Nama : Muhammad Dehan Al Kautsar
NIM : 13519200
Hands-On IF3260 Grafika Komputer - How It Works

Pada tutorial yang terdapat di laman web https://webglfundamentals.org/webgl/lessons/webgl-how-it-works.html, laman tersebut menjelaskan bagaimana sebuah bentuk (contoh di dalam laman tersebut adalah segitiga dan segiempat) dapat digambarkan *vertices* dan *coloring*-nya ke dalam sebuah canvas yang berada di dalam browser. Pada tutorial ini diperkenalkan yang namanya `gl_Position` dan `gl_FragColor` untuk menggambarkan bentuk yang kita inginkan menggunakan WebGL.

Pada laporan kali ini, saya akan melakukan *highlight* apa yang menjadi hal penting yang ditampilkan pada tutorial pada laman tersebut. Hal yang saya dapatkan akan dibahas di bawah paragraf ini.

```
<!-- vertex shader -->
<script  id="vertex-shader-2d" type="x-shader/x-vertex">
attribute vec2 a_position;
attribute vec4 a_color;

uniform mat3 u_matrix;

varying vec4 v_color;

void main() {
// Multiply the position by the matrix.
gl_Position = vec4((u_matrix * vec3(a_position, 1)).xy, 0, 1);

// Copy the color from the attribute to the varying.
v_color = a_color;
}
</script>
<!-- fragment shader -->
<script  id="fragment-shader-2d" type="x-shader/x-fragment">
precision mediump float;

varying vec4 v_color;

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

Pada HTML code di atas, dibuat vertex shader serta fragment shader yang berfungsi untuk menampilkan bentuk yang diinisialisasi di javascript file yang kita buat maupun import file. Vertex serta fragment shader apabila digabungkan menjadi shader program (atau yang biasa disebut dengan program saja).

```
// Create a buffer for the positions.
var positionBuffer = gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER, positionBuffer);
// Set Geometry.
setGeometry(gl);      You, 3 days ago • feat: how it works
```

Berikut adalah code untuk membentuk buffer position. Buffer position yang dibuat kemudian akan di-set geometrinya di dalam HTML canvas seperti pada gambar code di bawah ini:

```
function setGeometry(gl) {
  gl.bufferData(
      gl.ARRAY_BUFFER,
      new Float32Array([
          -150, -100,
          150, -100,
          -150,  100,
          150, -100,
          -150,  100,
          150,  100]),
      gl.STATIC_DRAW);
}
```

Kemudian untuk pewarnaannya, buffernya juga dibentuk melalui javascript code yang mirip dengan positionBuffer seperti di bawah ini:

