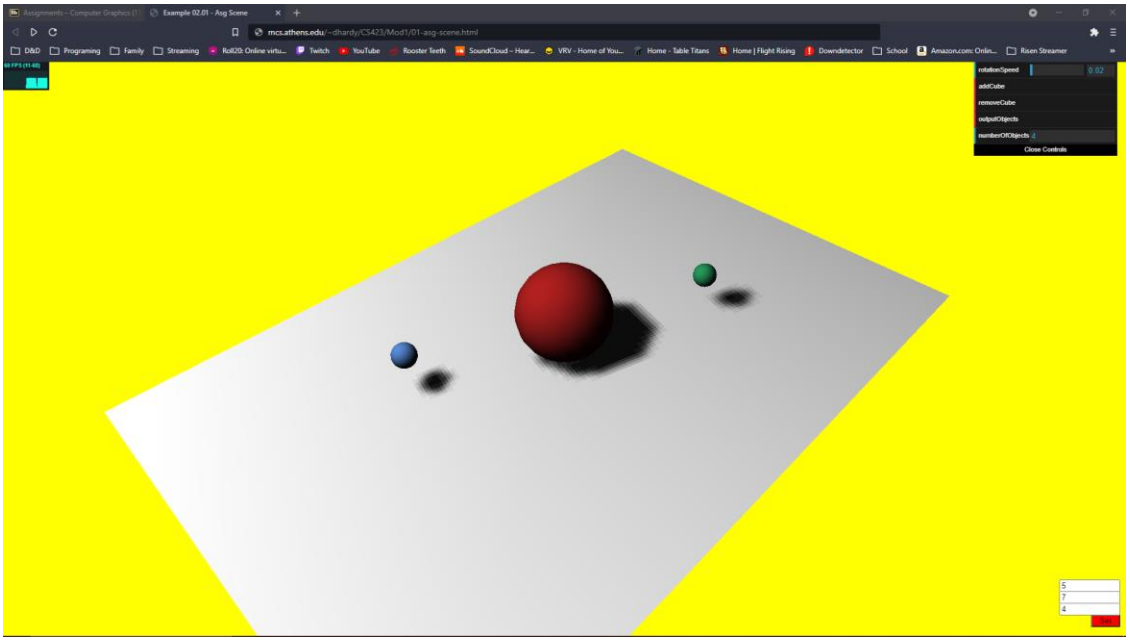
Devin Hardy

00076619

CS423



```
<!DOCTYPE html>
<HTML>
<HEAD>
	<TITLE>Example 02.01 - Asg Scene</TITLE>
	<SCRIPT TYPE="text/javascript" SRC="../libs/three.js"></SCRIPT>
	<SCRIPT TYPE="text/javascript" SRC="../libs/stats.min.js"></SCRIPT>
	<SCRIPT type="text/javascript" src="../libs/dat.gui.min.js"></SCRIPT>
	<style>
		body {
		/* set margin to 0 and overflow to hidden, to go fullscreen */
			margin: 0;
			overflow: hidden;
		}
	</style>
```

```
<!-- Attempt at adding a imput box positioning-->
        <style>
        .poz
        {
        position:absolute;bottom:35px; right:10px; width:5%;
        }
        .boz
        {
        position:absolute;bottom:55px; right:10px; width:5%;
        }
        .doz
        {
        position:absolute;bottom:75px; right:10px; width:5%;
        }
        button
        {
        background-color: red;
        position:absolute;bottom:15px; right:10px; width:50px;
        }
        </style>

</head>
<body>
<div id="Stats-output">
</div>

<!-- Area to input NO clue what else to do -->
```

```
<!-- Div which will hold the Output -->
<Div id="WebGL-output">
</Div>

<!-- Javascript code that runs our three.js examples -->
<script type="text/javascript" src="01-asg-scene.js">
</script>

<form>
        <input id = "x" type="number" step="0.01"  class="doz"> <!-- For x -->
        <input id = "y" type="number" step="0.01"  class="boz"> <!-- For y -->
        <input id = "z" type="number" step="0.01"  class="poz"> <!-- For z -->
        <button id="button">
                Set
        </button>
</form>



</body>
</HTML>



//
// File: 01-asg-scene.js
// Demo some of the basics of working with the scenegraph
// This is an extension of code from he Learning THREE.js textbook
// once everything is loaded. we run our Three.js stuff
var camera
```

```
function init() {

        var stats = initStats();

        function initStats() {
                var stats = new Stats();

                stats.setMode(0); // 0: fps, 1: ms

                //Align top-left
                stats.domElement.style.position = 'absolute';
                stats.domElement.style.left = '0px';
                stats.domElement.style.top = '0px';
                document.getElementById("Stats-output").appendChild(stats.domElement);
                return stats;
        }

        // create a scene, that will hold all our elements such as objects, cameras and lights.
    var scene = new THREE.Scene();

    // create a camera, which defines where we're looking at.
    camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);
    scene.add(camera);

    // create a render and set the size
    var renderer = new THREE.WebGLRenderer();

    renderer.setClearColor(new THREE.Color(0xEEEEEE, 1.0));
```

```
renderer.setSize(window.innerWidth, window.innerHeight);
renderer.shadowMapEnabled = true;

// create the ground plane
var planeGeometry = new THREE.PlaneGeometry(60, 40, 1, 1);
var planeMaterial = new THREE.MeshLambertMaterial({color: 0xffffff});
var plane = new THREE.Mesh(planeGeometry, planeMaterial);
plane.receiveShadow = true;

// rotate and position the plane
plane.rotation.x = -0.5 * Math.PI;
plane.position.x = 0;
plane.position.y = 0;
plane.position.z = 0;

// add the plane to the scene
scene.add(plane);

// position and point the camera to the center of the scene
camera.position.x = -30;
camera.position.y = 40;
camera.position.z = 30;
camera.lookAt(scene.position);


// add subtle ambient lighting
var ambientLight = new THREE.AmbientLight(0x0c0c0c);
scene.add(ambientLight);
```

```
// add spotlight for the shadows
var spotLight = new THREE.SpotLight(0xffffff);
spotLight.position.set(-40, 60, -10);
spotLight.castShadow = true;
scene.add(spotLight);

