Graphics Programming

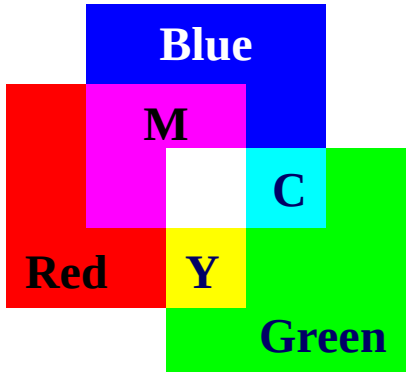
3RD WEEK, 2021

Attributes

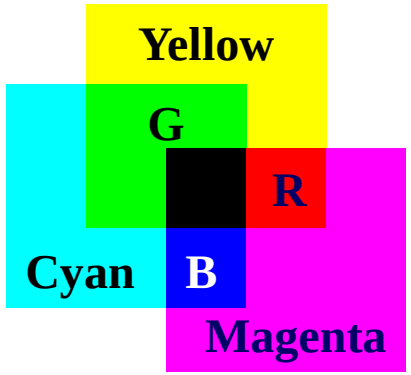
- Properties that determines how to render a geometric primitive (appearance of objects)
  - Color (points, lines, polygons)
  - Size and width (points, lines)
  - Stipple pattern (lines, polygons)
  - Polygon mode
    - Display as filled: solid color or stipple pattern
    - Display edges
    - Display vertices
- Only a few (`gl_PointSize`) are supported by OpenGL functions

Color

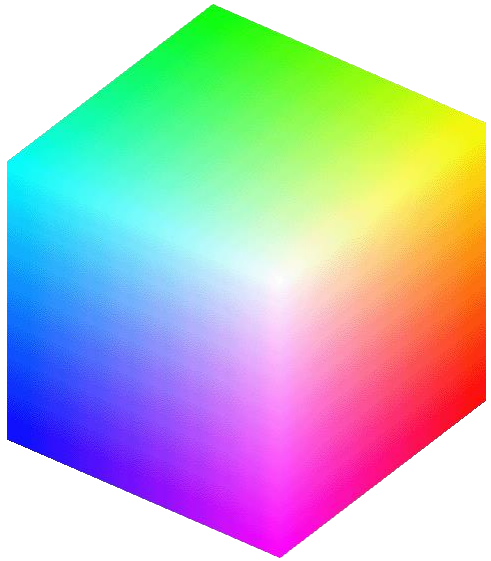
- Three color theory
  - Our brains do not receive the entire color distribution but rather than three values
- <u>Additive</u> color – ex) CRT
- <u>Subtractive</u> color – ex) printing



Additive Color



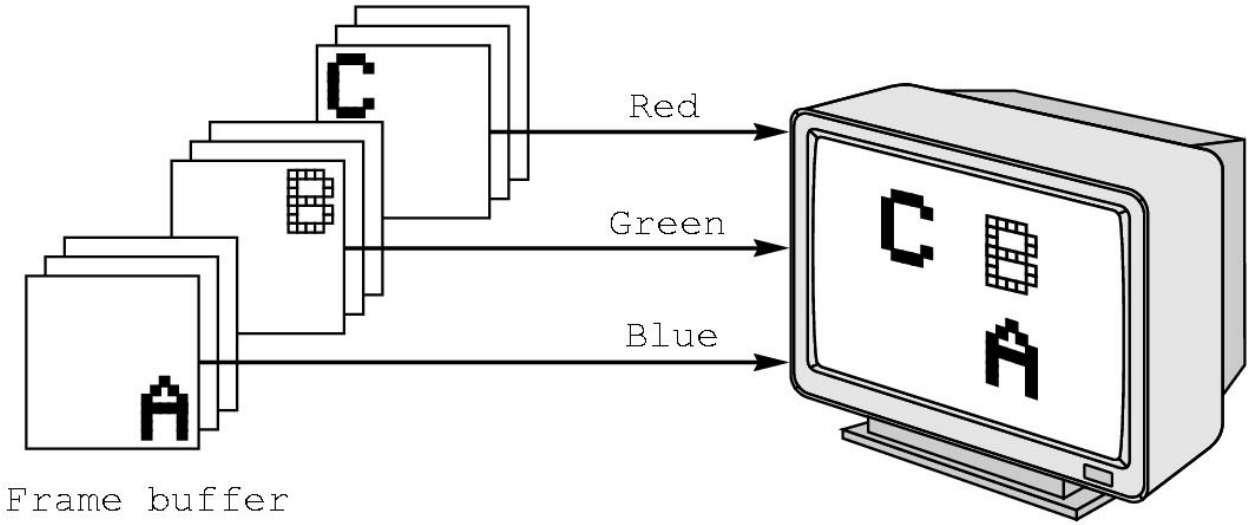
Subtractive Color



Color Solid

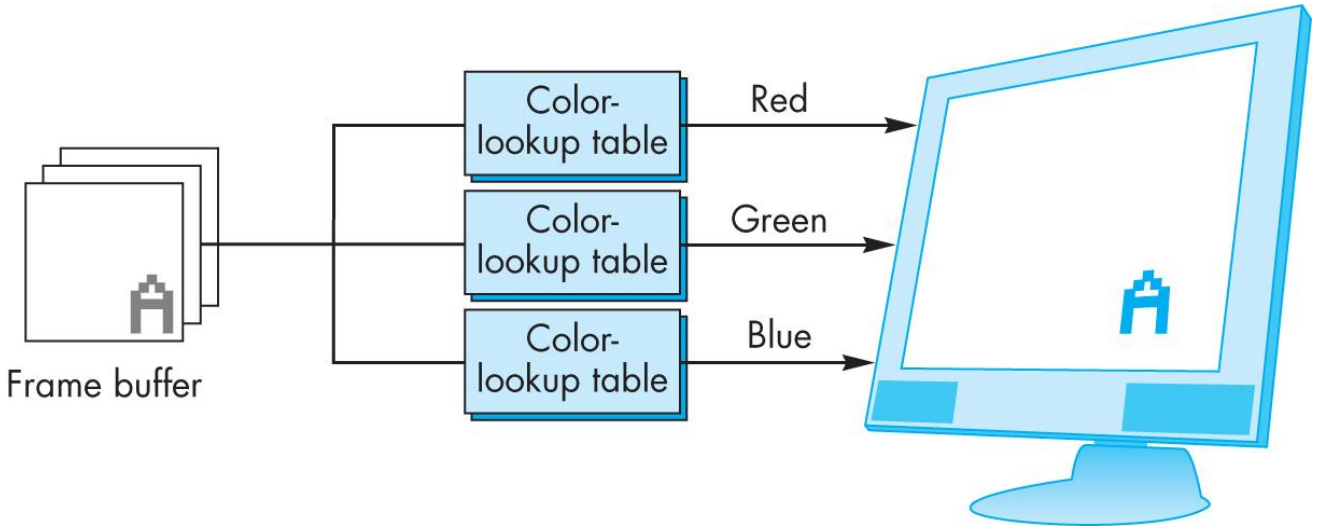
RGB Color

- Each color component is stored separately in the frame buffer
  - Usually 8 bits per component in buffer
  - Color values can range from 0.0 (none) to 1.0 (all) using floats or over the range from 0 to 255 using unsigned bytes