```
// Create a buffer for the colors.
var colorBuffer = gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER, colorBuffer);
// Set the colors.
setColors(gl);
```

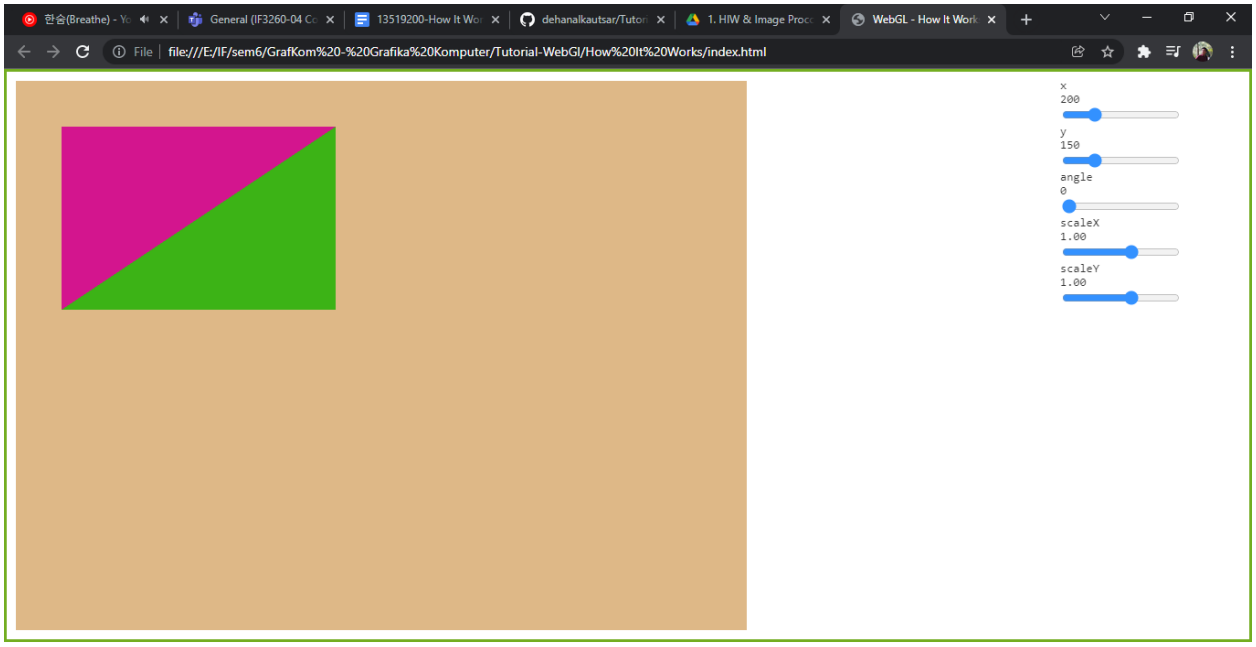
```
function setColors(gl) {
  // Pick 2 random colors.
  var r1 = Math.random() * 256; // 0 to 255.99999
  var b1 = Math.random() * 256; // these values
  var g1 = Math.random() * 256; // will be truncated
  var r2 = Math.random() * 256; // when stored in the
  var b2 = Math.random() * 256; // Uint8Array
  var g2 = Math.random() * 256;

  gl.bufferData(
      gl.ARRAY_BUFFER,
      new Uint8Array(    // Uint8Array
        [ r1, b1, g1, 255,
          r1, b1, g1, 255,
          r1, b1, g1, 255,
          r2, b2, g2, 255,
          r2, b2, g2, 255,
          r2, b2, g2, 255]),
      gl.STATIC_DRAW);
}

main();
```

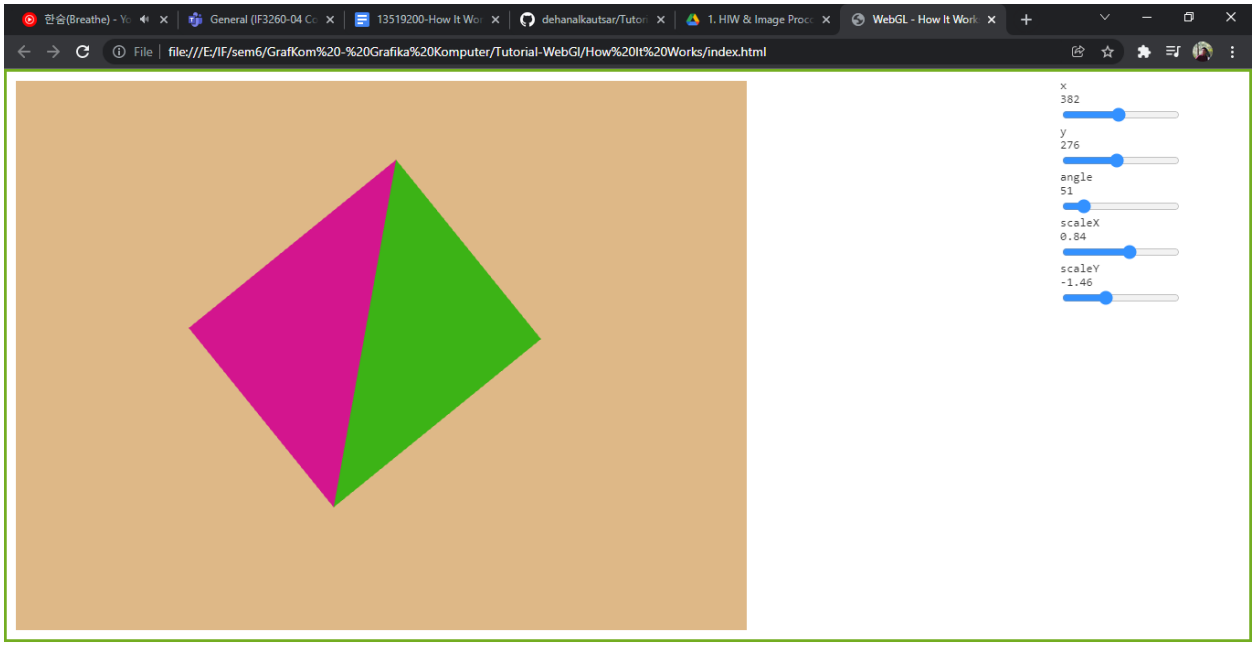
Kemudian, agar HTML dapat membaca apa yang telah dibuat pada javascript, diperlukan kode ini:

