Ankara University Faculty of Engineering Computer Engineering Department 2021-22 Fall Semester Computer Graphic Midterm Homework

Course Code: COM3037-B

Course Teacher:

Dr. Emin Emrah ÖZSAVAŞ

Course Asistant:

Arş. Gör. Mert ÇALIŞ

Task:

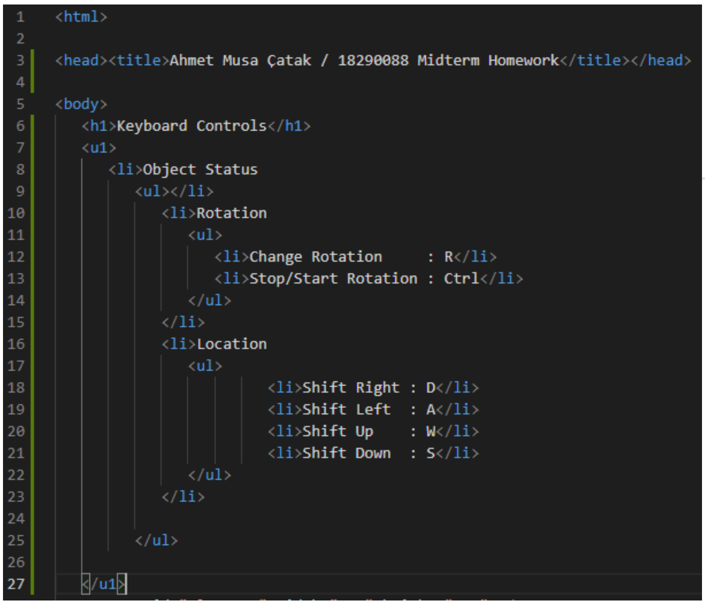
Homework_1.docx

Student Name:

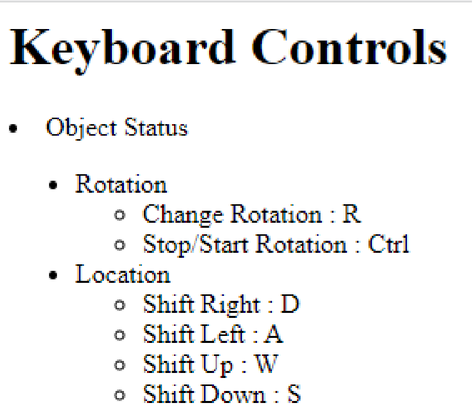
Ahmet Musa Çatak

ID:

18290088



```
1   <html>
2
3   <head><title>Ahmet Musa Çatak / 18290088 Midterm Homework</title></head>
4
5   <body>
6      <h1>Keyboard Controls</h1>
7      <ul>
8         <li>Object Status
9            <ul></li>
10              <li>Rotation
11                 <ul>
12                    <li>Change Rotation     : R</li>
13                    <li>Stop/Start Rotation : Ctrl</li>
14                 </ul>
15              </li>
16              <li>Location
17                 <ul>
18                          <li>Shift Right : D</li>
19                          <li>Shift Left  : A</li>
20                          <li>Shift Up    : W</li>
21                          <li>Shift Down  : S</li>
22                 </ul>
23              </li>
24
25           </ul>
26
27     </ul>
```



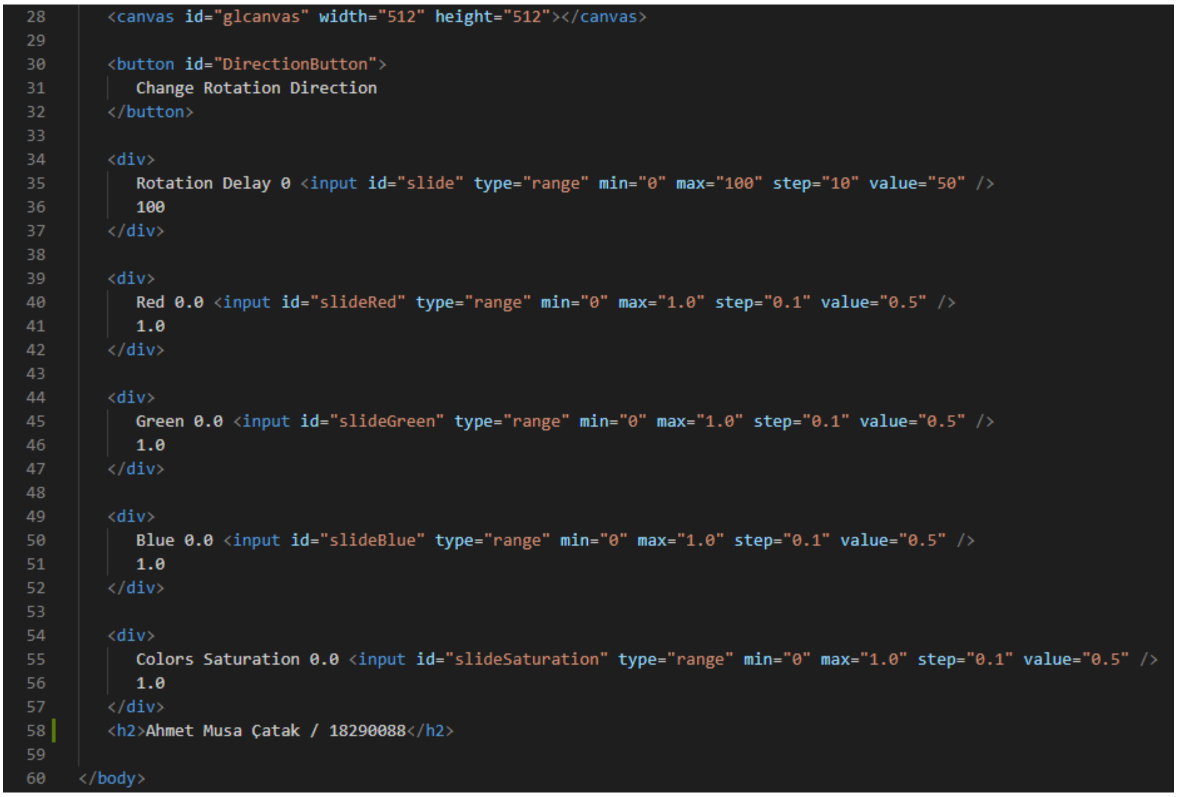
Keyboard Controls

- Object Status

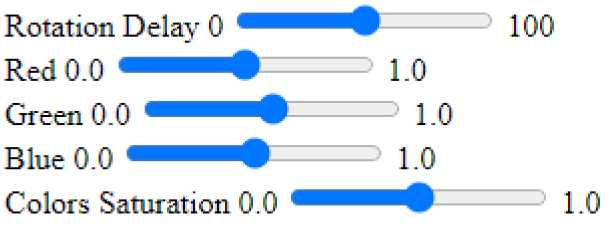
- Rotation
  - Change Rotation : R
  - Stop/Start Rotation : Ctrl
- Location
  - Shift Right : D
  - Shift Left : A
  - Shift Up : W
  - Shift Down : S

In html file line 1 to line 27:

This part of code print some explanation about keyboard controls.



```
28     <canvas id="glcanvas" width="512" height="512"></canvas>
29
30     <button id="DirectionButton">
31        Change Rotation Direction
32     </button>
33
34     <div>
35        Rotation Delay 0 <input id="slide" type="range" min="0" max="100" step="10" value="50" />
36        100
37     </div>
38
39     <div>
40        Red 0.0 <input id="slideRed" type="range" min="0" max="1.0" step="0.1" value="0.5" />
41        1.0
42     </div>
43
44     <div>
45        Green 0.0 <input id="slideGreen" type="range" min="0" max="1.0" step="0.1" value="0.5" />
46        1.0
47     </div>
48
49     <div>
50        Blue 0.0 <input id="slideBlue" type="range" min="0" max="1.0" step="0.1" value="0.5" />
51        1.0
52     </div>
53
54     <div>
55        Colors Saturation 0.0 <input id="slideSaturation" type="range" min="0" max="1.0" step="0.1" value="0.5" />
56        1.0
57     </div>
58     <h2>Ahmet Musa Çatak / 18290088</h2>
59
60  </body>
```



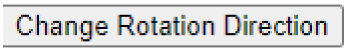
Rotation Delay 0 100

Red 0.0 1.0

Green 0.0 1.0

Blue 0.0 1.0

Colors Saturation 0.0 1.0



In html file line 28 : in this part we add an canvas that contains all objects and shapes. Give and id and space.