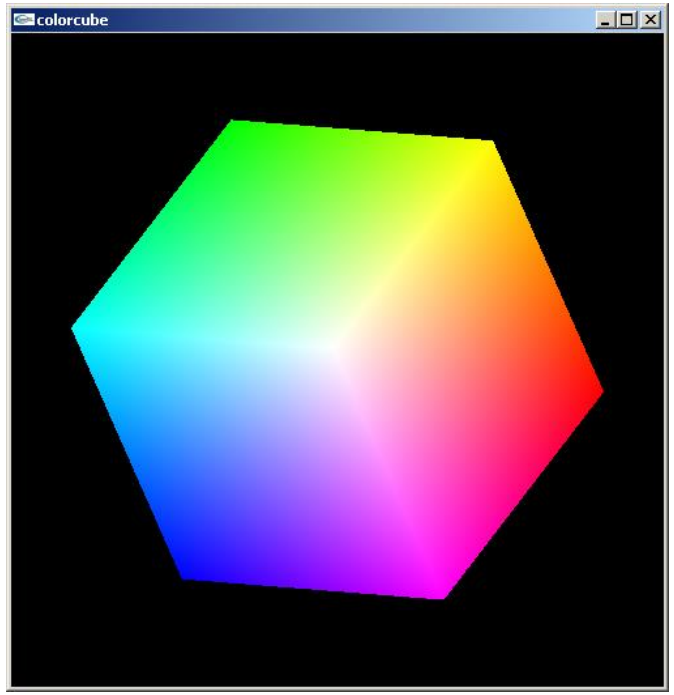
Indexed Color

- Colors are indices into tables of RGB values
- Requires less memory
  - indices usually 8 bits
  - not as important now
    - Memory inexpensive
    - Need more colors for shading



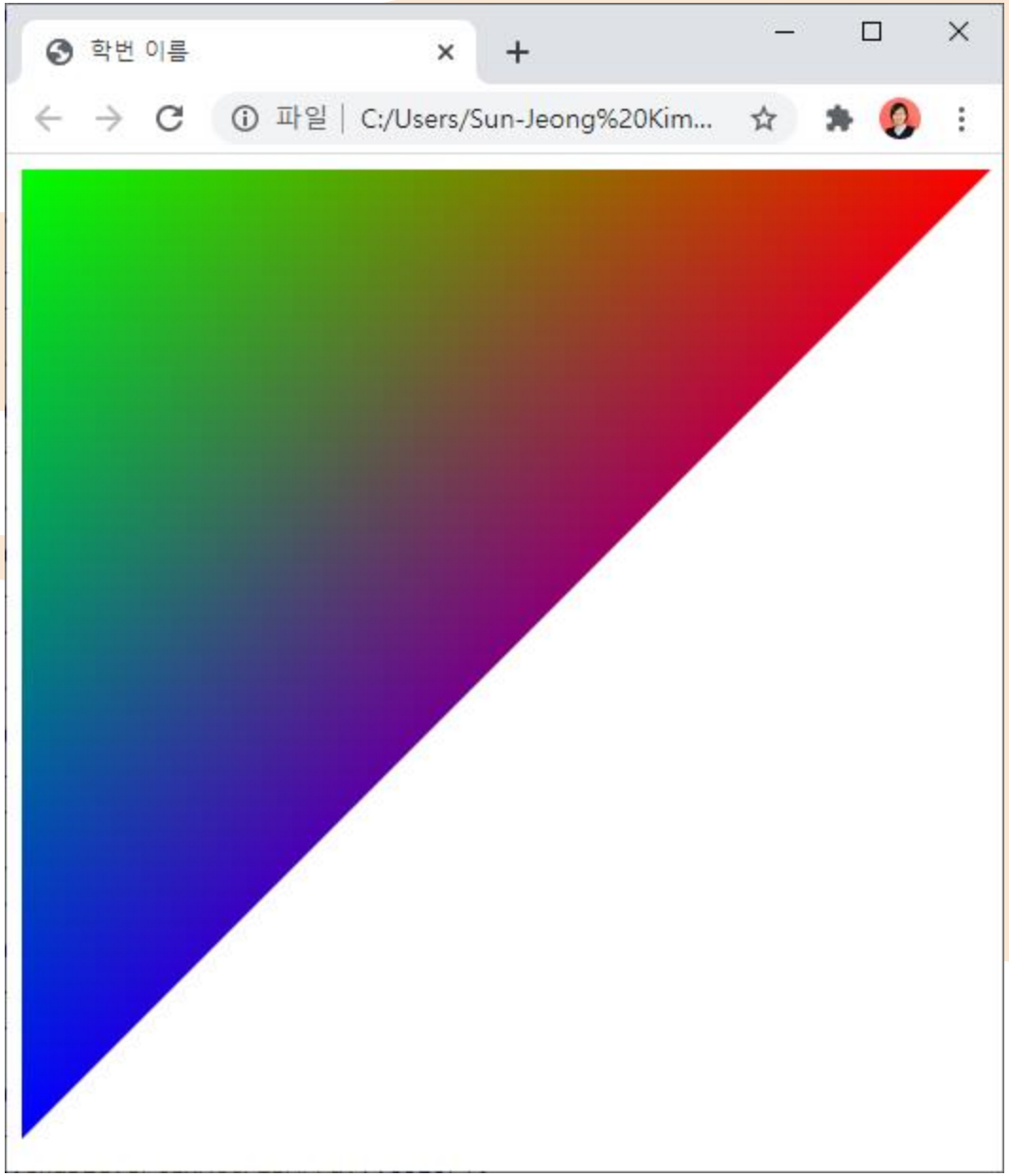
Smooth Color

- Default is smooth shading
  - OpenGL interpolates vertex colors across visible polygons
- Alternative is flat shading
  - Color of first vertex determines fill color
  - Handle in shader



Setting Colors

- Colors are ultimately set in the fragment shader but can be determined in either shader or in the application
- Application color: pass to vertex shader as a uniform variable or as a vertex attribute
- Vertex shader color: pass to fragment shader as varying variable
- Fragment color: can alter via shader code


학번 이름
파일 | C:/Users/Sun-Jeong%20Kim...

File Edit Selection View Go Run Terminal Help colorTriangle.html - Visual Studio Code

colorTriangle.html × | colorTriangle.js

C: > Users > Sun-Jeong Kim > Desktop > CG > colorTriangle.html > html > head > script

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        varying vec4 color;

        void main() {
            gl_Position = vPosition;
            color = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 color;

        void main() {
            gl_FragColor = color;
        }
        </script>

        <script type="text/javascript" src="Common/webgl-utils.js"></script>
        <script type="text/javascript" src="Common/initShaders.js"></script>
        <script type="text/javascript" src="Common/MV.js"></script>
        <script type="text/javascript" src="colorTriangle.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
    </body>
</html>
```

⊗ 0 ⚠ 0 Ln 28, Col 61 Spaces: 4 UTF-8 CRLF HTML

File Edit Selection View Go Run Terminal Help colorTriangle.js - Visual Studio Code

colorTriangle.html | colorTriangle.js

C: > Users > Sun-Jeong Kim > Desktop > CG > colorTriangle.js > init

```javascript
var gl;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    var vertices = [
        vec2(-1, -1),
        vec2(-1, 1),
        vec2(1, 1)
    ];

    var colors = [
        vec4(0, 0, 1, 1),
        vec4(0, 1, 0, 1),
        vec4(1, 0, 0, 1)
    ]

