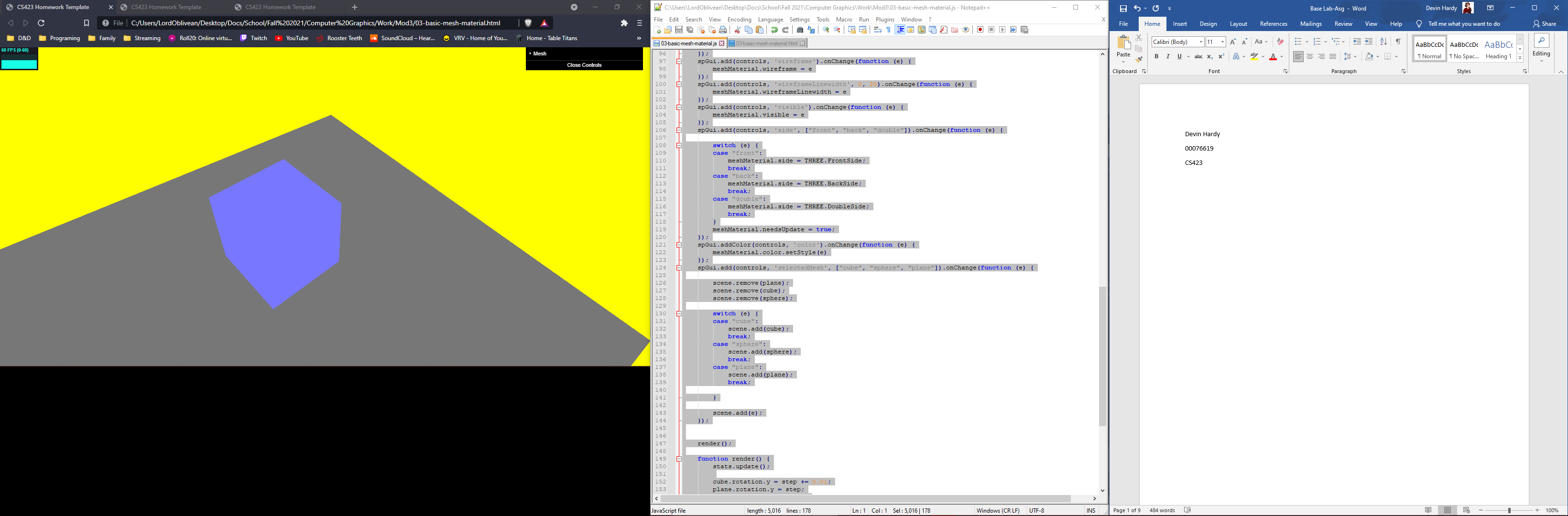
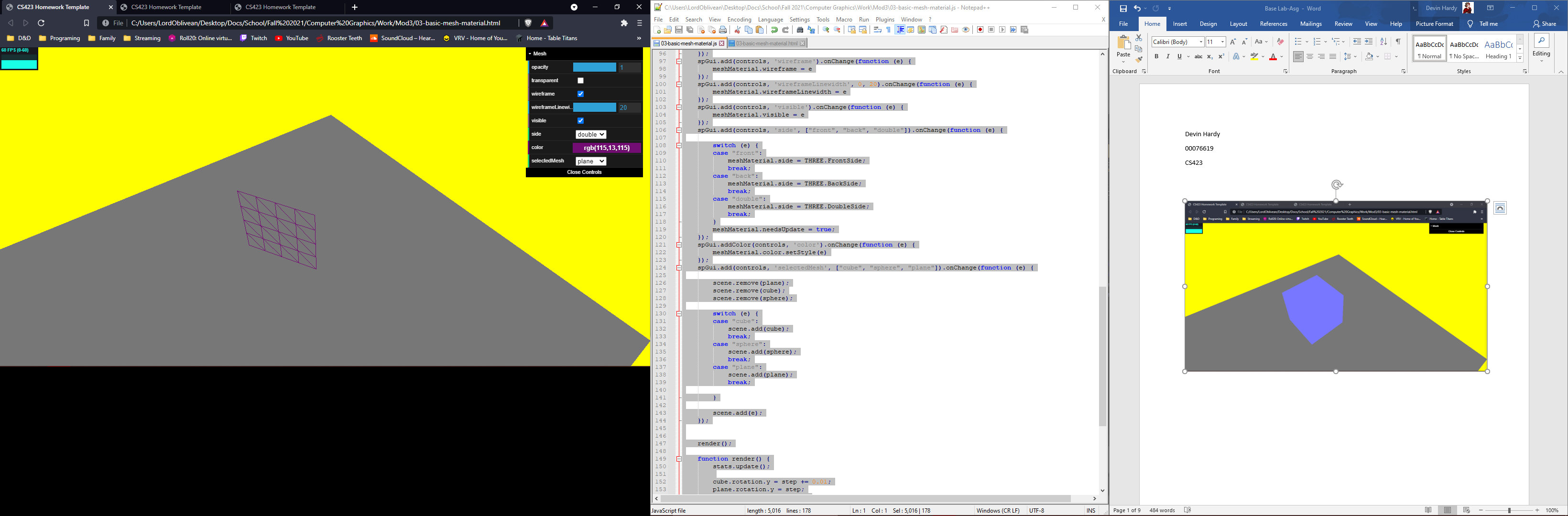
Devin Hardy