Line 30 to line 32: add a button ,that id is DirectionButton, to change direction shape.

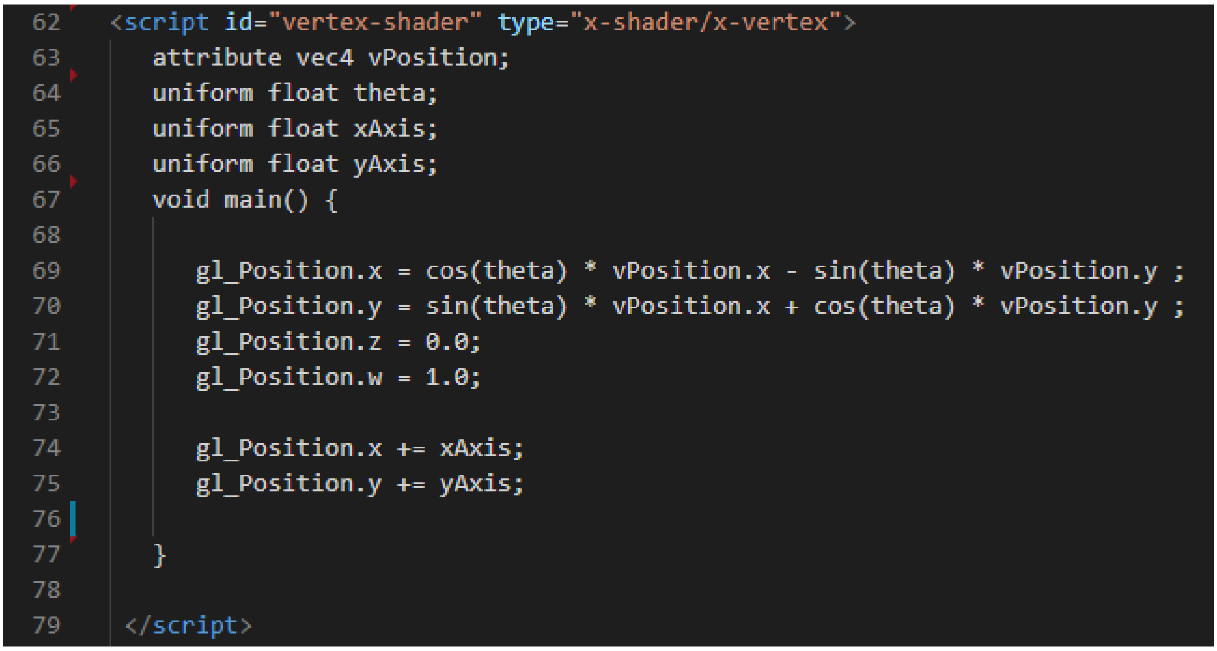
Line 34 to line 37: add a slider that id is slide to change delay variable . That part change the rotation speed of object.

Line 39 to line 42: add a slider that id is slideRed to change red variable. change RGB’s red pigment.

Line 44 to line 47: add a slider that id is slideGreen to change green variable. change RGB’s green pigment.

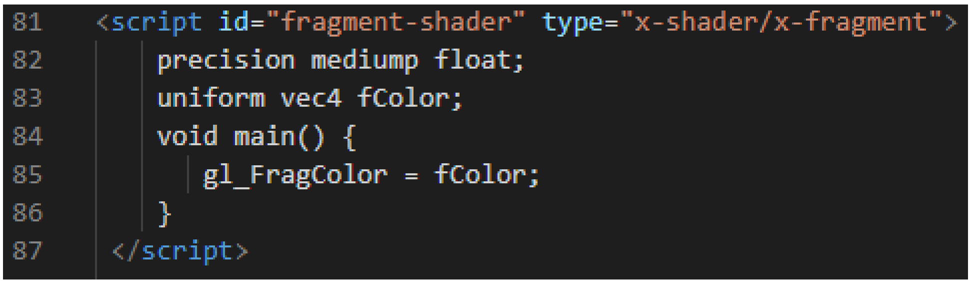
Line 49 to line 52: add a slider that id is slideBlue to change blue variable.change RGB’s blue pigment.

Line 52 to 57: add a slider that id is slideSaturation to change saturation variable.That part change saturation of colors.



```
62  <script id="vertex-shader" type="x-shader/x-vertex">
63      attribute vec4 vPosition;
64      uniform float theta;
65      uniform float xAxis;
66      uniform float yAxis;
67      void main() {
68
69          gl_Position.x = cos(theta) * vPosition.x - sin(theta) * vPosition.y ;
70          gl_Position.y = sin(theta) * vPosition.x + cos(theta) * vPosition.y ;
71          gl_Position.z = 0.0;
72          gl_Position.w = 1.0;
73
74          gl_Position.x += xAxis;
75          gl_Position.y += yAxis;
76
77      }
78
79  </script>
```

In html file line 62 to 77 : add a script, that id is vertex-shader, to keep shape position as a vertices. Some mathematical calculation for rotation , scaling and transition.



```
81  <script id="fragment-shader" type="x-shader/x-fragment">
82      precision mediump float;
83      uniform vec4 fColor;
84      void main() {
85          gl_FragColor = fColor;
86      }
87  </script>
```

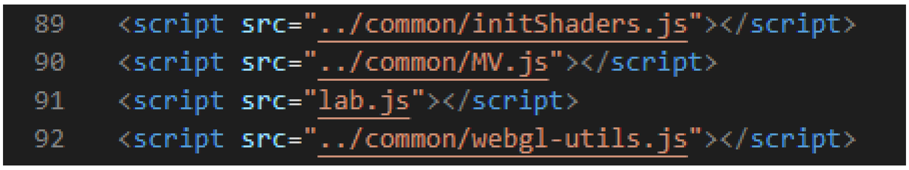
In html file line 81 to line 87 : add a script , that id is fragment-shader, to keep shapes color and precision value.

Line 82 : This part determines how much precision the GPU uses when calculating floats.

We may change this lowp or highp.

Line 82 : add an vector that contains 4 variable to use changing color.

