9-15-2015 – 3D Angry Birds Starter Kit

1. Start with boiler plate code: angry.html – also need libs folder and images folder

2. Need Physi.js, Physi\_worker.js, and ammo.js – in the startup script, you must tell Physijs where things are.

<head>

<script src="libs/three.js"></script>

<script src="libs/physi.js"></script>

<script src="libs/threex.keyboardstate.js"></script>

</head>

Physijs.scripts.worker = 'libs/physijs\_worker.js';

Physijs.scripts.ammo = 'ammo.js';

3. Initialize keyboard and utilize new keyboard syntax.

keyboard = new THREEx.KeyboardState();

and

if( keyboard.pressed("left") )

{

cannon.rotation.z += 0.01;

}

else if( keyboard.pressed("right") )

{

cannon.rotation.z -= 0.01;

}

The THREEx.keyboard.js file has keyboard delimitations as follows:

THREEx.KeyboardState.MODIFIERS = ['shift', 'ctrl', 'alt', 'meta'];

THREEx.KeyboardState.ALIAS = {

'left' : 76,

'up' : 85,

'right' : 82,

'down' : 68,

'space' : 32,

'pageup' : 33,

'pagedown' : 34,

'tab' : 75,

'escape' : 65

};

4. Initialize scene in a new way for Physijs as follows:

scene = new Physijs.Scene();

scene.setGravity(new THREE.Vector3( 0, 0, -30 ));

scene.addEventListener('update', function()

{

scene.simulate();

});

5. After everything is set up, before document.body.appendChild() and render(), call scene.simulate()

6. Generate ground plane

var texture = THREE.ImageUtils.loadTexture('images/groundterrain.jpg');

var planeMaterial = new Physijs.createMaterial(new THREE.MeshLambertMaterial({map:texture}), .4, .8 );

var planeGeometry = new THREE.PlaneGeometry( 200, 200, 6 );

plane = new Physijs.BoxMesh( planeGeometry, planeMaterial, 0 );

scene.add( plane );

7. Generate cannon

var cylinderGeometry = new THREE.CylinderGeometry( 2, 2, 10 );

var cylinderMaterial = new THREE.MeshLambertMaterial({color:'lightgray'});

var can = new THREE.Mesh( cylinderGeometry, cylinderMaterial );

can.position.y = -5;

cannon = new THREE.Object3D();

cannon.add( can );

cannon.rotation.z = Math.PI / 2;

cannon.position.x -= 84;

cannon.position.z += 20;

scene.add( cannon );

8. Generate cannon ball

var texture = THREE.ImageUtils.loadTexture('images/balltexture.jpg');

var ballGeometry = new THREE.SphereGeometry( 3 );

var ballMaterial = Physijs.createMaterial( new THREE.MeshLambertMaterial({map:texture}), .95, .95 );

ball = new Physijs.SphereMesh( ballGeometry, ballMaterial );

ball.position.x = cannon.position.x + 10;

ball.position.y = cannon.position.y;

ball.position.z = cannon.position.z;

9. For targets you will need to use a factory design pattern:

10. Create a method to generate each target type

targetlist = [];

for( var i=0; i<4; i++ )

{

var geo = new THREE.BoxGeometry( 4, 4, 12 );

var mat = Physijs.createMaterial( new THREE.MeshLambertMaterial({color:'blue'}), .95, .95 );

var msh = new Physijs.BoxMesh( geo, mat );

switch( i )

{

case 0: msh.position.x = 80; break;

case 1: msh.position.x = 85; msh.position.y = 5; break;

case 2: msh.position.x = 90; break;

case 3: msh.position.x = 85; msh.position.y = -5; break;

}

msh.position.z = 6;

targetlist.push( msh );

scene.add( msh );

}

11. Control the game:

left

z += 0.01

right

z -= 0.01

up

y -= 0.02 ( check < -( Math.PI / 3 ) )

down

y += 0.02 ( check > 0 )

tab

fire: balllaunched = true, scene.add( ball ),

ball.applyCentralImpulse(

new THREE.Vector3( 8000, -( Math.PI / 2 - cannon.rotation.z ) \* 4000,

-cannon.rotation.y \* 10000 ) )

escape

remove ball: balllaunched = false, scene.remove( ball ),

GenerateCannonBall();

if balllaunched

check .z < -5 then remove (same code as escape)

**Loading Models**

1. Include the javascript files as follows:

<script src="libs/three.js"></script>

<script src="libs/threex.keyboardstate.js"></script>

<script src='libs/loaders/STLLoader.js'></script>

<script src='libs/loaders/ColladaLoader.js'></script>

<script src='libs/loaders/OBJLoader.js'></script>

<script src='libs/loaders/MTLLoader.js'></script>

<script src='libs/loaders/OBJMTLLoader.js'></script>

<script src='libs/loaders/BinaryLoader.js'></script>

<script src='libs/threex.universalloader.js'></script>

Know the URLs of models. For this example we will use female02.obj and female02.mtl

Add the loader code

var loader = new THREEx.UniversalLoader();

var url = ['female02.obj', 'female.mtl'];

loader.load(url, function(object3d)

{

// this function will be notified when the model is loaded

scene.add(object3d)

render();

});