00076619

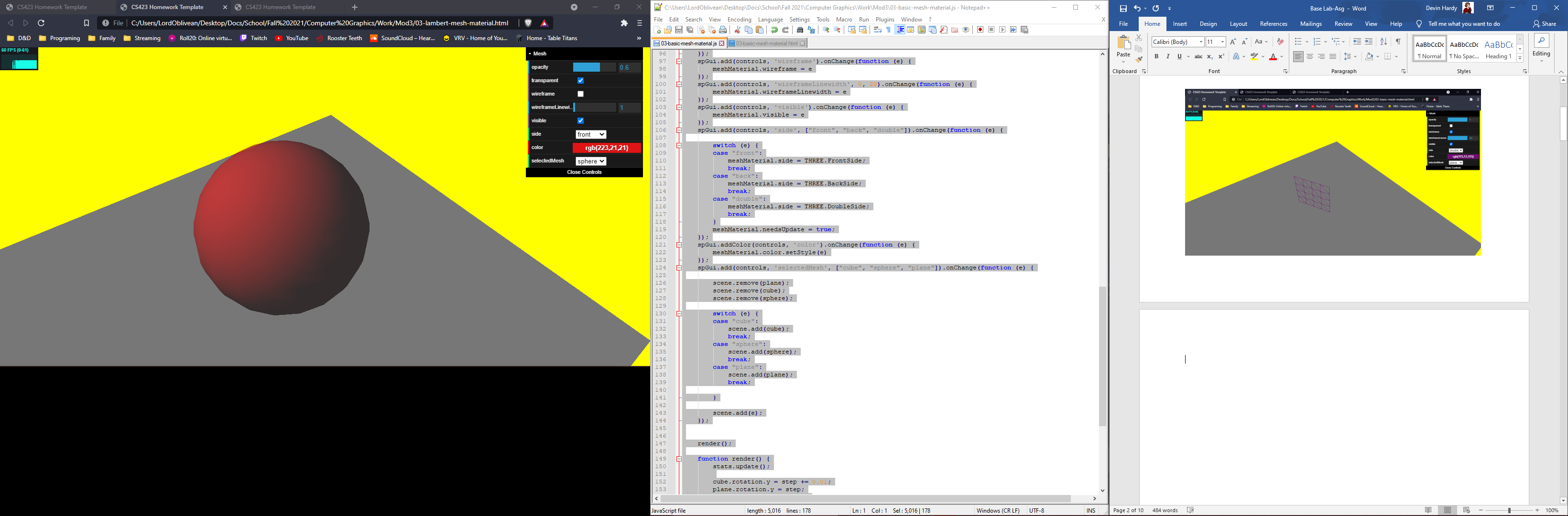
CS423

Basic

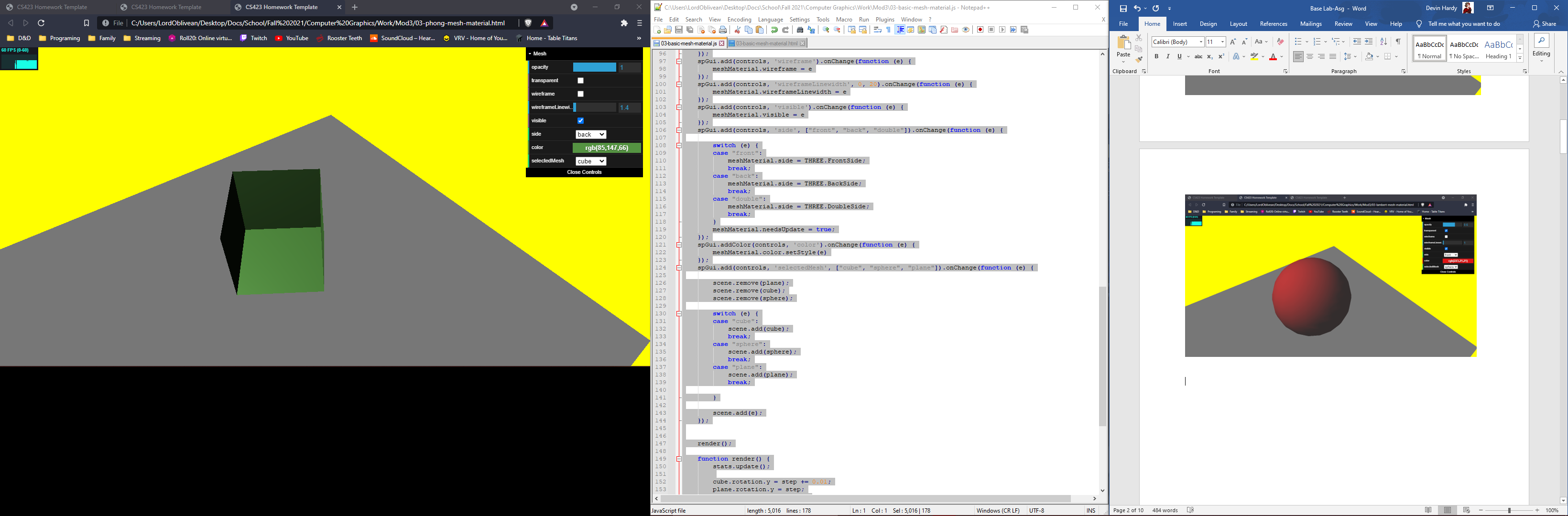




Lambert



Phong



**Basic**

<!DOCTYPE html>

<HTML>

<HEAD>

<TITLE>CS423 Homework Template</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 {

margin: 0;

overflow: hidden;

}

</STYLE>

</HEAD>

<BODY>

<DIV ID="Stats-output">

</DIV>

<DIV ID="WebGL-output">

</DIV>

<!-- Scripts that we use for running things -->

<SCRIPT TYPE="text/javascript" SRC="03-basic-mesh-material.js"></SCRIPT>

</BODY>

</HTML>

function init() {

var stats = initStats();

// 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.

var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);