Vec4(Red , Green, Blue, Saturation)

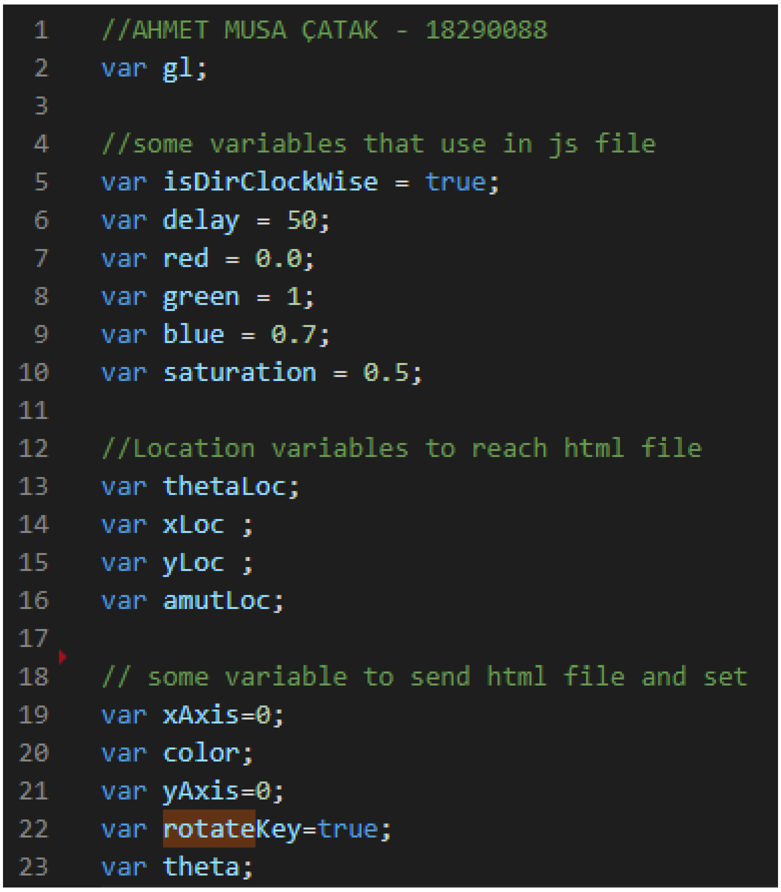


```
89    <script src="../common/initShaders.js"></script>
90    <script src="../common/MV.js"></script>
91    <script src="lab.js"></script>
92    <script src="../common/webgl-utils.js"></script>
```

Add some JavaScript file to our html file.

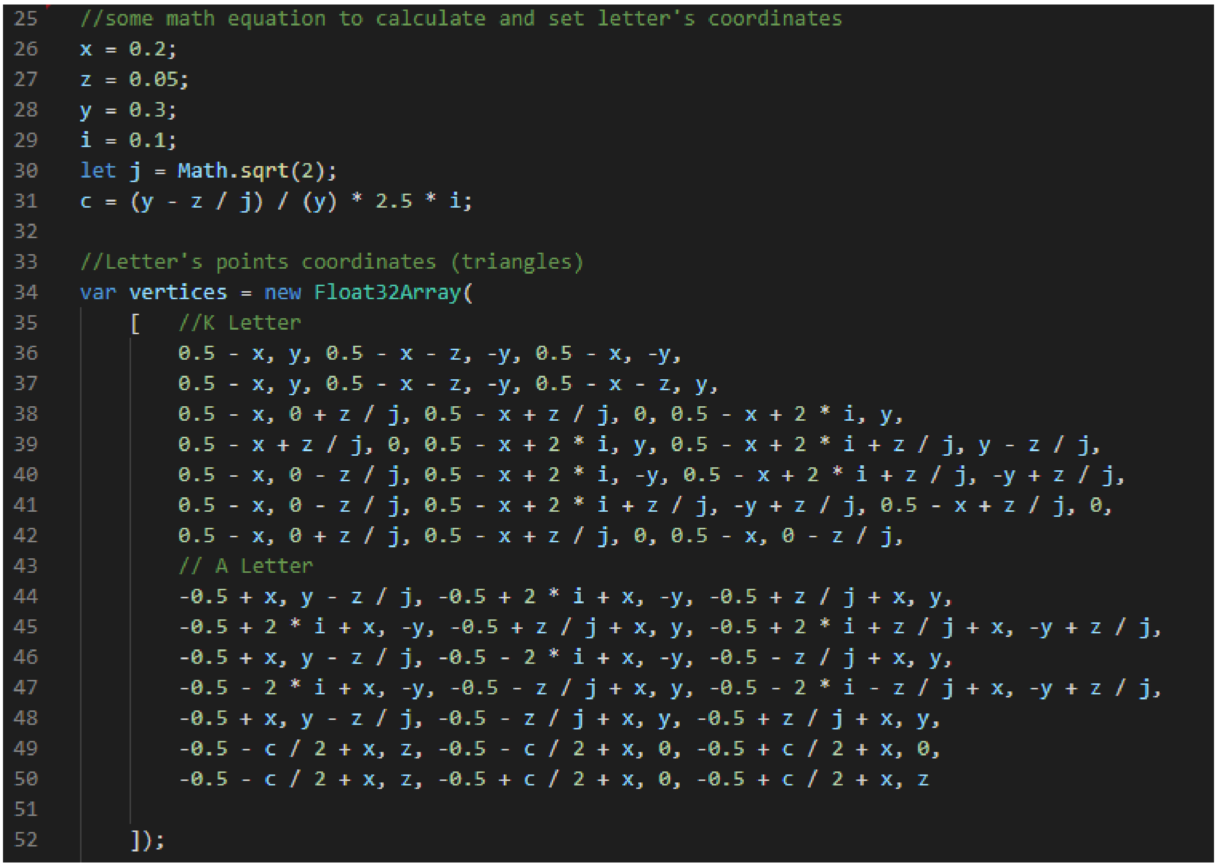
Initshader.js help us to initialize created shaders (vertex and fragment shader).

MV.js contains some functions that we used in main js file. Like vec4.

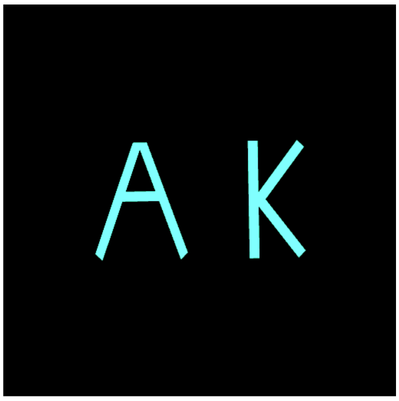


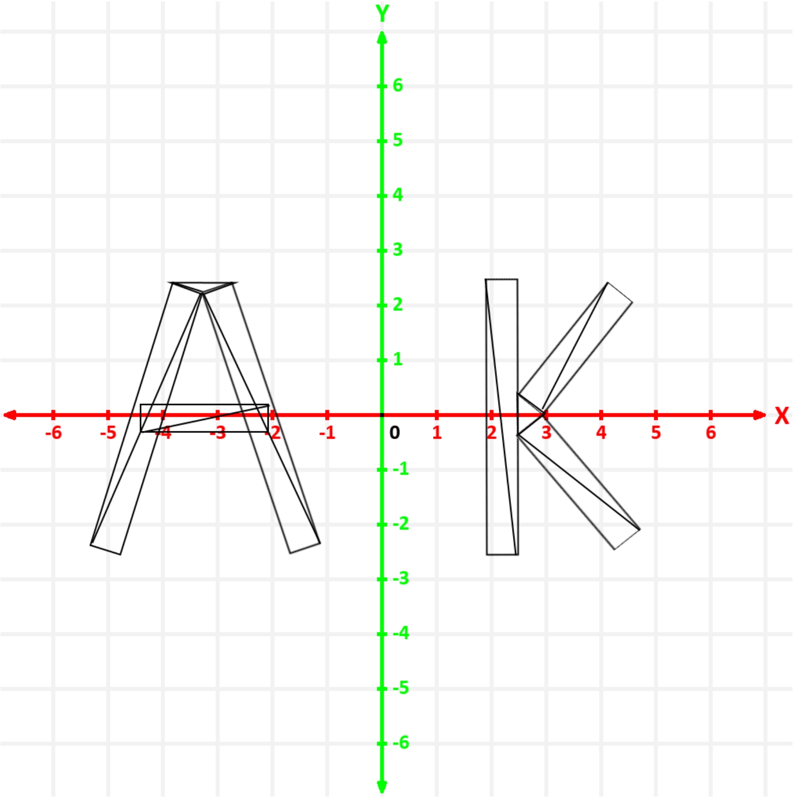
```
1    //AHMET MUSA ÇATAK - 18290088
2    var gl;
3
4    //some variables that use in js file
5    var isDirClockWise = true;
6    var delay = 50;
7    var red = 0.0;
8    var green = 1;
9    var blue = 0.7;
10   var saturation = 0.5;
11
12   //Location variables to reach html file
13   var thetaLoc;
14   var xLoc ;
15   var yLoc ;
16   var amutLoc;
17
18   // some variable to send html file and set
19   var xAxis=0;
20   var color;
21   var yAxis=0;
22   var rotateKey=true;
23   var theta;
```

