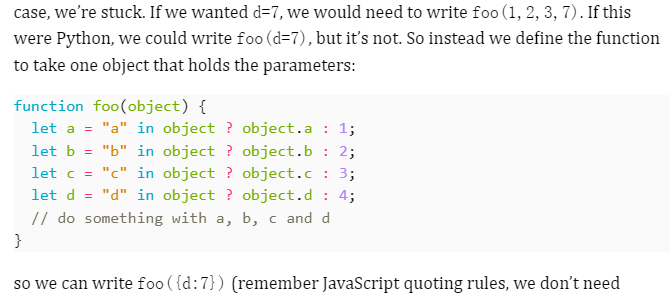
**workbook6**



光：<https://blog.csdn.net/qq_40712151/article/details/105339335>

**一、THREE.AmbientLight环境光**

THREE.AmbientLight创建的光源，颜色会被应用至全局。该光源没有特定来源方向，并且不会产生阴影。通常不会单独使用它，而是配合其他光源同时使用，目的是弱化阴影或给场景添加一些额外的颜色。

  创建环境光非常简单，只需new THREE.AmbientLight(),然后将其添加到场景中即可，无需设置光照位置

**环境光会均匀的照亮场景中的所有物体。**

**var light = new THREE.AmbientLight( 0x404040 ，1); // 白光，灯光强度为1；**

1. **THREE.PointLight点光源**

点光源。一种单点发光、照射所有方向的光源。例如蜡烛。

**var light = new THREE.PointLight(颜色，强度，距离)**

2.1 THREE.PointLight属性





1. **THREE.SpotLight聚光**

**var light = new THREE.new THREE.SpotLight(颜色, 强度, 距离, 夹角, 衰减指数);**

聚光灯(具有锥形效果的光源) 最常用的光源。锥形效果的光源，例如手电筒。

**跟点光源差别不大,差别是castShadow设置为true可以有阴影**

可以只传入颜色参数

设置target属性可以决定光源对准某个物体照射的对象(即使对象在移动,也会对准)

如果不想对准某个物体而是指定的某一点可以创建一个空的THREE.Object3D()实例如下:

var target = new THREE.Object3D();

target.position = new THREE.Vector3(5,0,0);

SpotLight.target = target; //设置光源的target属性

3.1 THREE.SpotLight属性

****

1. **THREE.DirectionalLight**

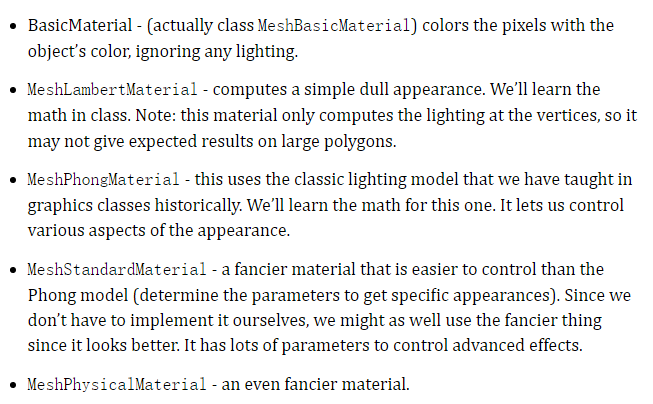
平行光源，常用来模拟太阳光。由特点方向发出的光，产生的光线都是平行的，太阳光可近似看为平行光。

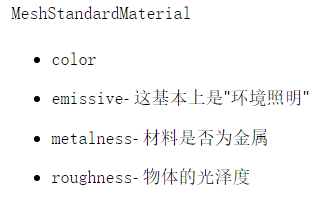
**var light = new THREE.DirectionalLight(颜色，亮度)；**

4.1THREE.DirectionalLight的属性



**Three的Material**





**Box 06-02-02 1 pt**

**move the yellow cube**

**Box 06-04-03 5 pt**

**stacking the boxes as described**

**Box 06-05-03 4 pt**

**add lights to scene so that sides of the cubes appear in different colors**

Box 06-06-01 5 pt

give each sphere a different, non-white material (1/2 pt each, 1/2 pt for all different)

Box 06-08-01 3 pt

Visible Objects (requires lights, camera, at least some shapes)