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(1.0, 1.0, 1.0, 1.0);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
```

⊗ 0 ⚠ 0 Ln 50, Col 38 Spaces: 4 UTF-8 CRLF JavaScript

File Edit Selection View Go Run Terminal Help colorTriangle.js - Visual Studio Code

colorTriangle.html | colorTriangle.js

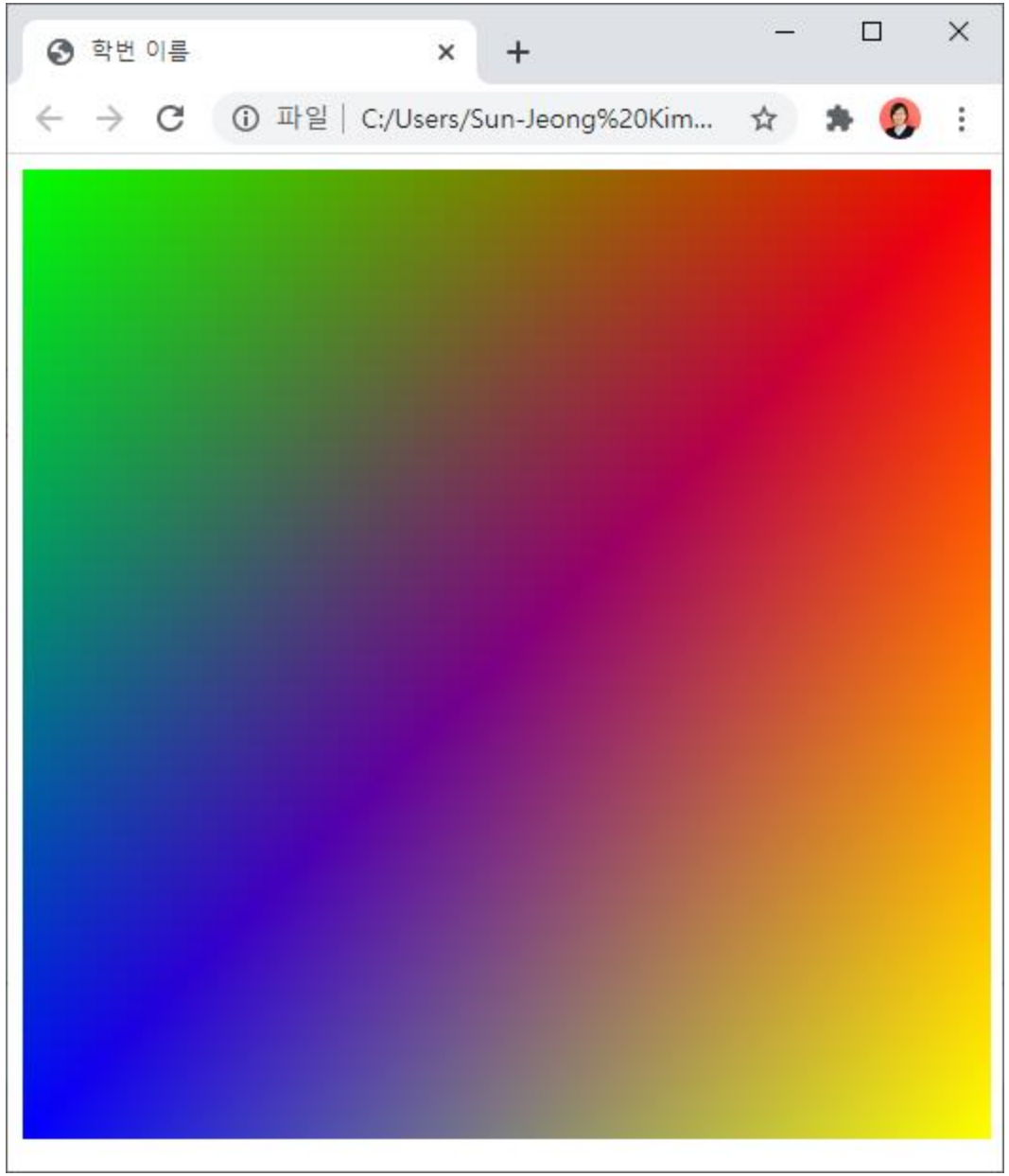
C: > Users > Sun-Jeong Kim > Desktop > CG > colorTriangle.js > init

```javascr
    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.TRIANGLES, 0, 3);
}
```

⊗ 0 ⚠ 0 Ln 50, Col 38 Spaces: 4 UTF-8 CRLF JavaScript

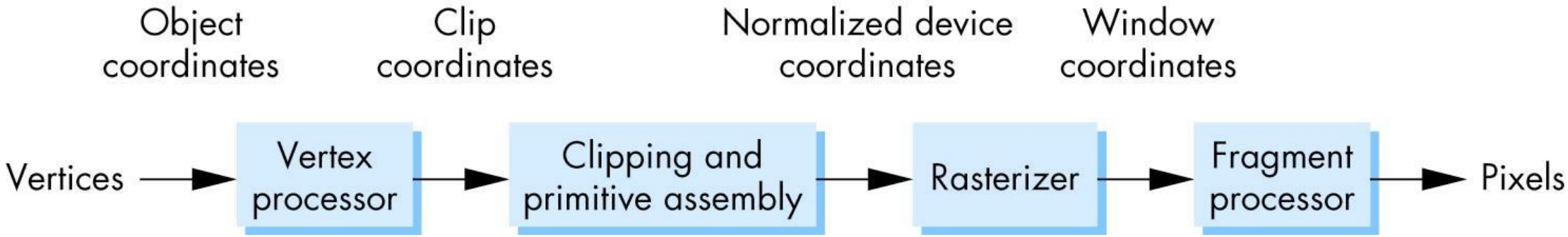
연습 문제 (1)

- 다음과 같이 그리시오.



Programmable Pipelines

- Two components
  - Vertex program (vertex shaders)
  - Fragment program (fragment shaders)
- In the pipeline architecture, the vertex processor and the fragment processor are programmable by application programs called shaders



Vertex Shader Applications

- Moving vertices
  - Morphing
  - Wave motion
  - Fractals
- Lighting
  - More realistic models
  - Cartoon shaders

Fragment Shader Applications (1)

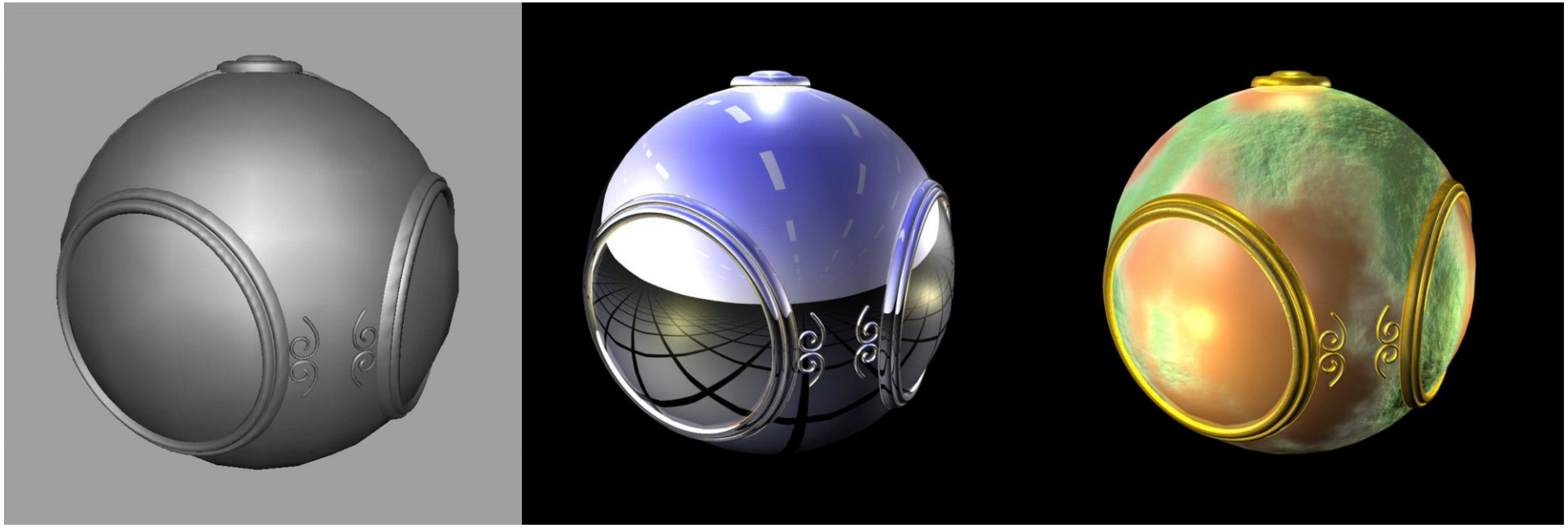
- Per fragment lighting calculations

Per vertex lighting

Per fragment lighting

Fragment Shader Applications (2)

- Texture mapping



Smooth shading | Environment mapping | Bump mapping

Simple Vertex Shader

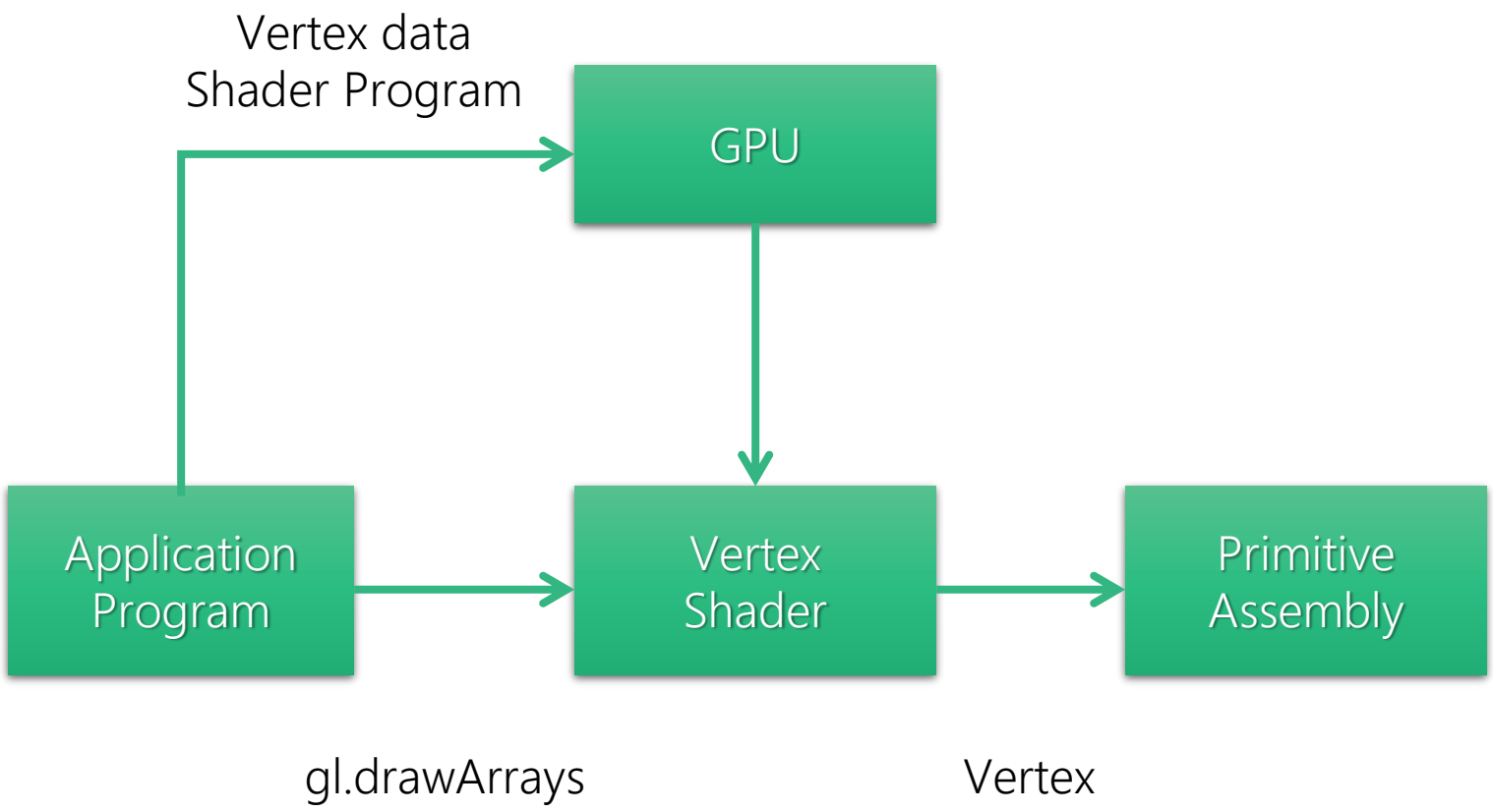
input from application

