Introduction to WebGL with three.js

https://threejs.org/

Main components of a three.js interactive

- scene : object containing all the different items that you want to draw (`THREE.Scene`)
- renderer : this draws the scene onto your computer screen (`THREE.WebGLRenderer`)
- camera : sets the viewing position and angle (`THREE.PerspectiveCamera`)
- controls : allows you to move the camera around with the mouse and/or keyboard (`THREE.TrackballControls`)

My usual code layout

1. Read in some data file (if relevant), using d3
2. Initialize the scene, renderer, camera, controls
3. Initialize a gui if needed (e.g., dat.gui: http://workshop.chromeexperiments.com/examples/gui)
4. Draw each item (i.e. each mesh) and add them to the scene
    a. A mesh consists of a geometry a material
5. Start the animation loop
    a. Checks for any updates from the controls, keyboard, etc.
    b. Redraws scene in your browser each refresh time (typically 60 times per second)
    c. Even if you don't change anything in the scene, this is still running the background

Mesh objects

- Geometries
    - In general, geometries are defined by x,y,z vertices that combine to draw triangles
    - Geometries define the shape of your object
    - Three.js has many different 3D polygons (e.g., `THREE.SphereGeometry`, `THREE.BoxGeometry`, etc) already built in
    - You can also construct 2D shapes (`THREE.Shape`)
    - You can also build your own custom 3D shapes by specifying vertices, or extruding from a shape, etc.
- Materials
    - In general, materials define the look of the object (e.g., the color, shininess, texture, etc.)
    - Three.js has many different materials, each with many different options to choose from
    - One particularly useful for us: if you want to plot a bunch of points in 3D space, you can use a point cloud method (`THREE.PointsMaterial`)
    - You can also define your own custom "shaders" to further manipulate the look of each geometry
    - You can apply a "texture" (i.e., an image) to a given geometry via the material

A simple script to get started:
https://threejs.org/docs/#manual/en/introduction/Creating-a-scene

```
<html>
      <head>
            <title>My first three.js app</title>
            <style>
                  body { margin: 0; }
                  canvas { width: 100%; height: 100% }
            </style>
      </head>
      <body>
            <script src="js/three.js"></script>
            <script>
                  var scene = new THREE.Scene();
                  var camera = new THREE.PerspectiveCamera( 75,
                  window.innerWidth/window.innerHeight, 0.1, 1000 );

                  var renderer = new THREE.WebGLRenderer();
                  renderer.setSize( window.innerWidth, window.innerHeight );
                  document.body.appendChild( renderer.domElement );

                  var geometry = new THREE.BoxGeometry( 1, 1, 1 );
                  var material = new THREE.MeshBasicMaterial( {color: 0x00ff00} );
                  var cube = new THREE.Mesh( geometry, material );

                  scene.add( cube );

                  camera.position.z = 5;

                  var animate = function () {
                        requestAnimationFrame( animate );
                        cube.rotation.x += 0.01;
                        cube.rotation.y += 0.01;
                        renderer.render( scene, camera );
                  };

                  animate();
            </script>
      </body>
</html>
```