GLSL: OpenGL Shading Language

History: Shading Languages

- Shade Trees
  - Proposed by Rob Cook in 1984
  - Tree structure used to represent:
    - Surface normals
    - Material properties
    - Texture maps
    - Light properties

History: Shading Languages

- Pixel Stream Editor (PSE)
  - Proposed by Ken Perlin 1985
  - Procedural programming language
  - More sophisticated than Shade Trees
    - Language supported flow of control functionality (selection and iteration constructs)
    - Provided stochastic noise functions for surface appearence

History: Shading Languages

- Renderman Interface Specification
  - Proposed by Pixar in 1988.
  - Industry standard in visual effects industry.
  - Based on Shade Trees.

GLSL: OpenGL Shading Language

- OpenGL Shading Language (GLSL)
- C like syntax and control functionality such as:
  - Looping constructs (for, while, do-while).
  - Conditionals (if-then, if-then-else).
  - Variables.
- GLSL is an interpreted language.

GLSL: Data types

GLSL data types:

- scalers
- vectors
- matrices
- samplers

GLSL: Scaler data types

Scaler data types:

- int (integer)
- uint (unsigned integer)
- bool (boolean)
- float (floating point)

GLSL: Vector data types

Vector data types:

- vec2 (two-dimensional vector)
- vec3 (three-dimensional vector)
- vec4 (four-dimensional vector)

GLSL: Matrice data types

Matrice data types:

- mat2 (2 by 2 matrix)
- mat3 (3 by 3 matrix)
- mat4 (4 by 4 matrix)
- mat**M**x**N** (user-defined matrix with M columns by N rows)

GLSL: Sampler data types

Samplers: a special GLSL data type bound to texture data.

GLSL arrays and user-defined structures.

- Note: User-defined structures and arrays of the GLSL data types can also be created.

- Example:

```
struct lightsource
{
    vec3 color;
    vec3 position;
}
```

GLSL example: toon shader teapot

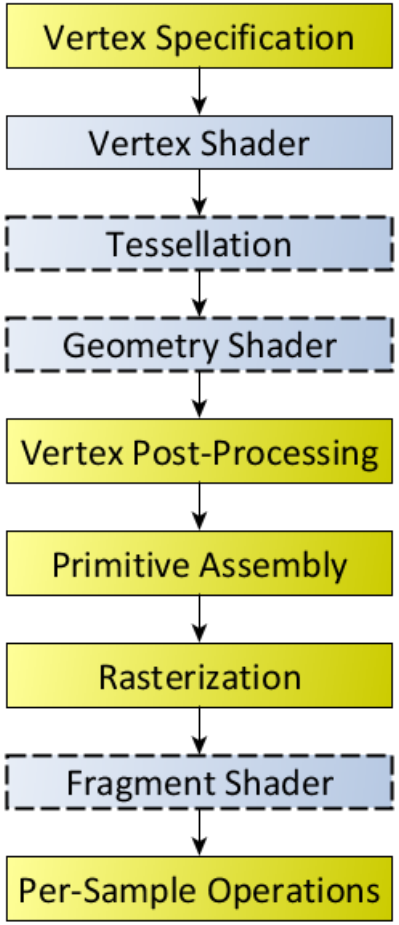
Define the GLSL shader files:

- Fragment shader
- Vertex shader

Create a generic loader to:

1. Creates an OpenGL canvas
2. Load then interpret shader files (above)
3. Apply shader files to a teapot model
   (a standard model in OpenGL library).

GLSL rendering pipeline



Source: https://www.opengl.org/wiki/Rendering_Pipeline_Overview

GLSL example: toon shader teapot

Fragment shader (toon.frag):

```
void main()
{
  gl_FragColor = gl_Color;
}
```

Source: http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/directional-lights-i/

GLSL example: toon shader teapot

Vertex shader (toon.vert):

```
void main()
{
  vec3 normal, lightDir;
  vec4 diffuse;
  float NdotL;

  normal = normalize(gl_NormalMatrix * gl_Normal);
  lightDir = normalize(vec3(gl_LightSource[0].position));
  NdotL = max(dot(normal, lightDir), 0.0);
  diffuse = gl_FrontMaterial.diffuse * gl_LightSource[0].diffuse;
  gl_FrontColor =  NdotL * diffuse;
  gl_Position = ftransform();
}
```

Source: http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/directional-lights-i/

GLSL example: toon shader teapot

For further reading…

- OpenGL Shading Language:
  http://www.opengl.org/registry/doc/GLSLangSpec.4.40.pdf
- *OpenGL 4 Shading Language Cookbook* (2nd Edition) by David Wolff.
- OpenGL Shading tutorial:
  https://www.opengl.org/sdk/docs/tutorials/ClockworkCoders/lighting.php
- Toon shading tutorial:
  http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/toon-shading/