In javascript file line 1 to line 23 : Define some variables that we use later.



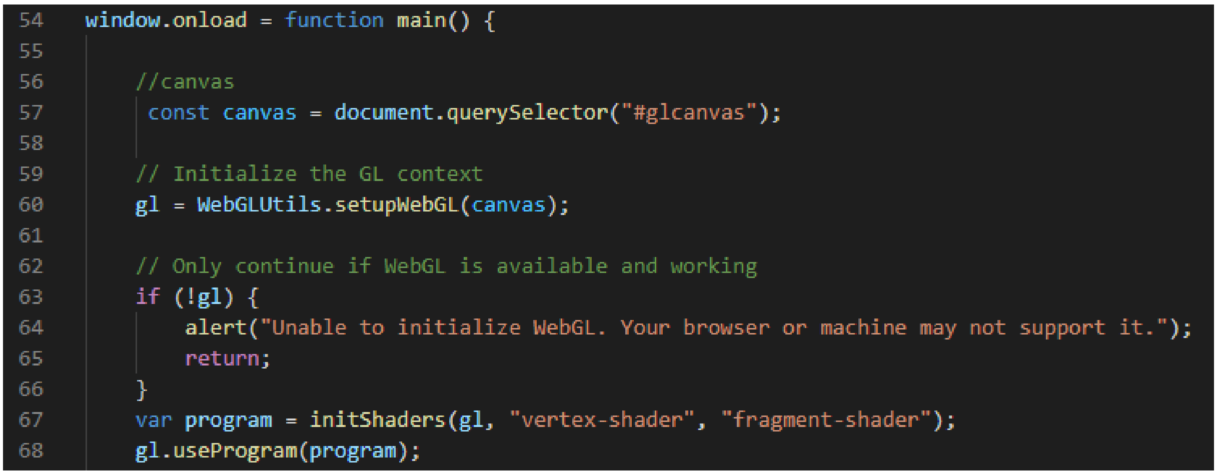
```
25   //some math equation to calculate and set letter's coordinates
26   x = 0.2;
27   z = 0.05;
28   y = 0.3;
29   i = 0.1;
30   let j = Math.sqrt(2);
31   c = (y - z / j) / (y) * 2.5 * i;
32
33   //Letter's points coordinates (triangles)
34   var vertices = new Float32Array(
35       [   //K Letter
36           0.5 - x, y, 0.5 - x - z, -y, 0.5 - x, -y,
37           0.5 - x, y, 0.5 - x - z, -y, 0.5 - x - z, y,
38           0.5 - x, 0 + z / j, 0.5 - x + z / j, 0, 0.5 - x + 2 * i, y,
39           0.5 - x + z / j, 0, 0.5 - x + 2 * i, y, 0.5 - x + 2 * i + z / j, y - z / j,
40           0.5 - x, 0 - z / j, 0.5 - x + 2 * i, -y, 0.5 - x + 2 * i + z / j, -y + z / j,
41           0.5 - x, 0 - z / j, 0.5 - x + 2 * i + z / j, -y + z / j, 0.5 - x + z / j, 0,
42           0.5 - x, 0 + z / j, 0.5 - x + z / j, 0, 0.5 - x, 0 - z / j,
43           // A Letter
44           -0.5 + x, y - z / j, -0.5 + 2 * i + x, -y, -0.5 + z / j + x, y,
45           -0.5 + 2 * i + x, -y, -0.5 + z / j + x, y, -0.5 + 2 * i + z / j + x, -y + z / j,
46           -0.5 + x, y - z / j, -0.5 - 2 * i + x, -y, -0.5 - z / j + x, y,
47           -0.5 - 2 * i + x, -y, -0.5 - z / j + x, y, -0.5 - 2 * i - z / j + x, -y + z / j,
48           -0.5 + x, y - z / j, -0.5 - z / j + x, y, -0.5 + z / j + x, y,
49           -0.5 - c / 2 + x, z, -0.5 - c / 2 + x, 0, -0.5 + c / 2 + x, 0,
50           -0.5 - c / 2 + x, z, -0.5 + c / 2 + x, 0, -0.5 + c / 2 + x, z
51
52       ]);
```





In javascript file line 25 to line 31 : define some variables for calculation of letter’s points.

Line 34 to line 52 : define vertices that contains all points of triangles. Every line contains one triangle points.



```
54  window.onload = function main() {
55  
56      //canvas
57       const canvas = document.querySelector("#glcanvas");
58  
59      // Initialize the GL context
60      gl = WebGLUtils.setupWebGL(canvas);
61  
62      // Only continue if WebGL is available and working
63      if (!gl) {
64          alert("Unable to initialize WebGL. Your browser or machine may not support it.");
65          return;
66      }
67      var program = initShaders(gl, "vertex-shader", "fragment-shader");
68      gl.useProgram(program);
```

In javascript file line 54 : implement our main function that run when browser is run.

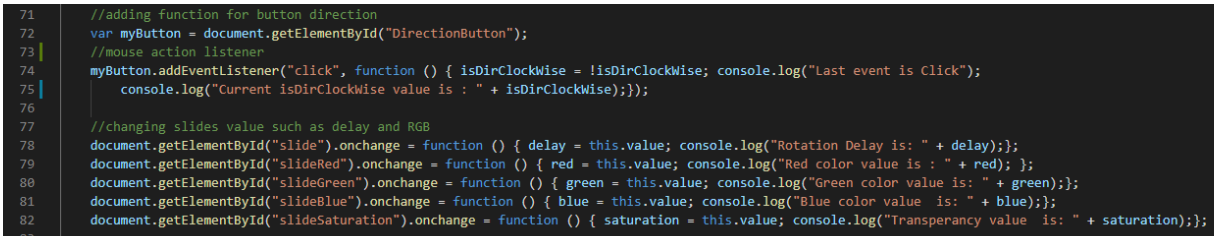
Line 57 : Initialize the GL context const gl = canvas.

Line 60 : initialize the gl context that is type of webglrenderingcontext

Line 63 to 66 : if gl is not implement or not run properly then Show a screen browser that contains this message.

Line 67 : initiliaze our shaders.