// 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;

var groundGeom = new THREE.PlaneGeometry(100, 100, 4, 4);

var groundMesh = new THREE.Mesh(groundGeom, new THREE.MeshBasicMaterial({color: 0x777777}));

groundMesh.rotation.x = -Math.PI / 2;

groundMesh.position.y = -20;

scene.add(groundMesh);

var sphereGeometry = new THREE.SphereGeometry(14, 20 ,20);

var cubeGeometry = new THREE.BoxGeometry(15, 15, 15);

var planeGeometry = new THREE.PlaneGeometry(14, 14, 4, 4);

var meshMaterial = new THREE.MeshBasicMaterial({color: 0x7777ff});

var sphere = new THREE.Mesh(sphereGeometry, meshMaterial);

var cube = new THREE.Mesh(cubeGeometry, meshMaterial);

var plane = new THREE.Mesh(planeGeometry, meshMaterial);

sphere.position.x = 0;

sphere.position.y = 3;

sphere.position.z = 2;

cube.position = sphere.position;

plane.position = sphere.position;

scene.add(cube);

camera.position.x = -20;

camera.position.y = 50;

camera. position.z = 40;

camera.lookAt(new THREE.Vector3(10, 0, 0));

var ambientLight = new THREE.AmbientLight(0x0c0c0c);

scene.add(ambientLight);

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);

// call the render function

var step = 0;

var oldContext = null;

var controls = new function () {

this.rotationSpeed = 0.02;

this.bouncingSpeed = 0.03;

this.opacity = meshMaterial.opacity;

this.transparent = meshMaterial.transparent;

this.visible = meshMaterial.visible;

this.side = "front";

this.color = meshMaterial.color.getStyle();

this.wireframe = meshMaterial.wireframe;

this.wireframeLinewidth = meshMaterial.wireframeLinewidth;

this.wireframeLineJoin = meshMaterial.wireframeLinejoin;

this.selectedMesh = "cube";

};

var gui = new dat.GUI();

var spGui = gui.addFolder("Mesh");

spGui.add(controls, 'opacity', 0, 1).onChange(function (e) {

meshMaterial.opacity = e

});

spGui.add(controls, 'transparent').onChange(function (e) {

meshMaterial.transparent = e

});

spGui.add(controls, 'wireframe').onChange(function (e) {

meshMaterial.wireframe = e

});

spGui.add(controls, 'wireframeLinewidth', 0, 20).onChange(function (e) {

meshMaterial.wireframeLinewidth = e

});

spGui.add(controls, 'visible').onChange(function (e) {

meshMaterial.visible = e

});

spGui.add(controls, 'side', ["front", "back", "double"]).onChange(function (e) {

switch (e) {

case "front":

meshMaterial.side = THREE.FrontSide;

break;

case "back":

meshMaterial.side = THREE.BackSide;

break;

case "double":

meshMaterial.side = THREE.DoubleSide;

break;

}

meshMaterial.needsUpdate = true;

});

spGui.addColor(controls, 'color').onChange(function (e) {

meshMaterial.color.setStyle(e)

});

spGui.add(controls, 'selectedMesh', ["cube", "sphere", "plane"]).onChange(function (e) {

scene.remove(plane);

scene.remove(cube);

scene.remove(sphere);

switch (e) {

case "cube":

scene.add(cube);

break;

case "sphere":

scene.add(sphere);

break;

case "plane":

scene.add(plane);

break;

}

scene.add(e);

});

render();

function render() {

stats.update();

cube.rotation.y = step += 0.01;

plane.rotation.y = step;

sphere.rotation.y = step;

// render using requestAnimationFrame

requestAnimationFrame(render);

renderer.render(scene, camera);

}

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;

}

}

window.onload = init

**Lambert**

<!DOCTYPE html>

<HTML>

<HEAD>

<TITLE>CS423 Homework Template</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 {

margin: 0;

overflow: hidden;

}

</STYLE>

</HEAD>

<BODY>

<DIV ID="Stats-output">

</DIV>

<DIV ID="WebGL-output">

</DIV>

<!-- Scripts that we use for running things -->

<SCRIPT TYPE="text/javascript" SRC="03-lambert-mesh-material.js"></SCRIPT>

</BODY>

</HTML>

function init() {

var stats = initStats();

// 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.

var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);