Box 06-08-01 2 pt

Different materials used (everything isn't the same color

Box 06-08-01 3 pt

There is a ground, Snowman on ground

Box 06-08-01 4 pt

Snowman Body as 3 stacked spheres

Box 06-08-01 4 pt

Snowman Face (eyes, nose, mouth

Box 06-09-01 2 pt

Museum cameras work (1/2 pt per camera)

Box 06-09-01 3 pt

Museum objects (1 pt for each object that is more than a simple primitive)

Box 06-09-01 3 pt

Museum objects correctly placed (1 pt per new object)

Box 06-09-01 3 pt

Museum objects move correctly (1 pt per new object), not all the same

Box 06-09-01 2 pt

Museum objects each have spotlights (1/2 pt per object)

Box 06-09-01 2 pt

Each new object has its own material

Advanced points (9 possible, of which you can earn a maximum of 7):

Box 06-08-01 2 pt

Animated snowman

Box 06-08-01 3 pt

Second, creative snowman

Box 06-09-01 2 pt

At least one object has articulated motion

Box 06-09-01 1 pt

Artistic merit: Cool museum objects (grader's discretion)

Box 06-09-01 1 pt

Artistic merit: Cool museum object motions (grader's discretion)

// @ts-check

import \* as T from "../libs/CS559-Three/build/three.module.js";

import { OrbitControls } from "../libs/CS559-Three/examples/jsm/controls/OrbitControls.js";

let renderer = new T.WebGLRenderer();

renderer.setSize(500, 500);

document.getElementById("div1").appendChild(renderer.domElement);

// student does the rest.

let scene = new T.Scene();

let camera = new T.PerspectiveCamera();

camera.position.z = 10;

camera.position.y = 5;

camera.position.x = 5;

camera.lookAt(0, 3, 0);

// since we're animating, add OrbitControls

let controls = new OrbitControls(camera, renderer.domElement);

// make lights

scene.add(new T.AmbientLight("white", 0.2));

let point1 = new T.PointLight("white", 1, 0, 0);

point1.position.set(20, 10, 15);

scene.add(point1);

let point2 = new T.PointLight("white", 1, 0, 0);

point2.position.set(-20, -10, 15);

scene.add(point2);

// make a ground plane

let groundBox = new T.BoxGeometry(10, 0.2, 10);

let groundMesh = new T.Mesh(groundBox, new T.MeshLambertMaterial({ color: 0x888888 }));

groundMesh.position.y = -0.1;

scene.add(groundMesh);

// snowman1

// make 2 bodies for the snowman1

let bodyMaterials1 = new T.MeshStandardMaterial();

let bodyGeometry1 = new T.SphereBufferGeometry(0.9, 20, 20);

let body1 = new T.Mesh(bodyGeometry1, bodyMaterials1);

body1.position.set(-0.3, 1, 4);

scene.add(body1);

let bodyMaterials2 = new T.MeshStandardMaterial();

let bodyGeometry2 = new T.SphereBufferGeometry(0.7, 20, 20);

let body2 = new T.Mesh(bodyGeometry2, bodyMaterials2);

body2.position.set(-0.3, 2.6, 4);

scene.add(body2);

//make head

let headMaterials = new T.MeshStandardMaterial();

let headGeometry = new T.SphereBufferGeometry(0.5, 10, 10);

let head = new T.Mesh(headGeometry, headMaterials);

head.position.set(-0.3, 3.8, 4);

scene.add(head);

//make face(eyes, nose, mouth)

//eyes

let eyeMaterials1 = new T.MeshStandardMaterial({color:"black"});

let eyeGeometry1 = new T.SphereBufferGeometry(0.05, 10, 10);

let eye1 = new T.Mesh(eyeGeometry1, eyeMaterials1);

eye1.position.set(-0.1, 3.8, 4.5);

scene.add(eye1);

let eyeMaterials2 = new T.MeshStandardMaterial({color:"black"});

let eyeGeometry2 = new T.SphereBufferGeometry(0.05, 10, 10);

let eye2 = new T.Mesh(eyeGeometry2, eyeMaterials2);

eye2.position.set(-0.5, 3.8, 4.5);

scene.add(eye2);

//mouth

let mouthMaterials = new T.MeshStandardMaterial({color:"red"});

let mouthGeometry = new T.BoxGeometry(0.2,0,0.2);

let mouth = new T.Mesh(mouthGeometry, mouthMaterials);

mouth.position.set(-0.3, 3.55, 4.43);

scene.add(mouth);