Line 68 : sets the specified WebGLProgram as part of the current rendering state.



```
71      //adding function for button direction
72      var myButton = document.getElementById("DirectionButton");
73      //mouse action listener
74      myButton.addEventListener("click", function () { isDirClockWise = !isDirClockWise; console.log("Last event is Click");
75          console.log("Current isDirClockWise value is : " + isDirClockWise);});
76  
77      //changing slides value such as delay and RGB
78      document.getElementById("slide").onchange = function () { delay = this.value; console.log("Rotation Delay is: " + delay);};
79      document.getElementById("slideRed").onchange = function () { red = this.value; console.log("Red color value is : " + red); };
80      document.getElementById("slideGreen").onchange = function () { green = this.value; console.log("Green color value is: " + green);};
81      document.getElementById("slideBlue").onchange = function () { blue = this.value; console.log("Blue color value  is: " + blue);};
82      document.getElementById("slideSaturation").onchange = function () { saturation = this.value; console.log("Transperancy value  is: " + saturation);};
```

Line 72 : add a button that contains in html button id.

Line74 : add eventlistener to mybutton , when event is click change the rotation.

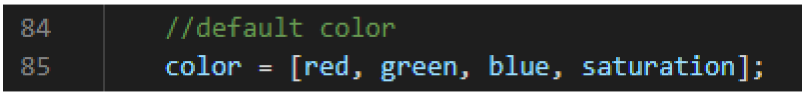
Line 78 : add function to slide which id is slide for changing delay.

Line 79 : add function to slide which id is slideRed for changing RGB’s Red value.

Line 80 : add function to slide which id is slideGreen for changing RGB’s Green value.

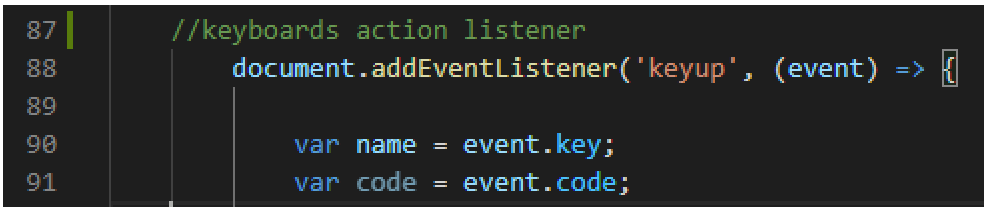
Line 81 : add function to slide which id is slideBlue for changing RGB’s Blue value.

Line 82 : add function to slide which id is slideSaturation for changin colors saturations.



```
84      //default color
85      color = [red, green, blue, saturation];
```

Line 85 : fill color array with default colors value and saturation value.

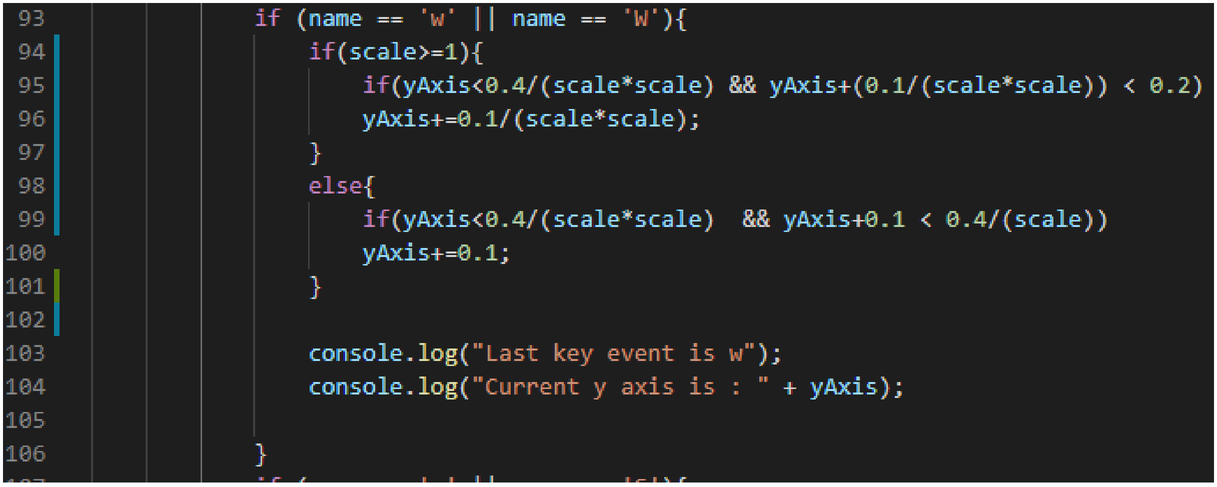


```
87      //keyboards action listener
88          document.addEventListener('keyup', (event) => {
89  
90              var name = event.key;
91              var code = event.code;
```

Line 88 : add eventlistener function that work when keyreleased.

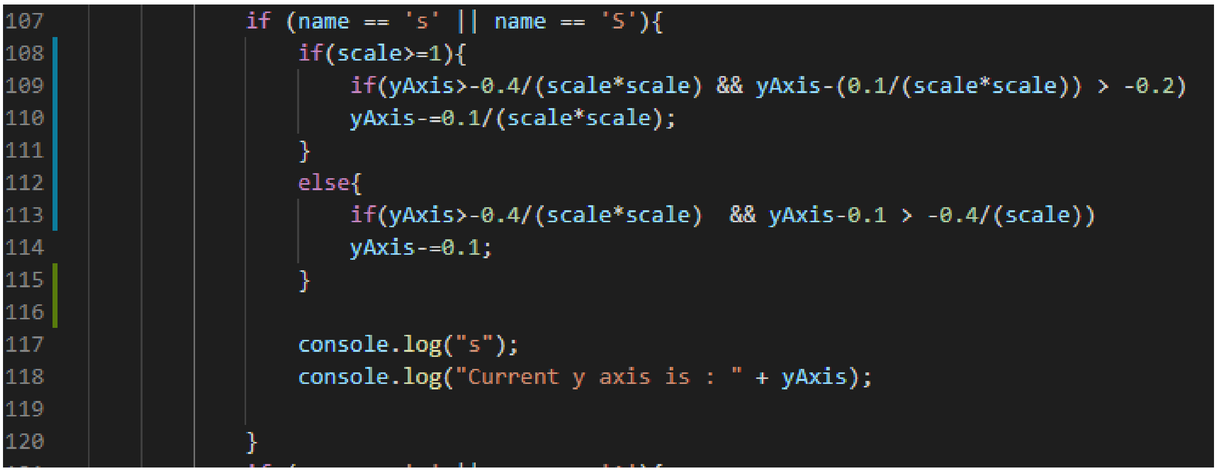
Line 90 : define variable that name is name for keeping which key is released.

Line 91 : define variable that name is code for keeping which key is released .



