WebGL Academy

Correction

```
GL = CANVAS.getContext("experimental-webgl", {antialias: false});
```

Both Edge and IE still require "experimental-webgl".

1. 2D coloured triangle Exercises

- Add a second triangle to fill the canvas.

  Lines 74

  (re-arrange to make meaning a bit clearer)

```
//POINTS :
var triangle_vertex = [
// colour   |  position
   -1, -1,    0, 0, 1,    //first corner: -> bottom left of the viewport
    1, -1,    1, 1, 0,    //bottom right of the viewport
    1,  1,    1, 0, 0     //top right of the viewport
];
```

```
//POINTS :
var triangle_vertex = [
// colour   |  position
   -1, -1,    0, 0, 1,    //first corner: -> bottom left of the viewport
    1, -1,    1, 1, 0,    //bottom right of the viewport
    1,  1,    1, 0, 0,    //top right of the viewport

   -1, -1,    0, 1, 1,    //first corner: -> bottom left of the viewport
   -1,  1,    1, 1, 1,    //top left of the viewport
    1,  1,    0, 0, 0     //top right of the viewport
];
```

```
//FACES :
var triangle_faces = [0, 1, 2, 3, 4, 5];
```

```
GL.drawElements(GL.TRIANGLES, 6, GL.UNSIGNED_SHORT, 0);
```

Repeats some data. Alternative, call drawElements twice, or use a strip of fan. Fiddly, (winding order).

```
void gl.drawElements(mode, count, type, offset);
```

https://developer.mozilla.org/en-US/docs/Web/API/WebGLRenderingContext/drawElements

- **Extra credit** - Try the above with only four points to render a square (`GL.TRIANGLE_STRIP` or `GL.TRIANGLE_FAN`).

  Notes: reuses colour, fan is usually slow.

```
//POINTS :
var triangle_vertex = [
// colour   |  position
    1, -1,    1, 1, 1,    //top left of the viewport
   -1, -1,    0, 0, 1,    //first corner: -> bottom left of the viewport
    1,  1,    1, 0, 0,    //top right of the viewport
   -1,  1,    1, 1, 0,    //bottom right of the viewport

   -1, -1,    0, 1, 1,    // not used
    1,  1,    0, 0, 0     // BUT!!!
];
```

```
GL.drawElements(GL.TRIANGLE_STRIP, 4, GL.UNSIGNED_SHORT, 0);
```

```
GL.vertexAttribPointer(_color, 3, GL.FLOAT, false, 4 * (2 + 3), 3 * 4);
```

2. 3D coloured triangle Exercises

- Try to rotate the triangle around each axis X,Y,Z

  (already done – derp)

- Change the triangle to a square,

  (as above – dorp)

- Move the square with a funky sinusoidal movement.

  Rather open to interpretation, but…

  Transformation matrix (4×4 matrix) – the notation is a bit ropey

`[1, 0, 0, 0,`
` 0, 1, 0, 0,`
` 0, 0, 1, 0,`
` 0, 0, 0, 1]`
	Identity i.e. no change

`[1, 0, 0, 0,`
` 0, 1, 0, 0,`
` 0, 0, 1, 0,`
` X, Y, Z, 1]`
	Translation Matrix

`[X, 0, 0, 0,`
` 0, Y, 0, 0,`
` 0, 0, Z, 0,`
` 0, 0, 0, 1]`
	Scaling matrix

`[cosθ, -sinθ, 0, 0,`
` sinθ,  cosθ, 0, 0,`
`    0,     0, 1, 0,`
`    0,     0, 0, 1]`
	Rotation matrix

`[cosθ, 0, -sinθ, 0,`
`    0, 1,    0, 0,`
` sinθ, 0,  cosθ, 0,`
`    0, 0,    0, 1]`
	Rotation matrix

`[ 1,    0,     0, 0,`
`  0, cosθ, -sinθ, 0,`
`  0, sinθ,  cosθ, 0,`
`  0,    0,    0, 1]`
	Rotation matrix

lib.js

```
moveToX: function(m, t) {
  m[12]=t;
},

moveToY: function(m, t) {
  m[13]=t;
},

translateX: function(m, t) {
  m[12]+=t;
},
```

script.js

Comment out the rotate* functions to calm things down:

```
var posX = 0;
var posY = 0;

var time_old=0;
var animate=function(time) {
  var dt=time-time_old;

  posX+=dt*0.003;
  posY+=dt*0.001;

  LIBS.moveToX(MOVEMATRIX, Math.cos(posX));
  LIBS.moveToY(MOVEMATRIX, Math.cos(posY));
```

2. 3D coloured cube Exercises

- Replace the cube by a cone.

  Change GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);

  Remove points at top to single point

  Centre top point

  Add points to the side

- Make it grayscale by modifying only the fragment shader.