```
attribute vec4 vPosition;
void main(void)
{
    gl_Position = vPosition;
}
```

must link to variable in application

built in variable

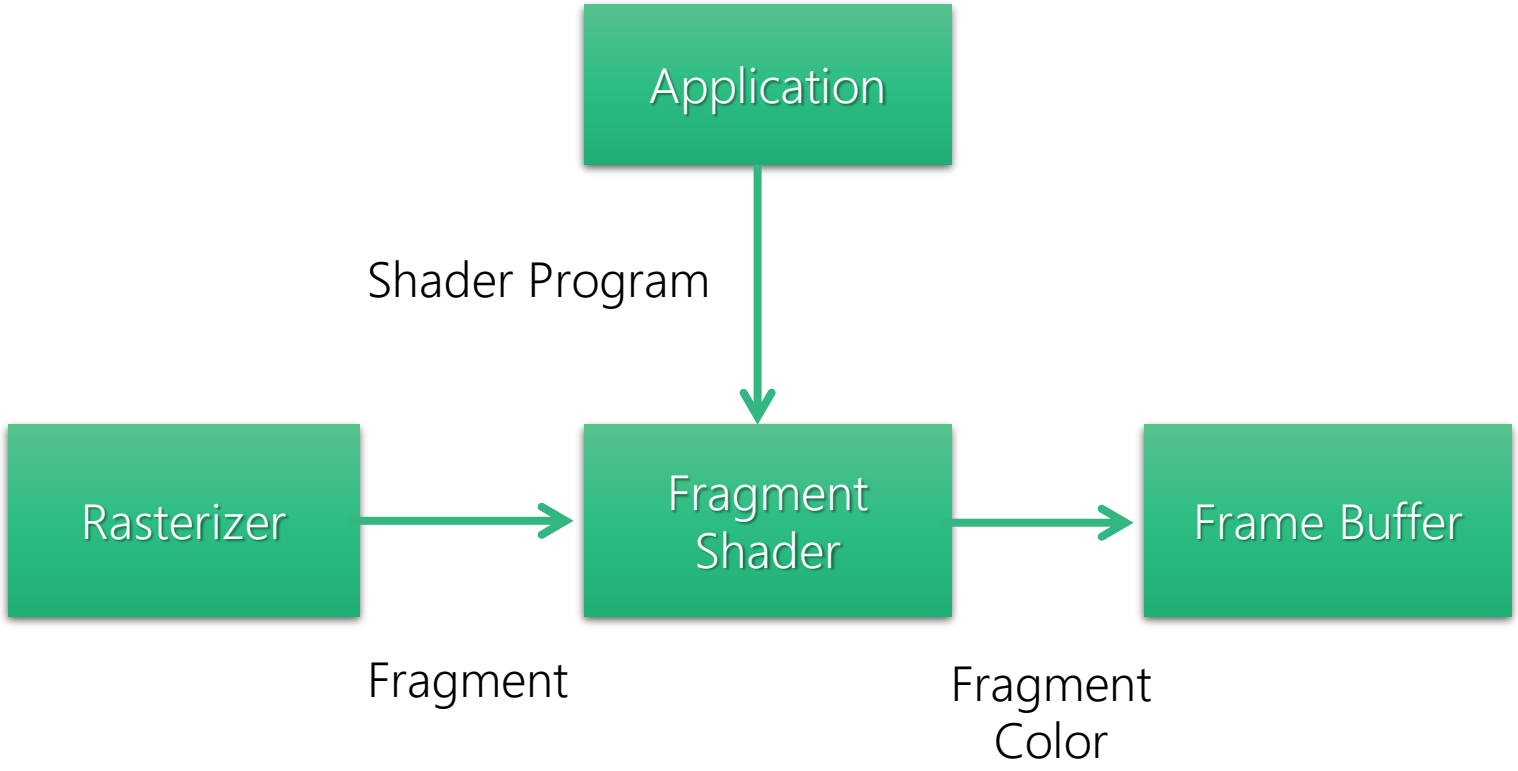
Execution Model – Vertex Shader



Simple Fragment Shader