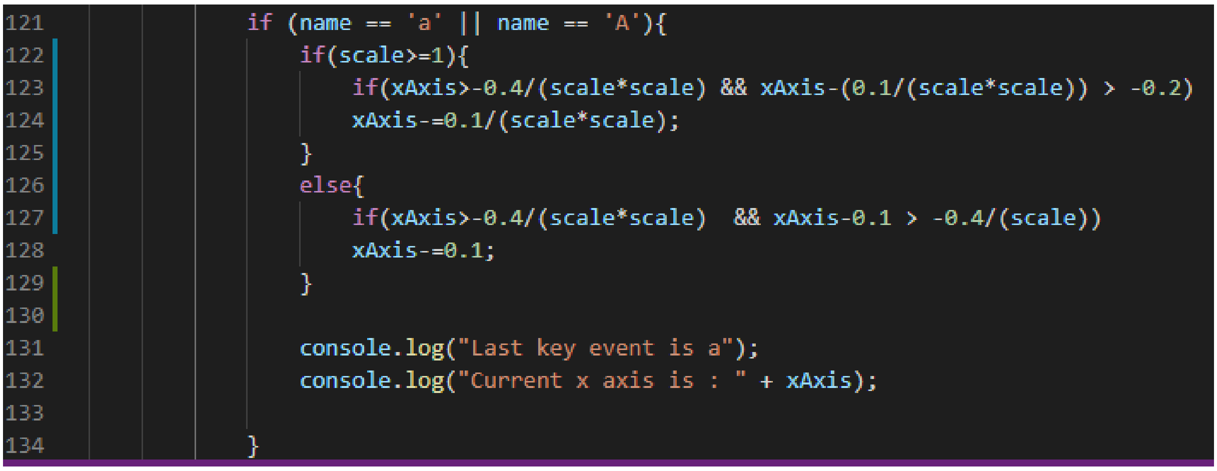
```
93                  if (name == 'w' || name == 'W'){
94                      if(scale>=1){
95                          if(yAxis<0.4/(scale*scale) && yAxis+(0.1/(scale*scale)) < 0.2)
96                          yAxis+=0.1/(scale*scale);
97                      }
98                      else{
99                          if(yAxis<0.4/(scale*scale)  && yAxis+0.1 < 0.4/(scale))
100                         yAxis+=0.1;
101                     }
102 
103                     console.log("Last key event is w");
104                     console.log("Current y axis is : " + yAxis);
105 
106                 }
```

Line 93 to 106 : In this part of code , if released key is w then do these mathematic calculation to increase y location of object.



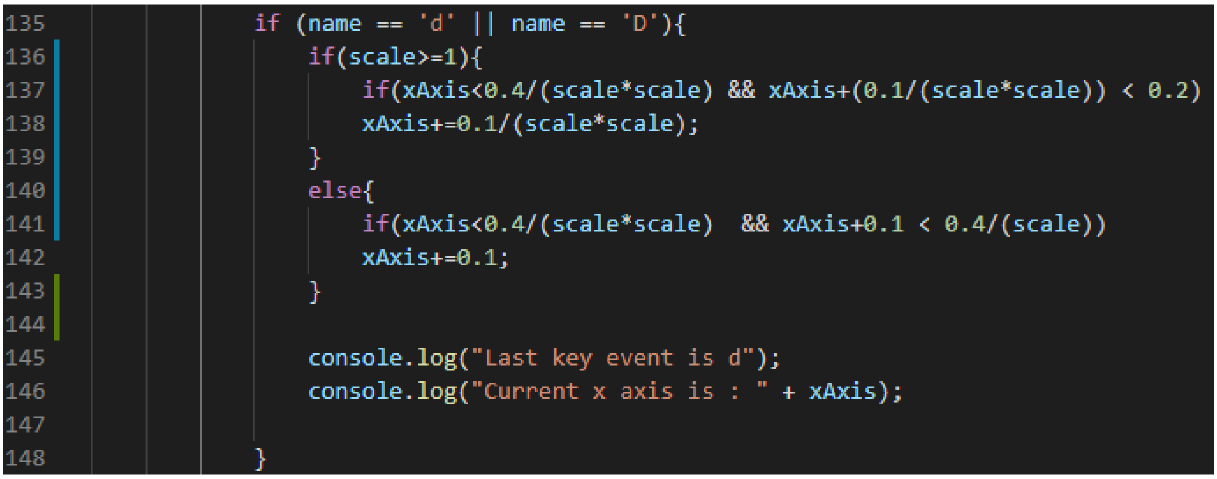
```
107                 if (name == 's' || name == 'S'){
108                     if(scale>=1){
109                         if(yAxis>-0.4/(scale*scale) && yAxis-(0.1/(scale*scale)) > -0.2)
110                         yAxis-=0.1/(scale*scale);
111                     }
112                     else{
113                         if(yAxis>-0.4/(scale*scale)  && yAxis-0.1 > -0.4/(scale))
114                         yAxis-=0.1;
115                     }
116 
117                     console.log("s");
118                     console.log("Current y axis is : " + yAxis);
119 
120                 }
```

Line 107 to 120 : In this part of code , if released key is S then do these mathematic calculation to decrease y location of object.



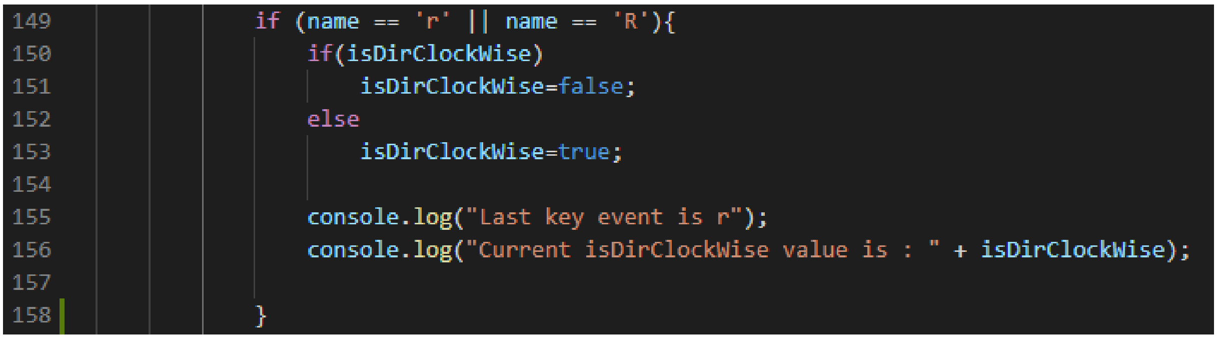
```
121         if (name == 'a' || name == 'A'){
122             if(scale>=1){
123                 if(xAxis>-0.4/(scale*scale) && xAxis-(0.1/(scale*scale)) > -0.2)
124                 xAxis-=0.1/(scale*scale);
125             }
126             else{
127                 if(xAxis>-0.4/(scale*scale)  && xAxis-0.1 > -0.4/(scale))
128                 xAxis-=0.1;
129             }
130 
131             console.log("Last key event is a");
132             console.log("Current x axis is : " + xAxis);
133 
134         }
```

Line 121 to 134 : In this part of code , if released key is A then do these mathematic calculation to decrease x location of object.



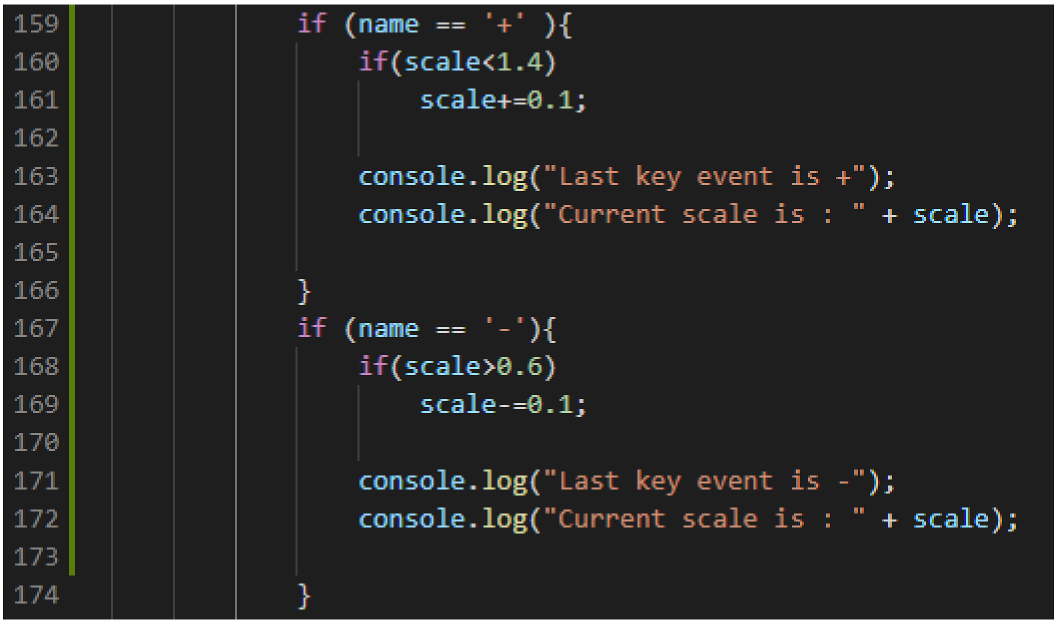
```
135         if (name == 'd' || name == 'D'){
136             if(scale>=1){
137                 if(xAxis<0.4/(scale*scale) && xAxis+(0.1/(scale*scale)) < 0.2)
138                 xAxis+=0.1/(scale*scale);
139             }
140             else{
141                 if(xAxis<0.4/(scale*scale)  && xAxis+0.1 < 0.4/(scale))
142                 xAxis+=0.1;
143             }
144 
145             console.log("Last key event is d");
146             console.log("Current x axis is : " + xAxis);
147 
148         }
```