//nose

let noseGeometry = new T.ConeGeometry(0.05, 0.3);

let noseMaterial = new T.MeshStandardMaterial({color:"orange"});

let nose = new T.Mesh(noseGeometry, noseMaterial);

nose.position.set(-0.3, 3.7, 4.7);

nose.rotateX(Math.PI/2);

scene.add(nose);

body1.position.y -= 0.25;

body2.position.y -= 0.25;

head.position.y -= 0.25;

eye1.position.y -= 0.25;

eye2.position.y -= 0.25;

nose.position.y -= 0.25;

mouth.position.y -= 0.25;

// snowman2

// make 2 bodies for the snowman2

let bodyMaterials1ForMan2 = new T.MeshStandardMaterial();

let bodyGeometry1ForMan2 = new T.SphereBufferGeometry(0.9, 20, 20);

let body1ForMan2 = new T.Mesh(bodyGeometry1ForMan2, bodyMaterials1ForMan2);

body1ForMan2.position.set(3, 1, 4);

scene.add(body1ForMan2);

let bodyMaterials2ForMan2 = new T.MeshStandardMaterial();

let bodyGeometry2ForMan2 = new T.SphereBufferGeometry(0.7, 20, 20);

let body2ForMan2 = new T.Mesh(bodyGeometry2ForMan2, bodyMaterials2ForMan2);

body2ForMan2.position.set(3, 2.6, 4);

scene.add(body2ForMan2);

//make head

let headMaterialsForMan2 = new T.MeshStandardMaterial();

let headGeometryForMan2 = new T.SphereBufferGeometry(0.5, 10, 10);

let headForMan2 = new T.Mesh(headGeometryForMan2, headMaterialsForMan2);

headForMan2.position.set(3, 3.8, 4);

scene.add(headForMan2);

//make face(eyes, nose, mouth)

//eyes

let eyeMaterials1ForMan2 = new T.MeshStandardMaterial({color:"black"});

let eyeGeometry1ForMan2 = new T.SphereBufferGeometry(0.05, 10, 10);

let eye1ForMan2 = new T.Mesh(eyeGeometry1ForMan2, eyeMaterials1ForMan2);

eye1ForMan2.position.set(3.2, 3.8, 4.5);

scene.add(eye1ForMan2);

let eyeMaterials2ForMan2 = new T.MeshStandardMaterial({color:"black"});

let eyeGeometry2ForMan2 = new T.SphereBufferGeometry(0.05, 10, 10);

let eye2ForMan2 = new T.Mesh(eyeGeometry2ForMan2, eyeMaterials2ForMan2);

eye2ForMan2.position.set(2.8, 3.8, 4.5);

scene.add(eye2ForMan2);

//mouth

let mouthMaterialsForMan2 = new T.MeshStandardMaterial({color:"red"});

let mouthGeometryForMan2 = new T.BoxGeometry(0.2,0,0.2);

let mouthForMan2 = new T.Mesh(mouthGeometryForMan2, mouthMaterialsForMan2);

mouthForMan2.position.set(3, 3.55, 4.43);

scene.add(mouthForMan2);

//nose

let noseGeometryForMan2 = new T.ConeGeometry(0.05, 0.3);

let noseMaterialForMan2 = new T.MeshStandardMaterial({color:"orange"});

let noseForMan2 = new T.Mesh(noseGeometryForMan2, noseMaterialForMan2);

noseForMan2.position.set(3, 3.7, 4.7);

noseForMan2.rotateX(Math.PI/2);

scene.add(noseForMan2);

let lastTimestamp; // undefined to start

function animate(timestamp) {

// Convert time change from milliseconds to seconds

lastTimestamp = timestamp;

console.log(timestamp);

// Slide the cube back and forth over the course of a second

let t\_x = (0.001 \* timestamp) % 1.0;

let x = Math.cos(Math.PI \* t\_x);

if (x < 0) x \*= -1;

body1.position.y += x;

body2.position.y += x;

head.position.y += x;

eye1.position.y += x;

eye2.position.y += x;

nose.position.y += x;

mouth.position.y += x;

renderer.render(scene, camera);

body1.position.y -= x;

body2.position.y -= x;

head.position.y -= x;

eye1.position.y -= x;

eye2.position.y -= x;

nose.position.y -= x;

mouth.position.y -= x;

window.requestAnimationFrame(animate);

}

window.requestAnimationFrame(animate);

renderer.render(scene, camera);