```
precision mediump float;


void main(void)
{
  gl_FragColor = vec4( 1.0, 0.0, 0.0, 1.0 );
}
```

Execution Model – Fragment Shader



Data Type

- C types: `int`, `float`, `bool`
- Vectors:
  - float `vec2`, `vec3`, `vec4`
  - Also int (`ivec`) and boolean (`bvec`)
- Matrices: `mat2`, `mat3`, `mat4`
  - Stored by columns
  - Standard referencing `m[row][column]`
- C++ style constructors
  - `vec3 a = vec3(1.0, 2.0, 3.0);`
  - `vec2 b = vec2(a);`

Pointers

- There are no pointers in GLSL
- We can use C structs which can be copied back from functions
- Because matrices and vectors are basic types they can be passed into and output from GLSL functions
  - Ex) `mat3 func(mat3 a);`
- Variables passed by copying

Qualifiers

- GLSL has many of the same qualifiers such as `const` as C/C++
- Need others due to the nature of the shader architecture
- Variables can change
  - Once per primitive
  - Once per vertex
  - Once per fragment
  - At any time in the application
- Vertex attributes are <u>interpolated</u> by the rasterizer into fragment attributes

Attribute Qualifiers

- Attribute-qualified variables <u>can change at most once per vertex</u>
- There are a few built in variables such as `gl_Position` but most have been deprecated
- User defined (in application program)
  - `attribute float temperature;`
  - `attribute vec3 velocity;`
  - Recent versions of GLSL use `in` and `out` qualifiers to get to and from shaders

Uniform Qualifiers

- Variables that are constant for an entire primitive
- Can be changed in application and sent to shaders
- <u>Cannot be changed in shader</u>
- Used to pass information to shader such as the bounding box of a primitive

Varying Qualifiers

- Variables that are passed from vertex shader to fragment shader
- Automatically interpolated by the rasterizer
- With WebGL, GLSL uses the varying qualifier in both shaders
  - `varying vec4 color;`
- More recent versions of WebGL use `out` in vertex shader and `in` in the fragment shader
  - `out vec4 color;  // vertex shader`
  - `in vec4 color;  // fragment shader`

Example: Vertex Shader

```
attribute vec4 vPosition;
attribute vec4 vColor;
varying vec4 fColor;
void main(void)
{
    gl_Position = vPosition;
    fColor = vColor;
}
```

Corresponding Fragment Shader

```
precision mediump float;

varying vec4 fColor;
void main(void)
{
  gl_FragColor = fColor;
}
```

Sending Colors from Application

```
var cBuffer = gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER, cBuffer);
gl.bufferData(gl.ARRAY_BUFFER, flatten(colors),
gl.STATIC_DRAW);

var vColor = gl.getAttribLocation(program, "vColor");
gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
gl.enableVertexAttribArray(vColor);
```

Sending a Uniform Variable

```
// in application
vec4 color = vec4(1.0, 0.0, 0.0, 1.0);
colorLoc = gl.getUniformLocation(program, ”uColor");
gl.uniform4f(colorLoc, color);

// in fragment shader (similar in vertex shader)
uniform vec4 uColor;
void main(){
    gl_FragColor = uColor;
}
```

Operators and Functions

- Standard C functions
  - Trigonometric
  - Arithmetic
  - Normalize, reflect, length

- Overloading of vector and matrix types

```
mat4 a;
vec4 b, c, d;
c = b*a; // a column vector stored as a 1d array
d = a*b; // a row vector stored as a 1d array
```

Swizzling and Selection

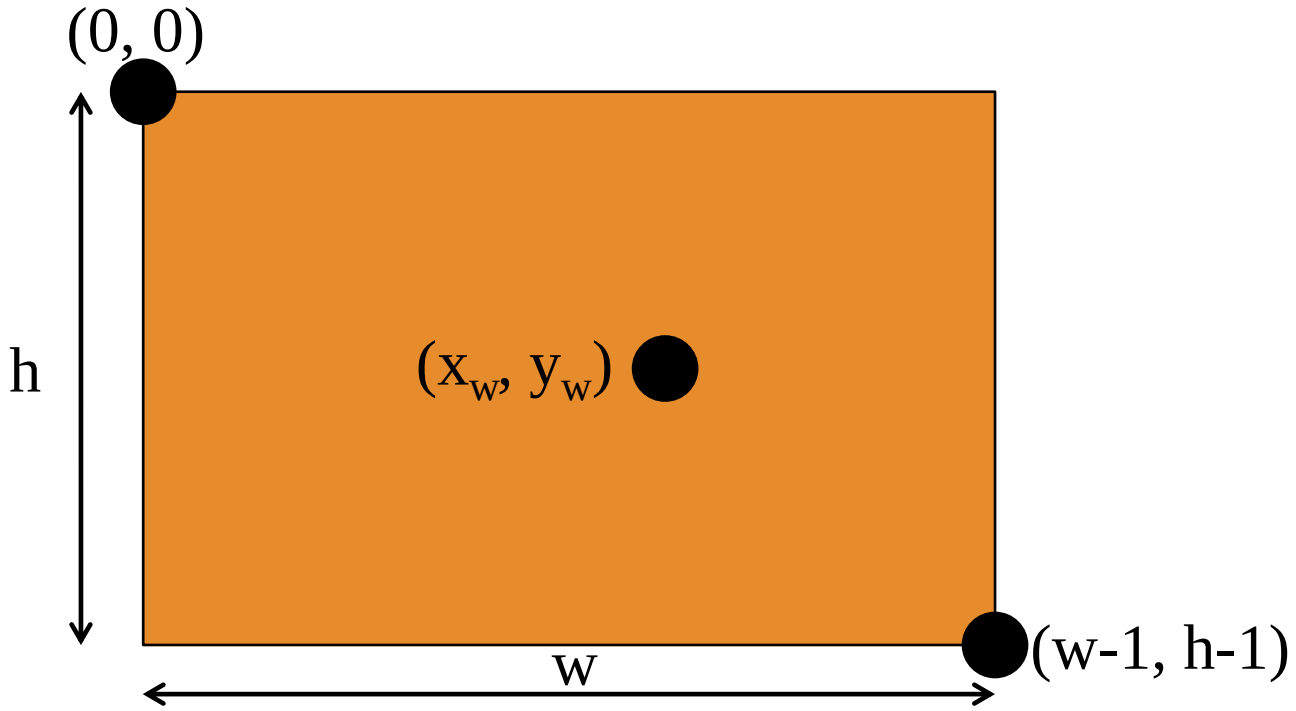
- Can refer to array elements by element using [ ] or selection (.) operator with
  - `x`, `y`, `z`, `w`
  - `r`, `g`, `b`, `a`
  - `s`, `t`, `p`, `q`
  - Ex) `a[2]`, `a.b`, `a.z`, `a.p` are the same

- Swizzling operator lets us manipulate components

```
vec4 a = vec4(1.0, 2.0, 3.0, 4.0);
a.yz = vec2(1.0, 2.0); // a = (1.0, 1.0, 2.0, 4.0)
```

Position Input

- Returning position from click event
  - Canvas specified in HTML file of size canvas.width x canvas.height
  - Returned window coordinates are event.clientX and event.clientY



$$(0,h) \rightarrow (-1,-1)$$

$$(w,0) \rightarrow (1,1)$$

$$x = -1 + \frac{2 * x_w}{w}$$

$$y = -1 + \frac{2 * (h - y_w)}{h}$$

File Edit Selection View Go Run Terminal Help drawPoints.html - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.html > html > head > script#fragment-shader

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;

        void main() {
            gl_PointSize = 5.0;
            gl_Position = vPosition;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        uniform vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="Common/webgl-utils.js"></script>
        <script type="text/javascript" src="Common/initShaders.js"></script>
        <script type="text/javascript" src="Common/MV.js"></script>
        <script type="text/javascript" src="drawPoints.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
    </body>
</html>
```

