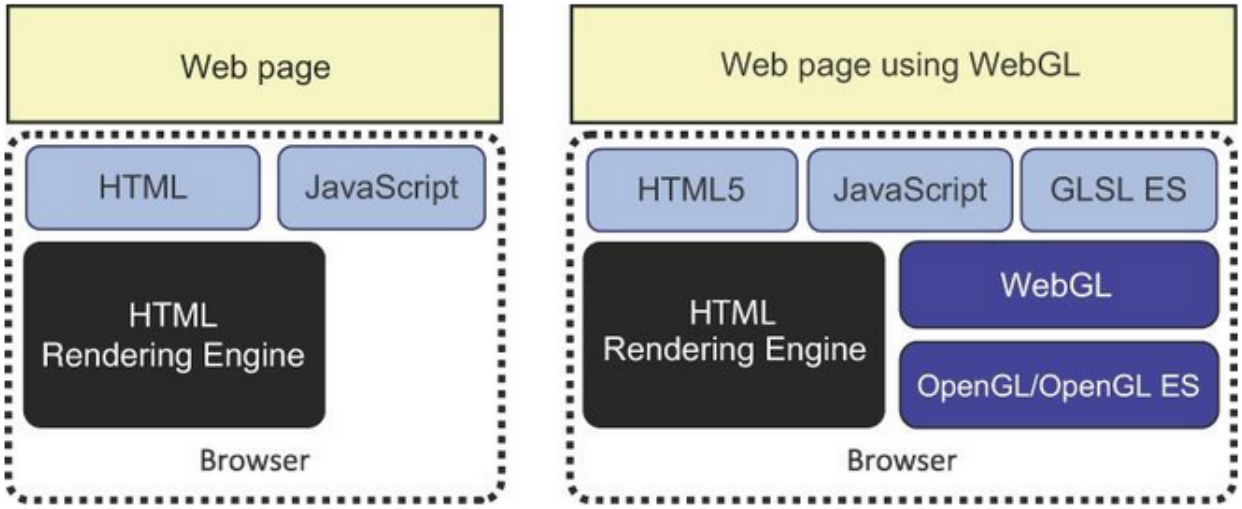
CMSC 161
Interactive Computer Graphics
Meeting 01 - Introduction to WebGL

What is WebGL?
WebGL is a technology that enables drawing, displaying, and interacting with sophisticated interactive *three-dimensional computer graphics* from **within browsers.**



<canvas> tag
File: 01-01-canvas2d.html
Introduced in HTML5, canvas tag offers a convenient way to draw computer graphics dynamically using JavaScript

Three steps required to draw 2D computer graphics on the canvas using Javascript
1. Retrieve the <canvas> element
2. Request the rendering *context* for the 2D graphics from the element.
3. Draw the 2D graphics using the methods that the context supports

Your First WebGL Program
File(s): 01-02-webgl.html,webgl-init.js

The same steps above also applies to WebGL except there are some modifications on step 2 and step 3
1. Retrieve the <canvas> element
2. Request the rendering *context* for the 2D graphics from the element.
    - WebGL Rendering Context
3. Draw the graphics using the methods that the context supports
    - Using built in functions or shaders

CMSC 161
Interactive Computer Graphics
Meeting 01 - Introduction to WebGL

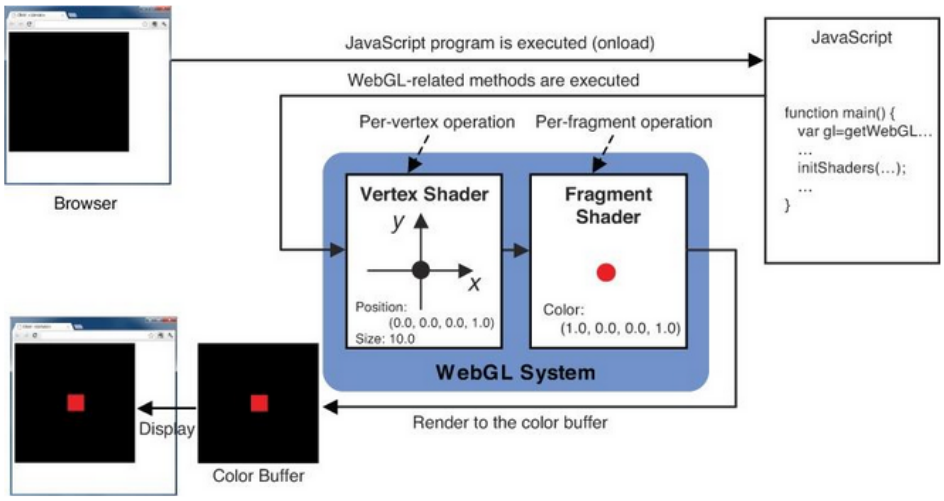
Drawing a Point in WebGL
File: 01-03-webgl.html

Drawing something on the screen in WebGL is different from Canvas2D of HTML5. Shader programs are necessary draw objects in WebGL.