Line 135 to 148 : In this part of code , if released key is D then do these mathematic calculation to increase x location of object.



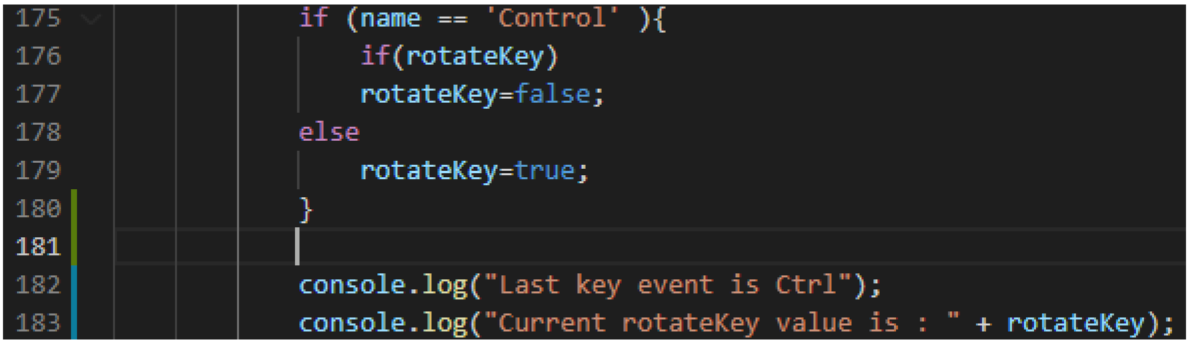
```
149         if (name == 'r' || name == 'R'){
150             if(isDirClockWise)
151                 isDirClockWise=false;
152             else
153                 isDirClockWise=true;
154 
155             console.log("Last key event is r");
156             console.log("Current isDirClockWise value is : " + isDirClockWise);
157 
158         }
```

Line 149 to 158 : In this part of code , if released key is R then change rotation direction.



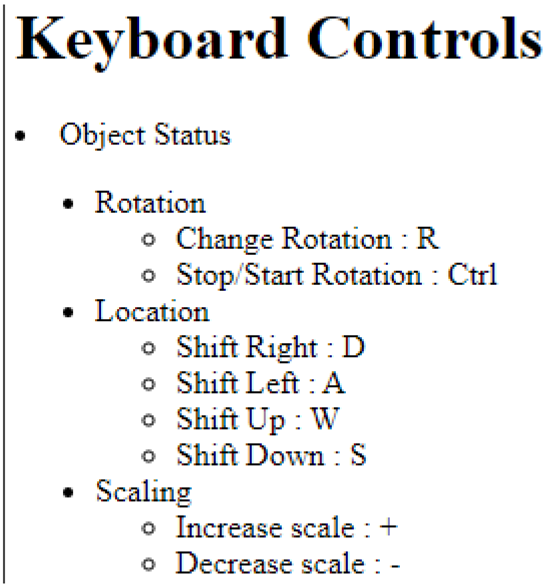
```
159 if (name == '+' ){
160     if(scale<1.4)
161         scale+=0.1;
162 
163     console.log("Last key event is +");
164     console.log("Current scale is : " + scale);
165 
166 }
167 if (name == '-'){
168     if(scale>0.6)
169         scale-=0.1;
170 
171     console.log("Last key event is -");
172     console.log("Current scale is : " + scale);
173 
174 }
```

Line 159 to 174 : In this part of code , if relesed key is + then increase the scale . Otherwise released key is – then decrease the scale.



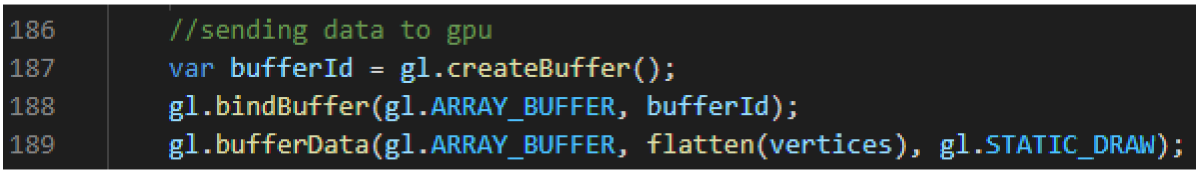
```
175 if (name == 'Control' ){
176     if(rotateKey)
177     rotateKey=false;
178 else
179     rotateKey=true;
180 }
181 
182 console.log("Last key event is Ctrl");
183 console.log("Current rotateKey value is : " + rotateKey);
```

Line 175 to 183 : In this part of code , if released key is Ctrl then stop or start rotating.



Keyboard Controls

- Object Status
  - Rotation
    - Change Rotation : R
    - Stop/Start Rotation : Ctrl
  - Location
    - Shift Right : D
    - Shift Left : A
    - Shift Up : W
    - Shift Down : S
  - Scaling
    - Increase scale : +
    - Decrease scale : -

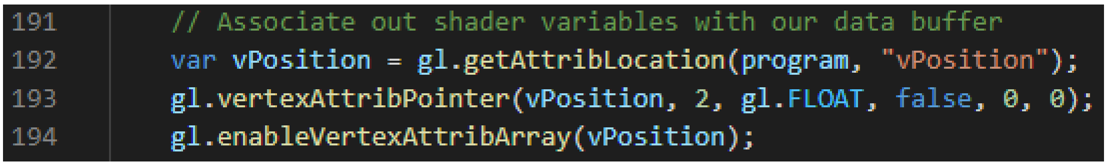


```
186      //sending data to gpu
187      var bufferId = gl.createBuffer();
188      gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
189      gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
```

Line 187 : Create a buffer.

Line 188 : change current buffer as created buffer.

Line 189 : get location data with buffer to draw.

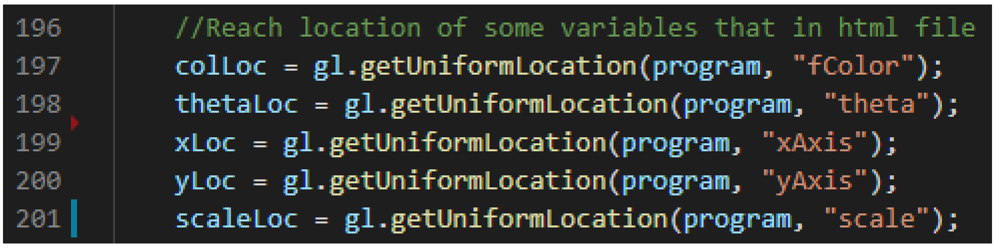


```
191      // Associate out shader variables with our data buffer
192      var vPosition = gl.getAttribLocation(program, "vPosition");
193      gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
194      gl.enableVertexAttribArray(vPosition);
```

Line 192 : create a variable that name is vPosition . Hold vPosition that in html file with that variable.

