Advanced Graphics

Lab 2 – Using dat.gui.min.js

Due: At the end of the lab (demonstration only)

> To prevent compatibility issues in marking and versioning, we will only use r100 of the three.js library

Objective for this lab:

- To comfortable creating a threejs application.
- To appreciate the convenience of using the dat.gui library to manipulate variables.
- To allow you to interact with your application.
- Do all the assigned problems on your own

The workflow for all of the labs in this will comprise of the following:

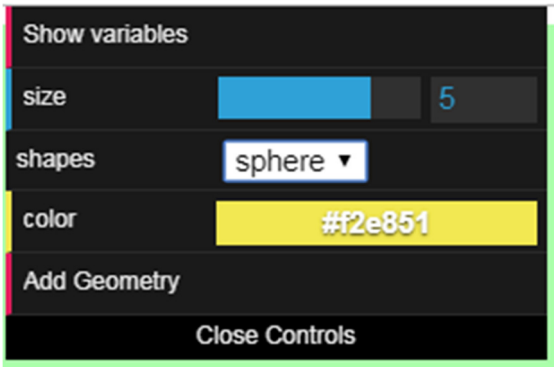
1. Create an appropriate folder structure for VS Code.
2. Add the necessary javascript libraries to the html page
3. Code the required javascript statements to complete the lab is a separate javascript file

In future, we will be using VS Code as our code editor. For now hosting will not be a problem because additional content will not be used. When that time come we will figure out something.

Tasks:

1 Marks

Using the base supplied by the instructor (see the appendix of this document), add dat.gui widget with the following:



It is reccommended to code this part of the solution in a single method and then call it from the `window.onload` handler.

1 Mark

1. `Show variables`: this will be a function to output to the console the value for size, shapes and color. your own color

1 Marks

2. `size`: this is a number value in the range of 2 to 6 in increments of 1.

1 Marks 3. `shapes`: this is a drop down list with two values `"cube"` and `"sphere"`.

1 Marks 4. `color`: this is a color picker.

5 Marks 5. `Add Geometry`: this is a function that creates an object that is specified by the three previous widgets and add it to the scene.

   1 Mark a. Correct object

   1 Mark b. Correct size

   1 Mark c. Correct color

   1 Mark d. Correct position. You may use the following to calculate the range of x
   `plane.geometry.parameters.width`
   You may use a similar technique to get the z range

`https://workshop.chromeexperiments.com/examples/gui/#1--Basic-Usage`

Appendices:

Html code

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="utf-8" />
 <title>COMP392: Lab 2 - using dat gui</title>
 <link rel="stylesheet" href="app.css" type="text/css" />
 <script src="../libs/three.min.js"></script>         <!--or the correct path to library-->
 <script src="../libs/dat.gui.min.js"></script>       <!--or the correct path to library-->
 <script src="../libs/TrackballControls.js"></script> <!--or the correct path to library-->
 <script src="02-lab-dat-gui.js"></script>
</head>
<body>
</body>
</html>
```

N.B. Unless instructed otherwise, this will be the structure of your html file

Javascript code

```
//declare recurrent global variables
const scene = new THREE.Scene();
var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);
const renderer = new THREE.WebGLRenderer({ antialias: true });
const clock = new THREE.Clock();


//declare global variables
var trackballControl,
    controls,
    plane; //this will be used to limit the position of the new objects


function init() {

    renderer.setClearColor(new THREE.Color(0xaaffaa));
    renderer.setSize(window.innerWidth, window.innerHeight);
    document.body.appendChild(renderer.domElement);

    camera.position.x = -30;
    camera.position.y = 40;
    camera.position.z = 30;
    camera.lookAt(scene.position);

    trackballControl = new THREE.TrackballControls(camera, renderer.domElement);

    let geom = new THREE.BoxGeometry(2, 2, 2);
    let mat = new THREE.MeshStandardMaterial({ color: 0xaaaaff });
    let mesh = new THREE.Mesh(geom, mat);
    mesh.position.y = 1;
    scene.add(mesh);

}

function createCameraAndLights() {

    camera.position.x = 120;
    camera.position.y = 60;
    camera.position.z = 80;
    camera.lookAt(scene.position);

    // add subtle ambient lighting
    var ambientLight = new THREE.AmbientLight(0x292929);
    scene.add(ambientLight);

    var directionalLight = new THREE.DirectionalLight(0xffffff, 0.7);
    directionalLight.position.set(-20, 40, 60);
    scene.add(directionalLight);

}

function createGeometry() {

    // create the ground plane
    var planeGeometry = new THREE.PlaneBufferGeometry(60, 20);
    var planeMaterial = new THREE.MeshBasicMaterial({ color: 0xcccccc });
    plane = new THREE.Mesh(planeGeometry, planeMaterial);

    // rotate and position the plane
    plane.rotation.x = -0.5 * Math.PI;
    plane.position.set(15, 0, 0);

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

}
```

```
function animate() {

    trackballControl.update(clock.getDelta());
    renderer.render(scene, camera);

    requestAnimationFrame(animate);

}

//javascript function to drive your scene
window.onload = function () {

    init();
    createCameraAndLights();
    createGeometry();
    animate();

}
```