⊗ 0 ⚠ 0 Ln 19, Col 35 Spaces: 4 UTF-8 CRLF HTML

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init

```javascript
var gl;
var points;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    points = [];
    var redraw = false;

    canvas.addEventListener("mousedown", function(event) {
        redraw = true;
    });

    canvas.addEventListener("mouseup", function(event) {
        redraw = false;
    });

    canvas.addEventListener("mousemove", function(event) {
        if (redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.push(p);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

            render();
        }
    });

    // Configure WebGL
```

⊗ 0 ⚠ 0 Ln 54, Col 44 Spaces: 4 UTF-8 CRLF JavaScript

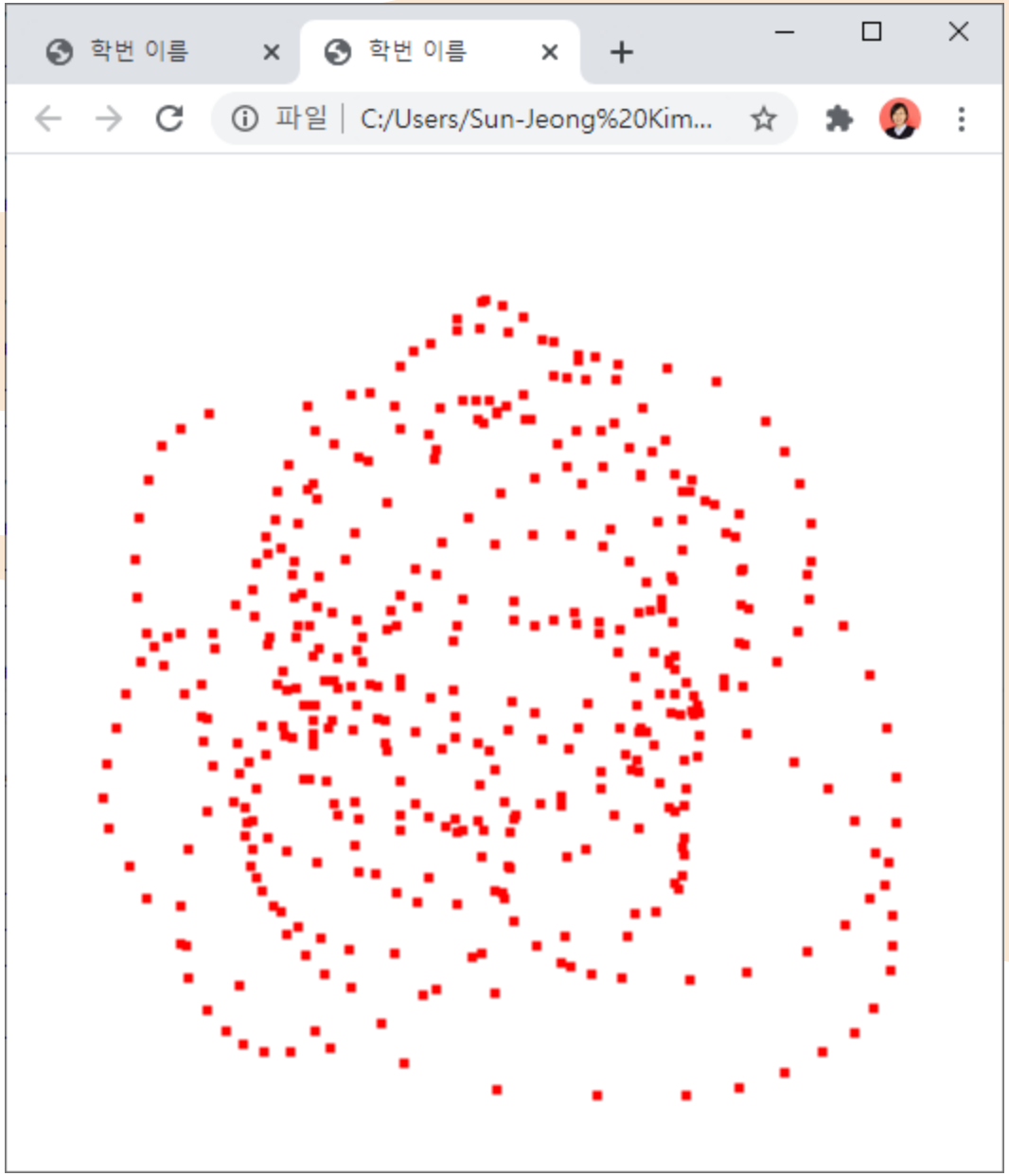
File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init

```javascript

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(1.0, 1.0, 1.0, 1.0);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    var fColor = gl.getUniformLocation(program, "fColor");
    gl.uniform4f(fColor, 1.0, 0.0, 0.0, 1.0);

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.POINTS, 0, points.length);
}

```

⊗ 0 ⚠ 0 Ln 54, Col 44 Spaces: 4 UTF-8 CRLF JavaScript


학번 이름
학번 이름
파일 | C:/Users/Sun-Jeong%20Kim...

연습 문제 (2)

- 웹페이지에 슬라이더를 추가하여, 점의 크기를 입력 받아 그리시오.

File Edit Selection View Go Run Terminal Help drawPoints.html - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.html > html > body > p

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        uniform float pointSize;