```
// Draw the scene.
function drawScene() {      You, 3 days ago • feat: how it works
  webglUtils.resizeCanvasToDisplaySize(gl.canvas);

  // Tell WebGL how to convert from clip space to pixels
  gl.viewport(0, 0, gl.canvas.width, gl.canvas.height);

  // Clear the canvas.
  gl.clear(gl.COLOR_BUFFER_BIT);

  // Tell it to use our program (pair of shaders)
  gl.useProgram(program);

  // Turn on the position attribute
  gl.enableVertexAttribArray(positionLocation);

  // Bind the position buffer.
  gl.bindBuffer(gl.ARRAY_BUFFER, positionBuffer);

  // Tell the position attribute how to get data out of positionBuffer (ARRAY_BUFFER)
  var size = 2;          // 2 components per iteration
  var type = gl.FLOAT;   // the data is 32bit floats
  var normalize = false; // don't normalize the data
  var stride = 0;        // 0 = move forward size * sizeof(type) each iteration to get the next position
  var offset = 0;        // start at the beginning of the buffer
  gl.vertexAttribPointer(
      positionLocation, size, type, normalize, stride, offset);

  // Turn on the color attribute
  gl.enableVertexAttribArray(colorLocation);

  // Bind the color buffer.
  gl.bindBuffer(gl.ARRAY_BUFFER, colorBuffer);

  // Tell the color attribute how to get data out of colorBuffer (ARRAY_BUFFER)
  var size = 4;                 // 4 components per iteration
  var type = gl.UNSIGNED_BYTE;  // the data is 8bit unsigned bytes
  var normalize = true;         // normalize the data
  var stride = 0;               // 0 = move forward size * sizeof(type) each iteration to get the next position
  var offset = 0;               // start at the beginning of the buffer
  gl.vertexAttribPointer(
      colorLocation, size, type, normalize, stride, offset);

  // Compute the matrix
  var matrix = m3.projection(gl.canvas.clientWidth, gl.canvas.clientHeight);
  matrix = m3.translate(matrix, translation[0], translation[1]);
  matrix = m3.rotate(matrix, angleInRadians);
  matrix = m3.scale(matrix, scale[0], scale[1]);

  // Set the matrix.
  gl.uniformMatrix3fv(matrixLocation, false, matrix);

  // Draw the geometry.
  var primitiveType = gl.TRIANGLES;
  var offset = 0;
  var count = 6;
  gl.drawArrays(primitiveType, offset, count);
}
```

Di dalam kode di atas terdapat beberapa fungsi yang berguna seperti resizeCanvasToDisplaySize(), enableVertexAttribArray(), bindBuffer(), dan lain-lainnya yang telah dijelaskan fungsinya telah dijelaskan pada tutorial sebelumnya.

Hasil Program





Link

Link video laporan Youtube: https://youtu.be/iboDliPQv4c
Link video laporan repository github:
https://github.com/dehanalkautsar/Tutorial-WebGl/tree/main/How%20It%20Works