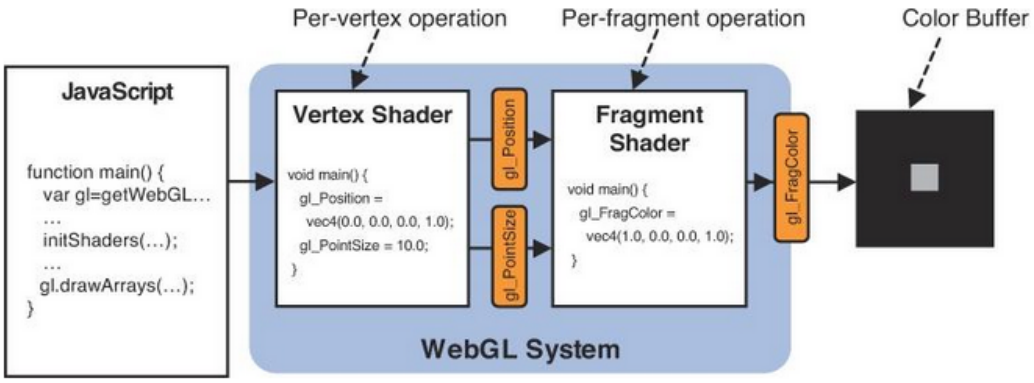
Shaders
Shaders are computer programs the is used to do shading (color processing, special effects). Shaders are programs that run mostly on your graphics processing unit. Shaders in WebGL use the OpenGL Shading Language for Embedded Systems (GLSL ES) syntax *(C-like syntax with built in variables and functions)*

There are two main shader programs in WebGL that we need to create to draw something on our canvas. These are:

1. Vertex Shader
    - Shader program that describe the traits of a vertex (position, colors, etc.)
2. Fragment Shader
    - Shader program that deals with per-fragment (“pixel” but not exact) processing such as lighting, post-processing



CMSC 161
Interactive Computer Graphics
Meeting 01 - Introduction to WebGL



Passing values to shaders

File: 01-04-webgl.html

There are two ways to pass data from the JavaScript to the vertex shader:

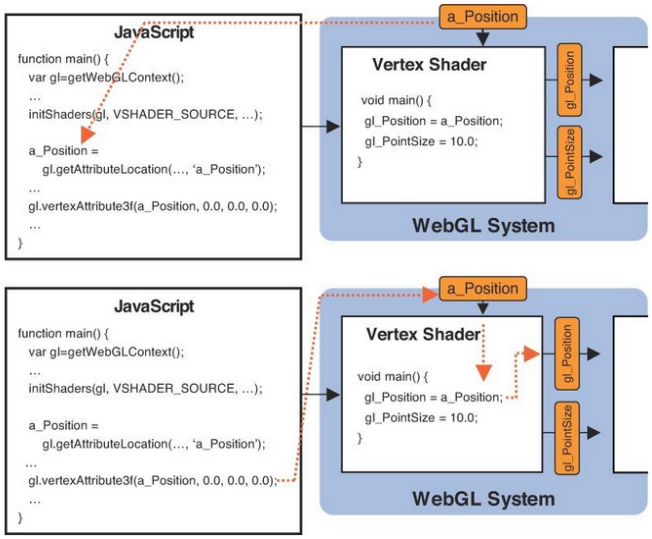
1. attribute variable
    - data that differs for each vertex
2. uniform variable
    - data that is the same (uniform) in each vertex

Declaring Special Variables in GLSL

```
<storage qualifier> <type> <variable name>;
attribute vec4 aPosition;
```

Assigning a value to an Attribute Variable

1. Get the storage location of the attribute variable using getAttribLocation()
2. Assign a value to the attribute variable using the location retrieved in first step



Passing values to shaders (fragment shader)

File: 01-05-webgl.html

We can also pass values to shaders using uniform storage qualifier variables. Uniform variables are data that supposed to be same in vertex.

Exercise

Create a webgl program that does the following:

1. Draws 10 random points in the canvas.
2. Points are randomly positioned in the canvas every time the page is refreshed.
3. Points have a color depending where on the screen they are positioned