// 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;

var groundGeom = new THREE.PlaneGeometry(100, 100, 4, 4);

var groundMesh = new THREE.Mesh(groundGeom, new THREE.MeshBasicMaterial({color: 0x777777}));

groundMesh.rotation.x = -Math.PI / 2;

groundMesh.position.y = -20;

scene.add(groundMesh);

var sphereGeometry = new THREE.SphereGeometry(14, 20 ,20);

var cubeGeometry = new THREE.BoxGeometry(15, 15, 15);

var planeGeometry = new THREE.PlaneGeometry(14, 14, 4, 4);

var meshMaterial = new THREE.MeshLambertMaterial({color: 0x7777ff});

var sphere = new THREE.Mesh(sphereGeometry, meshMaterial);

var cube = new THREE.Mesh(cubeGeometry, meshMaterial);

var plane = new THREE.Mesh(planeGeometry, meshMaterial);

sphere.position.x = 0;

sphere.position.y = 3;

sphere.position.z = 2;

cube.position = sphere.position;

plane.position = sphere.position;

scene.add(cube);

camera.position.x = -20;

camera.position.y = 50;

camera. position.z = 40;

camera.lookAt(new THREE.Vector3(10, 0, 0));

var ambientLight = new THREE.AmbientLight(0x0c0c0c);

scene.add(ambientLight);

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);

// call the render function

var step = 0;

var oldContext = null;

var controls = new function () {

this.rotationSpeed = 0.02;

this.bouncingSpeed = 0.03;

this.opacity = meshMaterial.opacity;

this.transparent = meshMaterial.transparent;

this.visible = meshMaterial.visible;

this.side = "front";

this.color = meshMaterial.color.getStyle();

this.wireframe = meshMaterial.wireframe;

this.wireframeLinewidth = meshMaterial.wireframeLinewidth;

this.wireframeLineJoin = meshMaterial.wireframeLinejoin;

this.selectedMesh = "cube";

};

var gui = new dat.GUI();

var spGui = gui.addFolder("Mesh");

spGui.add(controls, 'opacity', 0, 1).onChange(function (e) {

meshMaterial.opacity = e

});

spGui.add(controls, 'transparent').onChange(function (e) {

meshMaterial.transparent = e

});

spGui.add(controls, 'wireframe').onChange(function (e) {

meshMaterial.wireframe = e

});

spGui.add(controls, 'wireframeLinewidth', 0, 20).onChange(function (e) {

meshMaterial.wireframeLinewidth = e

});

spGui.add(controls, 'visible').onChange(function (e) {

meshMaterial.visible = e

});

spGui.add(controls, 'side', ["front", "back", "double"]).onChange(function (e) {

switch (e) {

case "front":

meshMaterial.side = THREE.FrontSide;

break;

case "back":

meshMaterial.side = THREE.BackSide;

break;

case "double":

meshMaterial.side = THREE.DoubleSide;

break;

}

meshMaterial.needsUpdate = true;

});

spGui.addColor(controls, 'color').onChange(function (e) {

meshMaterial.color.setStyle(e)

});

spGui.add(controls, 'selectedMesh', ["cube", "sphere", "plane"]).onChange(function (e) {

scene.remove(plane);

scene.remove(cube);

scene.remove(sphere);

switch (e) {

case "cube":

scene.add(cube);

break;

case "sphere":

scene.add(sphere);

break;

case "plane":

scene.add(plane);

break;

}

scene.add(e);

});

render();

function render() {

stats.update();

cube.rotation.y = step += 0.01;

plane.rotation.y = step;

sphere.rotation.y = step;

// render using requestAnimationFrame

requestAnimationFrame(render);

renderer.render(scene, camera);

}

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;

}

}

window.onload = init

**Phong**

<!DOCTYPE html>

<HTML>

<HEAD>

<TITLE>CS423 Homework Template</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 {

margin: 0;

overflow: hidden;