// @ts-check

import \* as T from "../libs/CS559-Three/build/three.module.js";

import { Vector3 } from "../libs/CS559-Three/build/three.module.js";

import { OrbitControls } from "../libs/CS559-Three/examples/jsm/controls/OrbitControls.js";

import { setupBasicScene } from "./06-09-01-helpers.js";

// students can use the object loader

// uncomment this if necessary

// import { OBJLoader } from "../libs/CS559-Three/examples/jsm/loaders/OBJLoader.js";

/\*\* Setup the window \*/

/\*\* @type{number} \*/

let wid = 670; // window.innerWidth;

/\*\* @type{number} \*/

let ht = 500; // window.innerHeight;

/\*\* @type{T.WebGLRenderer} \*/

let renderer = new T.WebGLRenderer();

renderer.setSize(wid, ht);

renderer.shadowMap.enabled = true;

document.getElementById("museum\_area").appendChild(renderer.domElement);

/\* setupBasicScene creates a scene and puts the pedestals in place \*/

/\*\* @type{T.Scene} \*/

let scene = setupBasicScene();

// Here, we add a basic, simple first object to the museum.

/\*\*@type{T.Material} \*/

let material = new T.MeshPhongMaterial({

color: "#00aa00",

shininess: 15,

specular: "#00ff00",

});

/\*\*@type{T.BufferGeometry} \*/

let geometry = new T.BoxGeometry(0.5, 0.5, 0.5);

/\*\*@type{T.Mesh} \*/

let cube = new T.Mesh(geometry, material);

cube.position.set(2, 1.35, 2);

cube.rotation.set(Math.PI / 4, 0, Math.PI / 4);

cube.castShadow = true;

// TODO: You need to create three more objects, and place them on pedestals.

// Objetc1: Robot with moving hand

// Object2: Jumping balls

let ballMaterials1 = new T.MeshStandardMaterial();

let ballGeometry1 = new T.SphereBufferGeometry(0.4, 20, 20);

let ball1 = new T.Mesh(ballGeometry1, ballMaterials1);

ball1.position.set(2, 1.46, -2);

scene.add(ball1);

let ballMaterials2 = new T.MeshStandardMaterial();

let ballGeometry2 = new T.SphereBufferGeometry(0.3, 20, 20);

let ball2 = new T.Mesh(ballGeometry2, ballMaterials2);

ball2.position.set(2, 2.06, -2);

scene.add(ball2);

let ballMaterials3 = new T.MeshStandardMaterial();

let ballGeometry3 = new T.SphereBufferGeometry(0.2, 20, 20);

let ball3 = new T.Mesh(ballGeometry3, ballMaterials3);

ball3.position.set(2, 2.46, -2);

scene.add(ball3);

// Object3: Quadcopter

let planeMaterials = new T.MeshStandardMaterial({color:"yellow"});

let planeGeometry = new T.BoxBufferGeometry(1,0.5,0.7);

let plane = new T.Mesh(planeGeometry, planeMaterials);

plane.position.set(-2, 1.5, -2);

plane.rotateY(-Math.PI/4);

scene.add(plane);

let stickMaterials = new T.MeshStandardMaterial({color:"green"});

let stickGeometry = new T.BoxBufferGeometry(0.1,0.3,0.1);

let stick = new T.Mesh(stickGeometry, stickMaterials);

stick.position.set(-2, 1.9, -2);

stick.rotateY(-Math.PI/4);

scene.add(stick);

let rotorMaterials1 = new T.MeshStandardMaterial({color:"red"});

let rotorGeometry1 = new T.BoxBufferGeometry(1.3,0.1,0.1);

let rotor1 = new T.Mesh(rotorGeometry1, rotorMaterials1);

rotor1.position.set(-2, 2.1, -2);

rotor1.rotateY(-Math.PI/4);

scene.add(rotor1);

let rotorMaterials2 = new T.MeshStandardMaterial({color:"red"});

let rotorGeometry2 = new T.BoxBufferGeometry(1.3,0.1,0.1);

let rotor2 = new T.Mesh(rotorGeometry2, rotorMaterials2);

rotor2.position.set(-2, 2.1, -2);

rotor2.rotateY(Math.PI/4);

scene.add(rotor2);

