THE UNIVERSITY of NEW MEXICO

Introduction to Computer Graphics with WebGL

Ed Angel

Square Program
Part 4

THE UNIVERSITY of NEW MEXICO

square.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" );
}
```

`onload`: determines where to start execution when all code is loaded

gl gets canvas from HTML file using id assigned there
gl gets WebGL context from utility read in by HTML file

THE UNIVERSITY of NEW MEXICO

square.js (cont)

```
// four vertices

  var vertices = [
     -0.5, -0.5,
     -0.5,  0.5,
      0.5, 0.5,
      0.5, -0.5
  ];
// alternative var vertices = [vec2(-0.5, -0.5), vec2(-0.5, 0.5), vec2(0.5, 0.5), vec2(0.5, 0.5)];
```

- JS array is not the same as a C or Java array
  - object with methods
  - vertices.length // 4
- Values in clip coordinates
  - square fits in default view volume
- Alternative uses data types in MV.js that match GLSL

square.js (cont)

```
// Configure WebGL

gl.viewport( 0, 0, canvas.width, canvas.height );
gl.clearColor( 0.0, 0.0, 0.0, 1.0 );

// Load shaders and initialize attribute buffers

var program = initShaders( gl, "vertex-shader", "fragment-shader" );
gl.useProgram( program );
```

- `gl.viewport` specifies area of canvas to use
- `initShaders` used to load, compile and link shaders to form a program object
- program object is a container for shaders
- `gl.useProgram` selects current program object

square.js (cont)

```
// Load the data into the GPU

var bufferId = gl.createBuffer();
gl.bindBuffer( gl.ARRAY_BUFFER, bufferId );
gl.bufferData( gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW );
```

- Load data onto GPU by
  - creating a **vertex buffer object** (VBO) on the GPU
  - making it the current VBO
  - loading data into it
  - Note use of `flatten()` to convert JS array to an array of float32's
- Other VBOs can contain data such as colors and texture coodinates

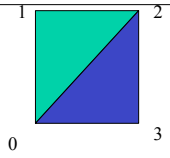
square.js (cont)

```
// Associate shader variables with variables in JS file

var vPosition = gl.getAttribLocation( program, "vPosition" );
gl.vertexAttribPointer( vPosition, 2, gl.FLOAT, false, 0, 0 );
gl.enableVertexAttribArray( vPosition );
```

- Finally we must connect each variable in program with corresponding variable in shader
  - need name, type, location in buffer

square.js (cont)

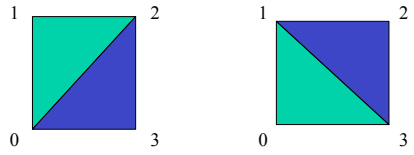
```
   render();
};

function render() {
   gl.clear( gl.COLOR_BUFFER_BIT );
   gl.drawArrays( gl.TRIANGLE_FAN, 0, 4 );
}
```



Triangles, Fans or Strips

gl.drawArrays(gl.TRIANGLES, 0, 6); // 0, 1, 2, 0, 2, 3

gl.drawArrays(gl.TRIANGLE_FAN, 0, 4); // 0, 1 , 2, 3



gl.drawArrays(gl.TRIANGLE_STRIP, 0, 4); // 0, 1, 3, 2