        void main() {
            gl_PointSize = pointSize;
            gl_Position = vPosition;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        uniform vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="Common/webgl-utils.js"></script>
        <script type="text/javascript" src="Common/initShaders.js"></script>
        <script type="text/javascript" src="Common/MV.js"></script>
        <script type="text/javascript" src="drawPoints.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
        <p>Point Size 1<input type="range" id="pointSize" min="1" max="10" step="1" value="5">10</p>
    </body>
</html>
```

⊗ 0 ⚠ 0 Ln 33, Col 101 Spaces: 4 UTF-8 CRLF HTML

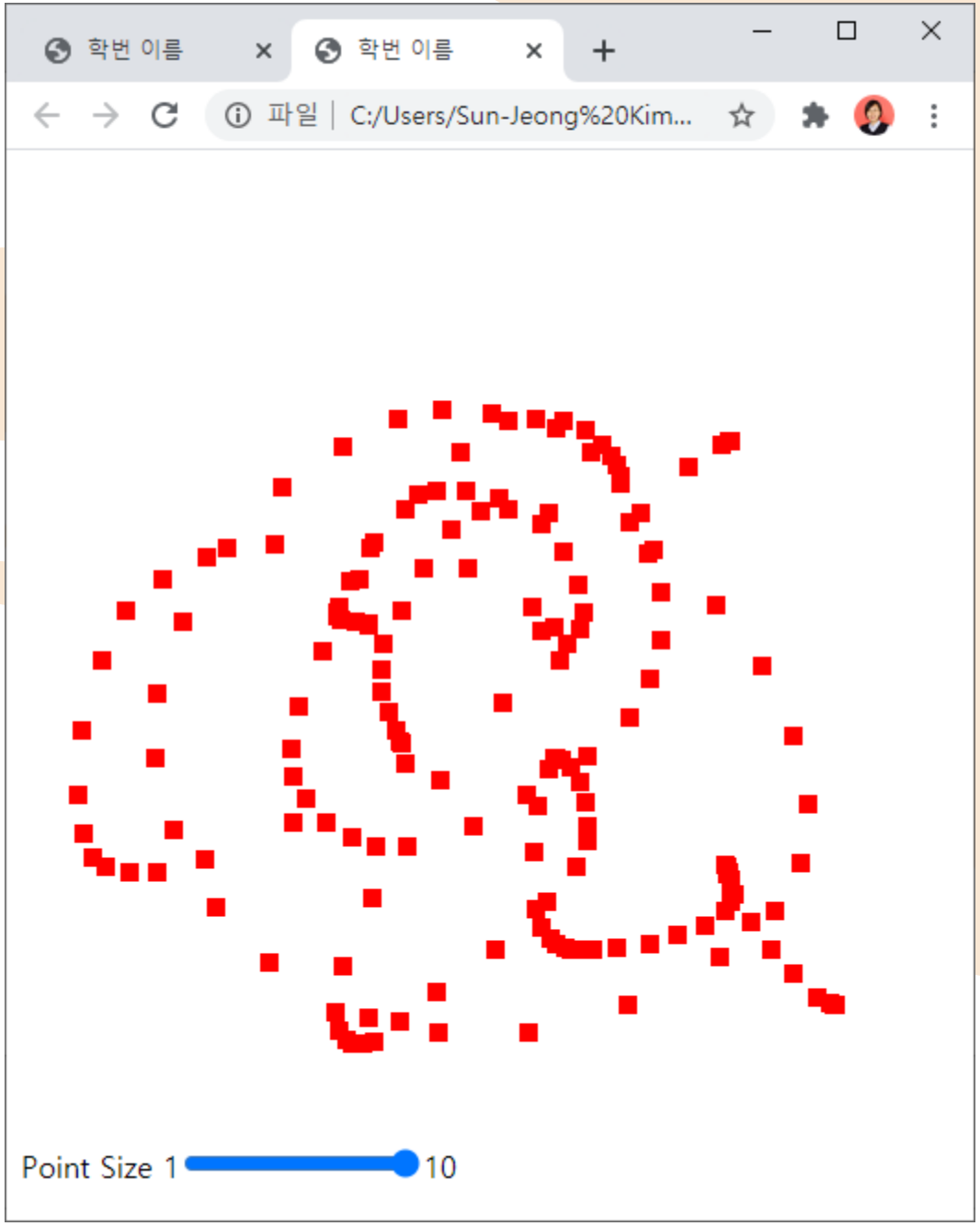
File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init > onchange

```javascript

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    var fColor = gl.getUniformLocation(program, "fColor");
    gl.uniform4f(fColor, 1.0, 0.0, 0.0, 1.0);

    var locPointSize = gl.getUniformLocation(program, "pointSize");
    gl.uniform1f(locPointSize, 5.0);

    document.getElementById("pointSize").onchange = function () {
        var size = this.value;
        gl.uniform1f(locPointSize, size);

        render();
    }

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.POINTS, 0, points.length);
}

```

⊗ 0 ⚠ 0 Ln 63, Col 18 Spaces: 4 UTF-8 CRLF JavaScript


학번 이름
학번 이름
파일 | C:/Users/Sun-Jeong%20Kim...
Point Size 1
10

연습 문제 (3)

- 점마다 색상을 다르게 채색하시오.

File Edit Selection View Go Run Terminal Help drawPoints.html - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.html > html > head > script#fragment-shader

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform float pointSize;
        varying vec4 fColor;

        void main() {
            gl_PointSize = pointSize;
            gl_Position = vPosition;
            fColor = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="Common/webgl-utils.js"></script>
        <script type="text/javascript" src="Common/initShaders.js"></script>
        <script type="text/javascript" src="Common/MV.js"></script>
        <script type="text/javascript" src="drawPoints.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
```

⊗ 0 ⚠ 0 Ln 20, Col 28 Spaces: 4 UTF-8 CRLF HTML

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init

```javascript
var gl;
var points, colors;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    points = [];
    colors = [];
    var redraw = false;

    canvas.addEventListener("mousedown", function(event) {
        redraw = true;
    });

    canvas.addEventListener("mouseup", function(event) {
        redraw = false;
    });

    canvas.addEventListener("mousemove", function(event) {
        if (redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.push(p);
            gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

            var c = vec4(Math.random(), Math.random(), Math.random(), 1.0);
            colors.push(c);
            gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
```

⊗ 0 ⚠ 0 Ln 71, Col 7 Spaces: 4 UTF-8 CRLF JavaScript

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init

```javascript
    canvas.addEventListener("mousemove", function(event) {
        if (redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.push(p);
            gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

            var c = vec4(Math.random(), Math.random(), Math.random(), 1.0);
            colors.push(c);
            gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

            render();
        }
    });

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(1.0, 1.0, 1.0, 1.0);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);
```

⊗ 0 ⚠ 0 Ln 71, Col 7 Spaces: 4 UTF-8 CRLF JavaScript

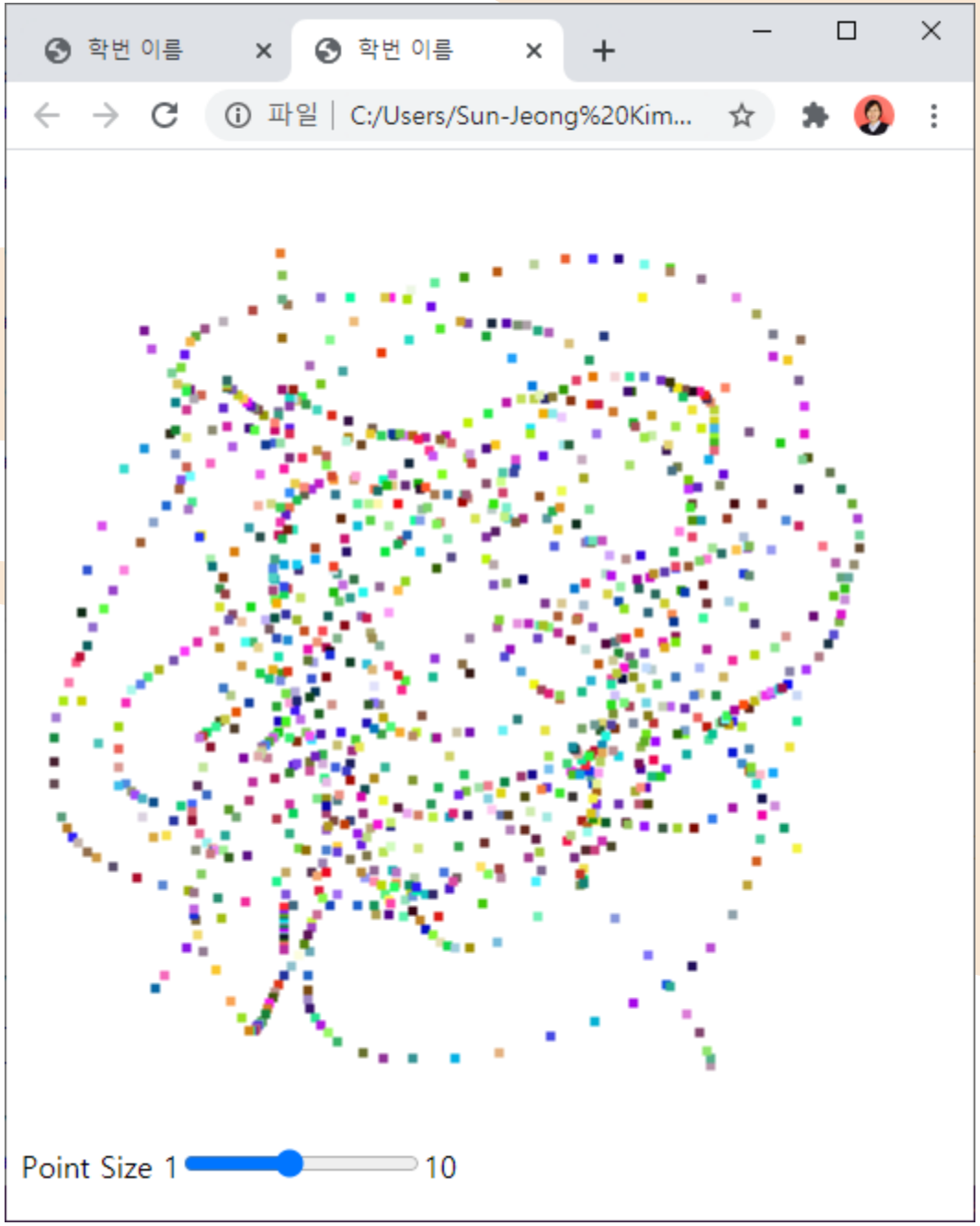
File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > init

```
    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    //var fColor = gl.getUniformLocation(program, "fColor");
    //gl.uniform4f(fColor, 1.0, 0.0, 0.0, 1.0);

    var locPointSize = gl.getUniformLocation(program, "pointSize");
    gl.uniform1f(locPointSize, 5.0);

    document.getElementById("pointSize").onchange = function () {
        var size = this.value;
        gl.uniform1f(locPointSize, size);

        render();
    }

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.POINTS, 0, points.length);
}
```