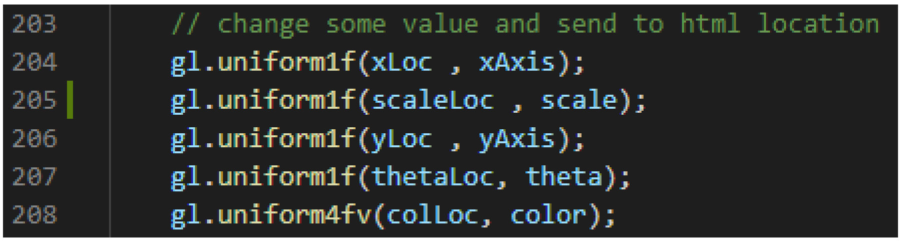
Line 193 : create space for transfer and implement which value is sending.

Line 194 : send back new value if vPositon to html file with that function.



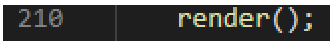
```
196      //Reach location of some variables that in html file
197      colLoc = gl.getUniformLocation(program, "fColor");
198      thetaLoc = gl.getUniformLocation(program, "theta");
199      xLoc = gl.getUniformLocation(program, "xAxis");
200      yLoc = gl.getUniformLocation(program, "yAxis");
201      scaleLoc = gl.getUniformLocation(program, "scale");
```

Line 196 to 201 : In this part of code , hold location of some variable id which created in html file.



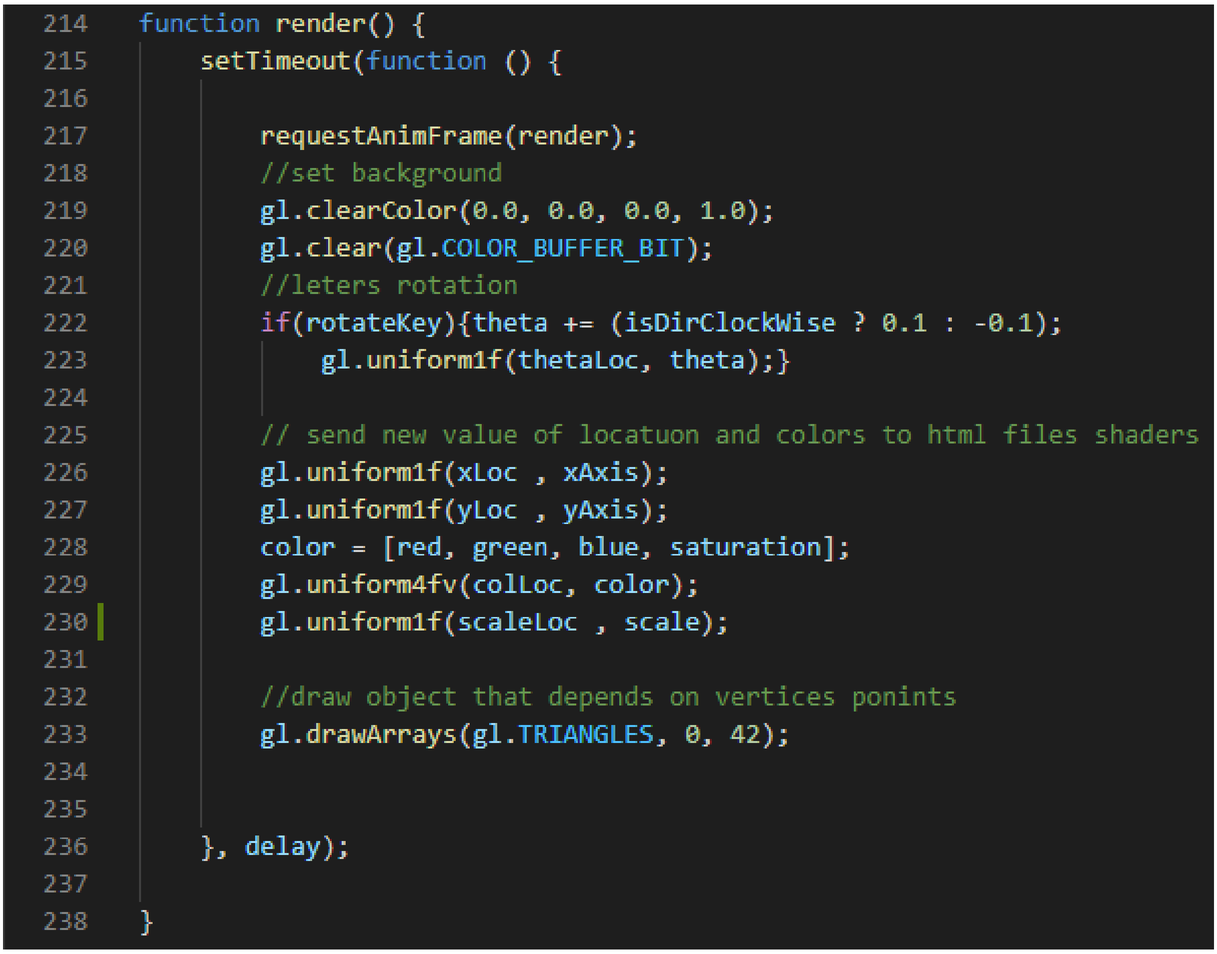
```
203      // change some value and send to html location
204      gl.uniform1f(xLoc , xAxis);
205      gl.uniform1f(scaleLoc , scale);
206      gl.uniform1f(yLoc , yAxis);
207      gl.uniform1f(thetaLoc, theta);
208      gl.uniform4fv(colLoc, color);
```

Line 204 to 208 : In this part of code , send back new value of these variables and chang variables value that in html file.



```
210      render();
```

Line 210: call render() function.



```
214 function render() {
215     setTimeout(function () {
216 
217         requestAnimFrame(render);
218         //set background
219         gl.clearColor(0.0, 0.0, 0.0, 1.0);
220         gl.clear(gl.COLOR_BUFFER_BIT);
221         //leters rotation
222         if(rotateKey){theta += (isDirClockWise ? 0.1 : -0.1);
223             gl.uniform1f(thetaLoc, theta);}
224 
225         // send new value of locatuon and colors to html files shaders
226         gl.uniform1f(xLoc , xAxis);
227         gl.uniform1f(yLoc , yAxis);
228         color = [red, green, blue, saturation];
229         gl.uniform4fv(colLoc, color);
230         gl.uniform1f(scaleLoc , scale);
231 
232         //draw object that depends on vertices ponints
233         gl.drawArrays(gl.TRIANGLES, 0, 42);
234 
235 
236     }, delay);
237 
238 }
```

In render() function :

Line 217 : this method tells the browser that you **wish to perform an animation** and requests that the browser calls a specified function to update an animation before the next repaint.

Line 219 : clear all canvas and paint black.

Line 220 : clear buffer data to reuse.

Line 222 to 224 : set rotation direction.

Line 226 to 230 : send changed data to shaders that in html.

Line 233 : draw the object that created before with triangles function.

Line 215 && 236 : This method calls a function or evaluates an expression after a specified number of milliseconds (delay).

I add a css after write these for smooth draw.