Three.js

Objekte

Würfel/Quader

*var wurfelgeometry = new THREE.BoxGeometry( 150, 250, 150 );*

Grün: Definieren die Breite/Tiefe

Rot: Definiert die Höche

Kegel/Pyramide

Für alle in eine Spitze verlaufenden Elemente:

*var pyramidGeometry = new THREE.CylinderGeometry(0 ,90, 90, 4, false);*

rot: Definiert, wie viel von der Spitze abgeschnitten ist

Blau: Definiert die Breite

Grün: Definiert die Höhe

Lila: Definiert die Anzahl Seitenflächen   
*var pyramidMaterial = new THREE.MeshBasicMaterial({ color: 0xffffff } );*   
*pyramidMesh = new THREE.Mesh(pyramidGeometry, pyramidMaterial);*

--> Material und Ojekt werden nerstellt

Kugel/Kugelförmige Vielecke

*var earthgeometry = new THREE.SphereGeometry( 160, 3, 4 );*

Rot: Definiert die Grösse

Grün: Definiert die Anzahl Seiten (1x Quer, 1x Hoch)   
*var earthmaterial = new THREE.MeshBasicMaterial( { color: 0xffffff } );*   
*earth = new THREE.Mesh( earthgeometry, earthmaterial );*

Dreieck

Ein Flaches Dreieck erstellen. Dazu kann das Material normal erstellt werden, die Geometrie jedoch folgendermassen:

*var triangleGeometry = new THREE.Geometry();*

*triangleGeometry.vertices.push(new THREE.Vector3( 0.0, 1.0, 0.0));*

*triangleGeometry.vertices.push(new THREE.Vector3(-1.0, -1.0, 0.0));*

*triangleGeometry.vertices.push(new THREE.Vector3( 1.0, -1.0, 0.0));*

*triangleGeometry.faces.push(new THREE.Face3(0, 1, 2));*

Rechteck

Ein Flaches Dreieck erstellen. Dazu kann das Material normal erstellt werden, die Geometrie jedoch folgendermassen:

*var squareGeometry = new THREE.Geometry();*

*squareGeometry.vertices.push(new THREE.Vector3(-1.0, 1.0, 0.0));*

*squareGeometry.vertices.push(new THREE.Vector3( 1.0, 1.0, 0.0));*

*squareGeometry.vertices.push(new THREE.Vector3( 1.0, -1.0, 0.0));*

*squareGeometry.vertices.push(new THREE.Vector3(-1.0, -1.0, 0.0));*

*squareGeometry.faces.push(new THREE.Face3(0, 1, 2));*

*squareGeometry.faces.push(new THREE.Face3(0, 2, 3));*

Viele gleiche Objekte erstellen und zufällig im Raum verteilen

Wenn mehrere Objekte in einem Loop erstellt werden, liegt der Nachteil darin, dass danach nicht mehr jedes einzelne Objekt angesteuert oder verändert werden kann. Um dies zu erreichen, kann in einem Loop ein Objekt erstellt werden (geometrie und material zuvor erstellt).

*for ( var i = 0; i < 2000; i ++ ) {*

*var object = new THREE.Mesh( geometry, material );*

*object.position.x = Math.random() \* 800 - 400;*

*object.position.y = Math.random() \* 800 - 400;*

*object.position.z = Math.random() \* 800 - 400;*

*object.rotation.x = Math.random() \* 2 \* Math.PI;*

*object.rotation.y = Math.random() \* 2 \* Math.PI;*

*object.rotation.z = Math.random() \* 2 \* Math.PI;*

*object.scale.x = Math.random() + 0.5;*

*object.scale.y = Math.random() + 0.5;*

*object.scale.z = Math.random() + 0.5;*

*scene.add( object );*

*}*

Rot: Das Objekt zufällig im Raum verteilen

Grün: Das Objekt schräg stellen

Blau: Die grösse des Objektes

Oberflächen

Einfarbig

*var Material = new THREE.MeshBasicMaterial({ color:0xFFFFFF, });*

Rot: Javascript Color Code

Textur

Eine Textur kann über ein Bild eingelesen werden:

var glassTexture = new THREE.ImageUtils.loadTexture("Glas.jpg");

