSource(GLuint shader, GLsizei count, const GLchar **string, const GLint *length);

Create our shader!

Now we compile ths shader with:
glCompileShader(GLuint_ID);

Using our shader we created

Dane Christensen, Brigham H. Keys, Esq.

Let OpenGL know you are using a shader

void GLuint glCreateProgram();
This behaves very much like a constructor.

We now attach both shaders to the program

void glAttachShader(program_ID, shader_ID);

Create our shader program, put it together

void glLinkProgram(program_ID);

Place this before you begin to render

void glUseProgram(program_ID);

References

 https://www.opengl.org/ (2015)