⊗ 0 ⚠ 0 Ln 71, Col 7 Spaces: 4 UTF-8 CRLF JavaScript


학번 이름
학번 이름
파일 | C:/Users/Sun-Jeong%20Kim...
Point Size 1
10

Linking Shader with Application

- Read shaders
- Compile shaders
- Create a program object
- Link everything together
- Link variables in application with variables in shaders
  - Vertex attributes
  - Uniform variables

Program Object

- Container for shaders
  - Can contain multiple shaders
  - Other GLSL functions

```
var program = gl.createProgram();

/* define shader objects here */

gl.attachShader( program, vertShdr );
gl.attachShader( program, fragShdr );
gl.linkProgram( program );
```

Reading a Shader

- Shaders are added to the program object and compiled
- Usual method of passing a shader is as a null-terminated string using the function `glShaderSource( shdr, text );`
- If the shader is in HTML file, we can get it into application by `getElementById` method
- If the shader is in a file, we can write a reader to convert the file to a string

Adding a Vertex Shader

```
var vertShdr;
var vertElem = document.getElementById(vertexShaderId);

vertShdr = gl.createShader( gl.VERTEX_SHADER );

gl.shaderSource( vertShdr, vertElem.text );
gl.compileShader( vertShdr );

// after program object created
gl.attachShader( program, vertShdr );
```

Shader Reader

- Following code may be a security issue with some browsers if you try to run it locally
  - Cross origin request

```
function getShader( gl, shaderName, type ) {
    var shader = gl.createShader( type );
    shaderScript = loadFileAJAX( shaderName );
    if( !shaderScript ) {
        alert( "Could not find shader source:"
               + shaderName );
    }
}
```

Precision Declaration

- In GLSL for WebGL we must specify desired precision in fragment shaders
  - Artifact inherited from OpenGL ES
  - ES must run on very simple embedded devices that may not support 32-bit floating point
  - All implementations must support mediump
  - No default for float in fragment shader
- Can use preprocessor directives (#ifdef) to check if highp supported and, if not, default to mediump

Pass Through Fragment Shader

```
#ifdef GL_FRAGMENT_SHADER_PRECISION_HIGH
precision highp float;
#else
precision mediump float;
#endif

varying vec4 fColor;
void main(void) {
    gl_FragColor = fColor;
}
```

연습 문제 (4)

- Drag하여 선분들을 그리시오.

File Edit Selection View Go Run Terminal Help drawPoints.html - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.html > html > body > div

```html
        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="Common/webgl-utils.js"></script>
        <script type="text/javascript" src="Common/initShaders.js"></script>
        <script type="text/javascript" src="Common/MV.js"></script>
        <script type="text/javascript" src="drawPoints.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
        <p>Point Size 1<input type="range" id="pointSize" min="1" max="10" step="1" value="5">10</p>
        <div>
            Choose color:
            <select id = "colors">
                <option value = "0">Black</option>
                <option value = "1">Red</option>
                <option value = "2">Yellow</option>
                <option value = "3">Green</option>
                <option value = "4">Blue</option>
                <option value = "5">Magenta</option>
                <option value = "6">Cyan</option>
            </select>
        </div>
    </body>
</html>
```

⊗ 0 ⚠ 0 Ln 38, Col 27 Spaces: 4 UTF-8 CRLF HTML

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > render

```javascript
var gl;
var points, colors;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    points = [];
    colors = [];
    var redraw = false;
    var colorArray = [
        vec4(0.0, 0.0, 0.0, 1.0),  // black
        vec4(1.0, 0.0, 0.0, 1.0),  // red
        vec4(1.0, 1.0, 0.0, 1.0),  // yellow
        vec4(0.0, 1.0, 0.0, 1.0),  // green
        vec4(0.0, 0.0, 1.0, 1.0),  // blue
        vec4(1.0, 0.0, 1.0, 1.0),  // magenta
        vec4(0.0, 1.0, 1.0, 1.0)   // cyan
    ];
    var cIndex = 0;

    document.getElementById("colors").onclick = function (event) {
        cIndex = event.target.value;
    }

    canvas.addEventListener("mousedown", function(event) {
        if (!redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.push(p);
```

⊗ 0 ⚠ 0 Ln 112, Col 27 Spaces: 4 UTF-8 CRLF JavaScript

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > render

```javascript
    canvas.addEventListener("mousedown", function(event) {
        if (!redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.push(p);
            points.push(p);
            gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

            var c = colorArray[cIndex];
            colors.push(c);
            colors.push(c);
            gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
        }
        redraw = true;
    });

    canvas.addEventListener("mouseup", function(event) {
        redraw = false;
    });

    canvas.addEventListener("mousemove", function(event) {
        if (redraw) {
            var p = vec2(2*event.clientX/canvas.width - 1,
                2*(canvas.height-event.clientY)/canvas.height -1);
            points.pop();
            points.push(p);
            gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

            render();
        }
    });
```

⊗ 0 ⚠ 0 Ln 112, Col 27 Spaces: 4 UTF-8 CRLF JavaScript

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > render

```javascript
    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(0.9, 0.9, 0.9, 1.0);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    //var fColor = gl.getUniformLocation(program, "fColor");
    //gl.uniform4f(fColor, 1.0, 0.0, 0.0, 1.0);

    var locPointSize = gl.getUniformLocation(program, "pointSize");
    gl.uniform1f(locPointSize, 5.0);

```

⊗ 0 ⚠ 0 Ln 112, Col 27 Spaces: 4 UTF-8 CRLF JavaScript

File Edit Selection View Go Run Terminal Help drawPoints.js - Visual Studio Code

colorTriangle.html | colorTriangle.js | drawPoints.html | drawPoints.js

C: > Users > Sun-Jeong Kim > Desktop > CG > drawPoints.js > render

```
    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    //var fColor = gl.getUniformLocation(program, "fColor");
    //gl.uniform4f(fColor, 1.0, 0.0, 0.0, 1.0);

    var locPointSize = gl.getUniformLocation(program, "pointSize");
    gl.uniform1f(locPointSize, 5.0);

    document.getElementById("pointSize").onchange = function () {
        var size = this.value;
        gl.uniform1f(locPointSize, size);

        render();
    }

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.LINES, 0, points.length);
}
```

⊗ 0 ⚠ 0 Ln 112, Col 27 Spaces: 4 UTF-8 CRLF JavaScript


학번 이름
파일 | C:/Users/Sun-Jeong%20Kim/...
Point Size 1
10
Choose color:
Black

연습 문제 (5)

- 세 점을 클릭하여 삼각형들을 그리시오.

- [선택 사항] - 가산점
  - Red / Green / Blue를 슬라이더 바 (Range)로 삼각형의 색상을 입력 받으시오.