}

</STYLE>

</HEAD>

<BODY>

<DIV ID="Stats-output">

</DIV>

<DIV ID="WebGL-output">

</DIV>

<!-- Scripts that we use for running things -->

<SCRIPT TYPE="text/javascript" SRC="03-phong-mesh-material.js"></SCRIPT>

</BODY>

</HTML>

function init() {

var stats = initStats();

// 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.

var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);

// 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;

var groundGeom = new THREE.PlaneGeometry(100, 100, 4, 4);

var groundMesh = new THREE.Mesh(groundGeom, new THREE.MeshBasicMaterial({color: 0x777777}));

groundMesh.rotation.x = -Math.PI / 2;

groundMesh.position.y = -20;

scene.add(groundMesh);

var sphereGeometry = new THREE.SphereGeometry(14, 20 ,20);

var cubeGeometry = new THREE.BoxGeometry(15, 15, 15);

var planeGeometry = new THREE.PlaneGeometry(14, 14, 4, 4);

var meshMaterial = new THREE.MeshPhongMaterial({color: 0x7777ff});

var sphere = new THREE.Mesh(sphereGeometry, meshMaterial);

var cube = new THREE.Mesh(cubeGeometry, meshMaterial);

var plane = new THREE.Mesh(planeGeometry, meshMaterial);

sphere.position.x = 0;

sphere.position.y = 3;

sphere.position.z = 2;

cube.position = sphere.position;

plane.position = sphere.position;

scene.add(cube);

camera.position.x = -20;

camera.position.y = 50;

camera. position.z = 40;

camera.lookAt(new THREE.Vector3(10, 0, 0));

var ambientLight = new THREE.AmbientLight(0x0c0c0c);

scene.add(ambientLight);

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);

// call the render function

var step = 0;

var oldContext = null;

var controls = new function () {

this.rotationSpeed = 0.02;

this.bouncingSpeed = 0.03;

this.opacity = meshMaterial.opacity;

this.transparent = meshMaterial.transparent;

this.visible = meshMaterial.visible;

this.side = "front";

this.color = meshMaterial.color.getStyle();

this.wireframe = meshMaterial.wireframe;

this.wireframeLinewidth = meshMaterial.wireframeLinewidth;

this.wireframeLineJoin = meshMaterial.wireframeLinejoin;

this.selectedMesh = "cube";

};

var gui = new dat.GUI();

var spGui = gui.addFolder("Mesh");

spGui.add(controls, 'opacity', 0, 1).onChange(function (e) {

meshMaterial.opacity = e

});

spGui.add(controls, 'transparent').onChange(function (e) {

meshMaterial.transparent = e

});

spGui.add(controls, 'wireframe').onChange(function (e) {

meshMaterial.wireframe = e

});

spGui.add(controls, 'wireframeLinewidth', 0, 20).onChange(function (e) {

meshMaterial.wireframeLinewidth = e

});

spGui.add(controls, 'visible').onChange(function (e) {

meshMaterial.visible = e

});

spGui.add(controls, 'side', ["front", "back", "double"]).onChange(function (e) {

switch (e) {

case "front":

meshMaterial.side = THREE.FrontSide;

break;

case "back":

meshMaterial.side = THREE.BackSide;

break;

case "double":

meshMaterial.side = THREE.DoubleSide;

break;

}

meshMaterial.needsUpdate = true;

});

spGui.addColor(controls, 'color').onChange(function (e) {

meshMaterial.color.setStyle(e)

});

spGui.add(controls, 'selectedMesh', ["cube", "sphere", "plane"]).onChange(function (e) {

scene.remove(plane);

scene.remove(cube);

scene.remove(sphere);

switch (e) {

case "cube":

scene.add(cube);

break;

case "sphere":

scene.add(sphere);

break;

case "plane":

scene.add(plane);

break;

}

scene.add(e);

});

render();

function render() {

stats.update();

cube.rotation.y = step += 0.01;

plane.rotation.y = step;

sphere.rotation.y = step;

// render using requestAnimationFrame

requestAnimationFrame(render);

renderer.render(scene, camera);

}

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;

}

}

window.onload = init