

```

////////////////////////////////////
// Sónidmi Tlvugrafk
// Sónir notkun lyklaboratburum til að hreyfa spáa
//
// Hjálmtir Hafsteinsson, janúar 2022
////////////////////////////////////

var canvas;

var gl;

var ymove = 0.1;

var jumping = false;

var xmove = 0;

var program;

var program1;

var sr_vPosition;

var vPosition;

var vertices = [
    vec2( -0.1, -0.9 ),
    vec2( -0.1, -0.4),
    vec2( 0.1, -0.65 ),
];

window.onload = function init() {

    canvas = document.getElementById( "gl-canvas" );

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

    gl.viewport( 0, 0, canvas.width, canvas.height );

```

```

gl.clearColor( 0.8, 0.8, 0.8, 1.0 );

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


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


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


// Event listener for keyboard
window.addEventListener("keydown", function(e){
    switch( e.keyCode ) {
        case 37: // vinstri ⬅
            xmove = -0.04;

            if(vertices[2][0]>0 && vertices[1][0]>0 && vertices[2][0]-
vertices[1][0]>0){
                vertices[2][0] -= 0.4;
            }

            else if(vertices[2][0]>0 && vertices[1][0]<0){
                vertices[2][0] -= 0.4;
            }
    }
}

```

```

    }
    else if(vertices[2][0]<0 && vertices[1][0]<0 && vertices[2][0]-
vertices[1][0]>0){
        vertices[2][0] -= 0.4;
    }

    break;
case 39: // h◇gri◇r
    xmove = 0.04;

    if(vertices[2][0]>0 && vertices[1][0]>0 && vertices[2][0]-
vertices[1][0]<0){
        vertices[2][0] += 0.4;
    }

    else if(vertices[2][0]<0 && vertices[1][0]>0){
        vertices[2][0] += 0.4;
    }

    else if(vertices[2][0]<0 && vertices[1][0]<0 && vertices[2][0]-
vertices[1][0]<0){
        vertices[2][0] += 0.4;
    }

    break;

case 38:
    jumping = true;
    if( vertices[0][1] <= -0.8){
        for(i = 0; i<3; i++){
            vertices[i][1] += ymove;
        }
    }

    break;

case 32:
    jumping = true;

```

```

        if( vertices[0][1] <= -0.8){
            for(i = 0; i<3; i++){
                vertices[i][1] += ymove;
            }
        }
        break;

    default:
        xmove = 0.0;
    }
    for(i=0; i<3; i++) {
        vertices[i][0] += xmove;
    }

    gl.bufferSubData(gl.ARRAY_BUFFER, 0, flatten(vertices));
});

render();
}

function rand(){
    var ran = Math.random();
    if(ran<0.5){
        ran = ran*10;
        ran = Math.round(ran);
        ran = ran/10;
        if(vertices[2][0]>0 && vertices[1][0]>0 && vertices[2][0]-vertices[1][0]>0){
            var gold = [vec2(vertices[2][0]-ran, -0.9) , vec2(vertices[2][0]-ran-0.1, -0.9),
            vec2(vertices[2][0]-ran, -0.8)];
        }

        else if(vertices[2][0]>0 && vertices[1][0]<0){

```

```

        var gold = [vec2(vertices[2][0]+ran, -0.9) , vec2(vertices[2][0]+ran+0.1, -0.9),
vec2(vertices[2][0]+ran, -0.8)];
    }

    else if(vertices[2][0]<0 && vertices[1][0]<0 && vertices[2][0]-vertices[1][0]>0){

        var gold = [vec2(vertices[2][0]+ran, -0.9) , vec2(vertices[2][0]+ran+0.1, -0.9),
vec2(vertices[2][0]+ran, -0.8)];
    }

    else if(vertices[2][0]>0 && vertices[1][0]>0 && vertices[2][0]-vertices[1][0]<0){

        var gold = [vec2(vertices[2][0]+ran, -0.9) , vec2(vertices[2][0]+ran+0.1, -0.9),
vec2(vertices[2][0]+ran, -0.8)];
    }

}

    else if(vertices[2][0]<0 && vertices[1][0]>0){

        var gold = [vec2(vertices[2][0]+ran, -0.9) , vec2(vertices[2][0]+ran+0.1, -0.9),
vec2(vertices[2][0]+ran, -0.8)];
    }

    else if(vertices[2][0]<0 && vertices[1][0]<0 && vertices[2][0]-vertices[1][0]<0){

        var gold = [vec2(vertices[2][0]+ran, -0.9) , vec2(vertices[2][0]+ran+0.1, -0.9),
vec2(vertices[2][0]+ran, -0.8)];
    }
}

    program1 = initShaders( gl, "vertex-shader", "fragment-shader" );

    // Load the data into the GPU

    sr_bufferId = gl.createBuffer();

    gl.bindBuffer( gl.ARRAY_BUFFER, sr_bufferId );

    gl.bufferData( gl.ARRAY_BUFFER, flatten(gold), gl.STATIC_DRAW );

    // Associate out shader variables with our data buffer

    sr_vPosition = gl.getAttribLocation( program, "vPosition" );

}

var sr_bufferID;

var bufferId;

```

```

function render() {

    //rand();

    gl.clear( gl.COLOR_BUFFER_BIT );
    gl.drawArrays( gl.TRIANGLE_FAN, 0, 3 );

    if(jumping && vertices[0][1] == -0.8){
        for(i = 0; i<3; i++){
            vertices[i][1] -= ymove;
            jumping = false;
        }
    }
    xmove = 0;

    window.requestAnimationFrame(render);
}

```

```

<!DOCTYPE html>

```

```

<html>

```

```

<script id="vertex-shader" type="x-shader/x-vertex">

```

```

attribute vec4 vPosition;

```

```

void

```

```

main()

```

```

{

```

```
    gl_Position = vPosition;
}
</script>
```

```
<script id="fragment-shader" type="x-shader/x-fragment">
```

```
precision mediump float;
```

```
varying vec4 fColor;
void
main()
{
    gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
}
</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="marius.js"></script>
```

```
<body>
```

```
<canvas id="gl-canvas" width="1000" height="600">
```

```
</body>
```

```
</html>
```

<https://github.com/tfj2/tolvugrafik/tree/master/Verkefni%201>