Rot: Pfad zum Bild

Transparent

Nachdem die Textur erstellt wurde, kann diese noch transparent gemacht werden. Dazu nach dem erstellen der Textur folgenden Code einfügen:

*var glasmaterial = new THREE.MeshBasicMaterial( {*    
*map: glassTexture,*    
*depthWrite: false,*    
*transparent: true,*    
*opacity: 0.5,*    
*side:THREE.DoubleSide,*    
*combine: THREE.MixOperation*    
*} );*

*Rot: Der Name der zuvor erstellten Textur. Hier kann auch eine Farbe genommen werden*

*Grün: Wie Transparent soll die Seite sein*

Spiegeln

Um eine Oberfläche spiegelnd zu Machen, muss zuerst eine weitere Kamera eingebaut werden. Danach kann das Objekt erstellt werden.

*mirrorSphereCamera = new THREE.CubeCamera( 0.1, 5000, 512 );*

*scene.add( mirrorSphereCamera );*

*var sphereGeom = new THREE.SphereGeometry( 50, 32, 16 );*

*var mirrorSphereMaterial = new THREE.MeshBasicMaterial( { envMap: mirrorSphereCamera.renderTarget, side:THREE.DoubleSide } );*

*mirrorSphere = new THREE.Mesh( sphereGeom, mirrorSphereMaterial );*

*mirrorSphere.position.set(75,50,0);*

*mirrorSphereCamera.position = mirrorSphere.position;*

Rot: Kamera erstellen

Grün: Mit Hilfe dieser Kamera das Material des Objektes verändern

Wiederholende Texturen

Um eine Textur wiederholend zu machen, kann zuerst das Material normal erstellt werden. Die Geometrie und das Objekt werden normal erstellt. Nachdem das Material erstellt wurde, kann mit den markierten Zeilen gesagt werden, dass die textur wiederholend sein soll.

*var floorTexture = new THREE.ImageUtils.loadTexture( 'checkerboard.jpg' );*

*floorTexture.wrapS = floorTexture.wrapT = THREE.RepeatWrapping;*

*floorTexture.repeat.set( 10, 10 );*

*var floorMaterial = new THREE.MeshBasicMaterial( { map: floorTexture, side:THREE.BackSide } );*

*var floorGeometry = new THREE.PlaneGeometry(1000, 1000, 10, 10);*

*var floor = new THREE.Mesh(floorGeometry, floorMaterial);*

*floor.rotation.x = Math.PI / 2;*

Rot: Wie oft das Element in jede Richtung wiederholt werden soll.

SkyBox

*var Pfad = "Images/SkyBox/"*

*var urls = [Pfad + 'px.jpg', Pfad + 'nx.jpg', Pfad + 'py.jpg', Pfad + 'ny.jpg', Pfad + 'pz.jpg', Pfad + 'nz.jpg'];*

*var cubemap = THREE.ImageUtils.loadTextureCube(urls); // load textures*

*cubemap.format = THREE.RGBFormat;*

*var shader = THREE.ShaderLib['cube'];*

*shader.uniforms['tCube'].value = cubemap;*

*// create shader material*

*var skyBoxMaterial = new THREE.ShaderMaterial( {*

*fragmentShader: shader.fragmentShader,*

*vertexShader: shader.vertexShader,*

*uniforms: shader.uniforms,*

*depthWrite: false,*

*side: THREE.BackSide*

*});*

*// create skybox mesh*

*var skybox = new THREE.Mesh(new THREE.CubeGeometry(100000, 100000, 100000),skyBoxMaterial);*

*scene.add(skybox);*