

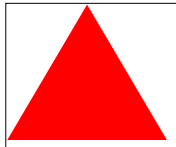
Introduction to Computer Graphics with WebGL

Ed Angel

A Simple Example

A Really Simple Example


- Generate one red triangle
- Has all the elements of a more complex application
 - vertex shader
 - fragment shader
 - HTML canvas



- www.cs.unm.edu/~angel/WebGL/COURSE/triangle.html

Tasks

- HTML file
 - describe page layout including canvas element
 - read in utility and application JS files
 - contain vertex and fragment shaders
- JS file
 - set up WebGL context
 - set up buffers in GPU
 - compile and link shaders with application
 - generate data and place on GPU
 - render data




THE UNIVERSITY of
NEW MEXICO

triangle.html

```
<html>
<head>
<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;
void main()
{
    gl_FragColor = vec4( 1.0, 0.0, 0.0, 1.0 );
}
</script>
```

Computer Graphics with WebGL © Ed Angel, 2014

4



THE UNIVERSITY of
NEW MEXICO


triangle.html

```
<script type="text/javascript" src="../Common/webgl-utils.js"></script>
<script type="text/javascript" src="../Common/initShaders.js"></script>
<script type="text/javascript" src="triangle.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>
```

Computer Graphics with WebGL © Ed Angel, 2014

5



THE UNIVERSITY of
NEW MEXICO

triangle.js

```
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" ); }
}

var vertices = new Float32Array([-1, -1, 0, 1, 1, -1]);

// Configure WebGL

gl.viewport( 0, 0, canvas.width, canvas.height );
gl.clearColor( 1.0, 1.0, 1.0, 1.0 );
```

Computer Graphics with WebGL © Ed Angel, 2014

6

triangle.js

```
// 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, vertices,
    gl.STATIC_DRAW );
```

triangle.js

```
// 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 );

render();
};

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