  Equal RGB values / averaged values (R + G + B / 3)

  Weighted values ((0.3 * R) + (0.59 * G) + (0.11 * B))

  Weighted values more closely represent sensitivity of the eye to each colour

Pixel Spirit

Book of Shadows
http://editor.thebookofshaders.com

http://www.iquilezles.org/apps/graphtoy/
https://thebookofshaders.com/glossary/

Some On-line editors
https://www.shadertoy.com
https://shaderfrog.com/app
http://glslsandbox.com
http://shdr.bkcore.com

Shader Material Editors
http://cyos.babylonjs.com – for materials useful templates
http://www.kickjs.org/example/shader_editor/shader_editor.html

Some Background

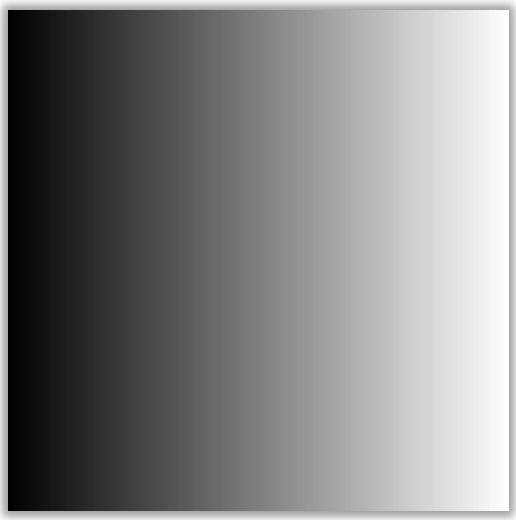
linear
y = mx + c
quadratic
y = x*x*x + x*x - x
sinusoidal
y = sin(x)
y = cos(x)

Step and Smoothstep
Basic

```
void main() {
    vec2 st = gl_FragCoord.xy/u_resolution.xy;
    st.x *= u_resolution.x/u_resolution.y;

    vec3 color = vec3(0.);
    color = vec3(st.x);

    gl_FragColor = vec4(color,1.0);
}
```

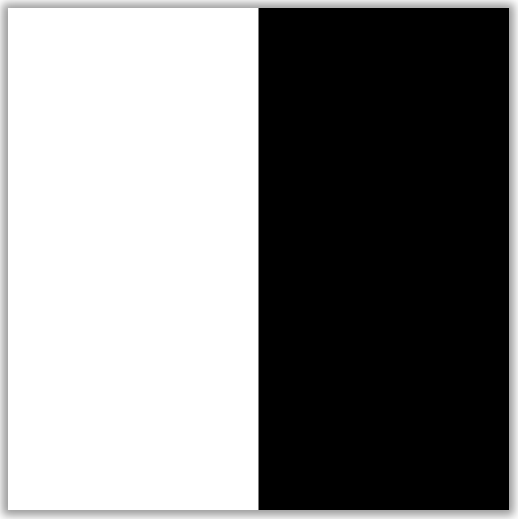


Step non-function

```
void main() {
    vec2 st = gl_FragCoord.xy/u_resolution.xy;
    st.x *= u_resolution.x/u_resolution.y;

    vec3 color = vec3(0.);
    st.x = st.x > 0.5 ? 0.0: 1.0;
    color = vec3(st.x);

    gl_FragColor = vec4(color,1.0);
}
```

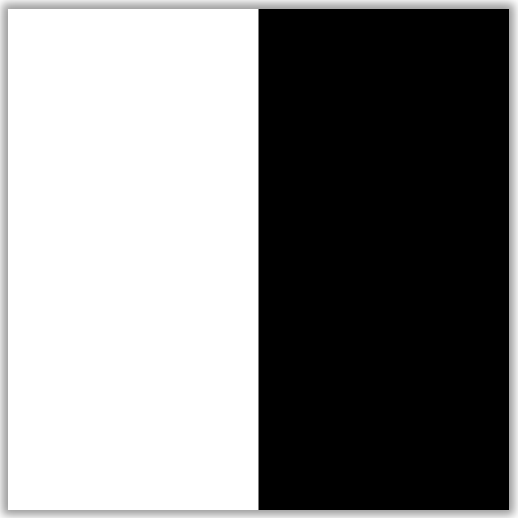


Step function

```
void main() {
    vec2 st = gl_FragCoord.xy/u_resolution.xy;
    st.x *= u_resolution.x/u_resolution.y;

    vec3 color = vec3(0.);
    st.x = step(st.x, 0.5);
    color = vec3(st.x);

    gl_FragColor = vec4(color,1.0);
}
```

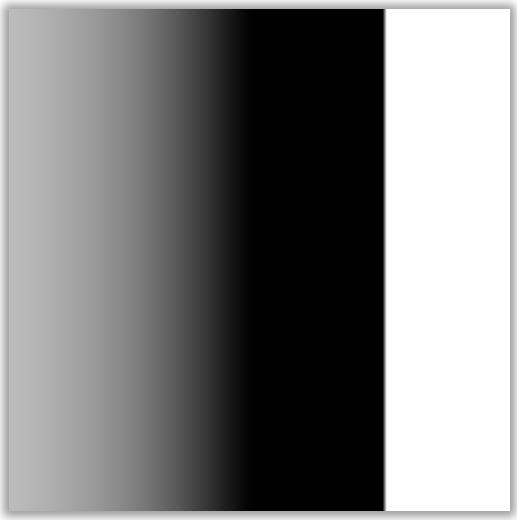


Smoothstep function

```
void main() {
    vec2 st = gl_FragCoord.xy/u_resolution.xy;
    st.x *= u_resolution.x/u_resolution.y;

    vec3 color = vec3(0.);
    st.x = smoothstep(st.x, 0.75, 0.5);
    color = vec3(st.x);

    gl_FragColor = vec4(color,1.0);
}
```



Solutions

https://github.com/patriciogonzalezvivo/PixelSpiritDeck/tree/master/00-elements