/\* put a spotlight on the first object \*/

/\*\*@type{T.SpotLight} \*/

let spotlight\_1 = new T.SpotLight(0xaaaaff, 0.5);

spotlight\_1.angle = Math.PI / 16;

spotlight\_1.position.set(2, 5, 2);

spotlight\_1.target = cube;

spotlight\_1.castShadow = true;

scene.add(spotlight\_1);

// TODO: You need to place the lights.

let spotlight\_2 = new T.SpotLight(0xaaaaff, 0.5);

spotlight\_2.angle = Math.PI / 16;

spotlight\_2.castShadow = true;

spotlight\_2.position.set(-2, 5, 2);

spotlight\_2.target.position.set(-2, 0, 2);

scene.add(spotlight\_2.target);

scene.add(spotlight\_2);

let spotlight\_3 = new T.SpotLight(0xaaaaff, 0.5);

spotlight\_3.angle = Math.PI / 16;

spotlight\_3.castShadow = true;

spotlight\_3.position.set(-2, 5, -2);

spotlight\_3.target.position.set(-2, 0, -2);

scene.add(spotlight\_3.target);

scene.add(spotlight\_3);

let spotlight\_4 = new T.SpotLight(0xaaaaff, 0.5);

spotlight\_4.angle = Math.PI / 16;

spotlight\_4.castShadow = true;

spotlight\_4.position.set(2, 5, -2);

spotlight\_4.target.position.set(2, 0, -2);

scene.add(spotlight\_4.target);

scene.add(spotlight\_4);

/\*\* create a "main camera" \*/

/\*\* @type{T.PerspectiveCamera} \*/

let main\_camera = new T.PerspectiveCamera(60, wid / ht, 1, 100);

main\_camera.position.set(0, 4, 6);

main\_camera.rotation.set(-0.5, 0, 0);

/\*\* this will be the "current camera" - we will switch when a button is pressed \*/

let active\_camera = main\_camera;

// TODO: You need to place these cameras.

let camera\_1 = new T.PerspectiveCamera(60, wid / ht, 1, 100);

camera\_1.position.set(2, 2.6, 4.5);

camera\_1.rotation.set(-0.5, 0, 0);

let camera\_2 = new T.PerspectiveCamera(60, wid / ht, 1, 100);

camera\_2.position.set(-2, 4, 2.3);

camera\_2.rotation.set(-0.5, 0, 0);

let camera\_3 = new T.PerspectiveCamera(60, wid / ht, 1, 100);

camera\_3.position.set(2, 3.5, 2);

camera\_3.rotation.set(-0.5, 0, 0);

let camera\_4 = new T.PerspectiveCamera(60, wid / ht, 1, 100);

camera\_4.position.set(-2, 3.5, 5);

camera\_4.rotation.set(-0.5, 0, 0);

scene.add(cube);

// add orbit controls - but only to the main camera

let controls = new OrbitControls(main\_camera, renderer.domElement);

/\*\* Tie the buttons to the cameras \*/

function setupCamButton(name, camera) {

const button = document.getElementById(name);

if (!(button instanceof HTMLButtonElement))

throw new Error(`Button ${name} doesn't exist`);

button.onclick = function () {

active\_camera = camera;

renderer.render(scene, active\_camera);

};

}

setupCamButton("main\_cam", main\_camera);

setupCamButton("cam\_1", camera\_1);

setupCamButton("cam\_2", camera\_2);

setupCamButton("cam\_3", camera\_3);

setupCamButton("cam\_4", camera\_4);

// finally, draw the scene. Also, add animation.

renderer.render(scene, active\_camera);

let lastTimestamp; // undefined to start

function animate(timestamp) {

// Convert time change from milliseconds to seconds

let timeDelta = 0.001 \* (lastTimestamp ? timestamp - lastTimestamp : 0);

lastTimestamp = timestamp;

// Animate the cube (basic object)

cube.rotateOnWorldAxis(new T.Vector3(0, 1, 0), timeDelta);

// TODO: animate your objects

let t\_x = (0.001 \* timestamp/2.4) % 1.0;

let x = Math.cos(Math.PI \* t\_x);

if (x < 0) x \*= -1;

ball1.position.y += x/2;

ball2.position.y += x/2;

ball3.position.y += x/2;

// draw and loop

renderer.render(scene, active\_camera);

ball1.position.y -= x/2;

ball2.position.y -= x/2;

ball3.position.y -= x/2;

window.requestAnimationFrame(animate);

}

window.requestAnimationFrame(animate);