// add the output of the renderer to the html element
document.getElementById("WebGL-output").appendChild(renderer.domElement);



var controls = new function () {
  this.rotationSpeed = 0.02;
  this.numberOfObjects = scene.children.length;

  this.removeCube = function () {
    var allChildren = scene.children;
    var lastObject = allChildren[allChildren.length - 1];
    if (lastObject instanceof THREE.Mesh) {
      scene.remove(lastObject);
      this.numberOfObjects = scene.children.length;
    }
  };

  this.addCube = function () {

    var cubeSize = Math.ceil((Math.random() * 3));
    var cubeGeometry = new THREE.BoxGeometry(cubeSize, cubeSize, cubeSize);
    var cubeMaterial = new THREE.MeshLambertMaterial({color: Math.random() * 0xffffff});
    var cube = new THREE.Mesh(cubeGeometry, cubeMaterial);
```

```
        cube.castShadow = true;
        cube.name = "cube-" + scene.children.length;



        // position the cube randomly in the scene

        cube.position.x = -30 + Math.round((Math.random() * planeGeometry.parameters.width));
        cube.position.y = Math.round((Math.random() * 5));
        cube.position.z = -20 + Math.round((Math.random() * planeGeometry.parameters.height));

        // add the cube to the scene
        scene.add(cube);
        this.numberOfObjects = scene.children.length;
    };

    this.outputObjects = function () {
        console.log(scene.children);
    }
};

        //Placing other objects
        // Add a Large sphere
var sphereGeometry = new THREE.SphereGeometry(4, 20, 20);
var sphereMaterial = new THREE.MeshLambertMaterial({color: 0xAC2020, wireframe: false});
var sphere = new THREE.Mesh(sphereGeometry, sphereMaterial);
        sphere.castShadow = true;
sphere.position.x = 0;
sphere.position.y = 4;
sphere.position.z = 0;
```

```
scene.add(sphere);

        // Add a Small sphere 1
var sphereGeometry = new THREE.SphereGeometry(1, 20, 20);
var sphereMaterial = new THREE.MeshLambertMaterial({color: 0x27955A, wireframe: false});
var sphere = new THREE.Mesh(sphereGeometry, sphereMaterial);
        sphere.castShadow = true;
sphere.position.x = 12;
sphere.position.y = 4;
sphere.position.z = 5;
scene.add(sphere);

        // Add a Small sphere 2
var sphereGeometry = new THREE.SphereGeometry(1, 20, 20);
var sphereMaterial = new THREE.MeshLambertMaterial({color: 0x5787D2, wireframe: false});
var sphere = new THREE.Mesh(sphereGeometry, sphereMaterial);
        sphere.castShadow = true;
sphere.position.x = -12;
sphere.position.y = 4;
sphere.position.z = -5;
scene.add(sphere);

        //GUI
var gui = new dat.GUI();
gui.add(controls, 'rotationSpeed', 0, 0.5);
gui.add(controls, 'addCube');
gui.add(controls, 'removeCube');
gui.add(controls, 'outputObjects');
gui.add(controls, 'numberOfObjects').listen();
```

```
function render() {
    stats.update();

    // rotate the cubes around its axes
    scene.traverse(function (e) {
        if (e instanceof THREE.Mesh && e != plane) {

            e.rotation.x += controls.rotationSpeed;
            e.rotation.y += controls.rotationSpeed;
            e.rotation.z += controls.rotationSpeed;
        }
    });

    // render using requestAnimationFrame
    requestAnimationFrame(render);
    renderer.render(scene, camera);
}
        render();

        document.getElementById('button').onclick = function() {
                var cordx = document.getElementById('x').value;
                var cordy = document.getElementById('y').value;
                var cordz = document.getElementById('z').value;
```

```
                camera.position.x = cordx;
                camera.position.y = cordy;
                camera.position.z = cordz;
                render();

                }

}
window.onload = init;
```

Could not get the camera to reposition