HTML5
Canvas Element

3D WebGL with THREE Framework

http://threejs.org

http://threejs.org

three.js r59

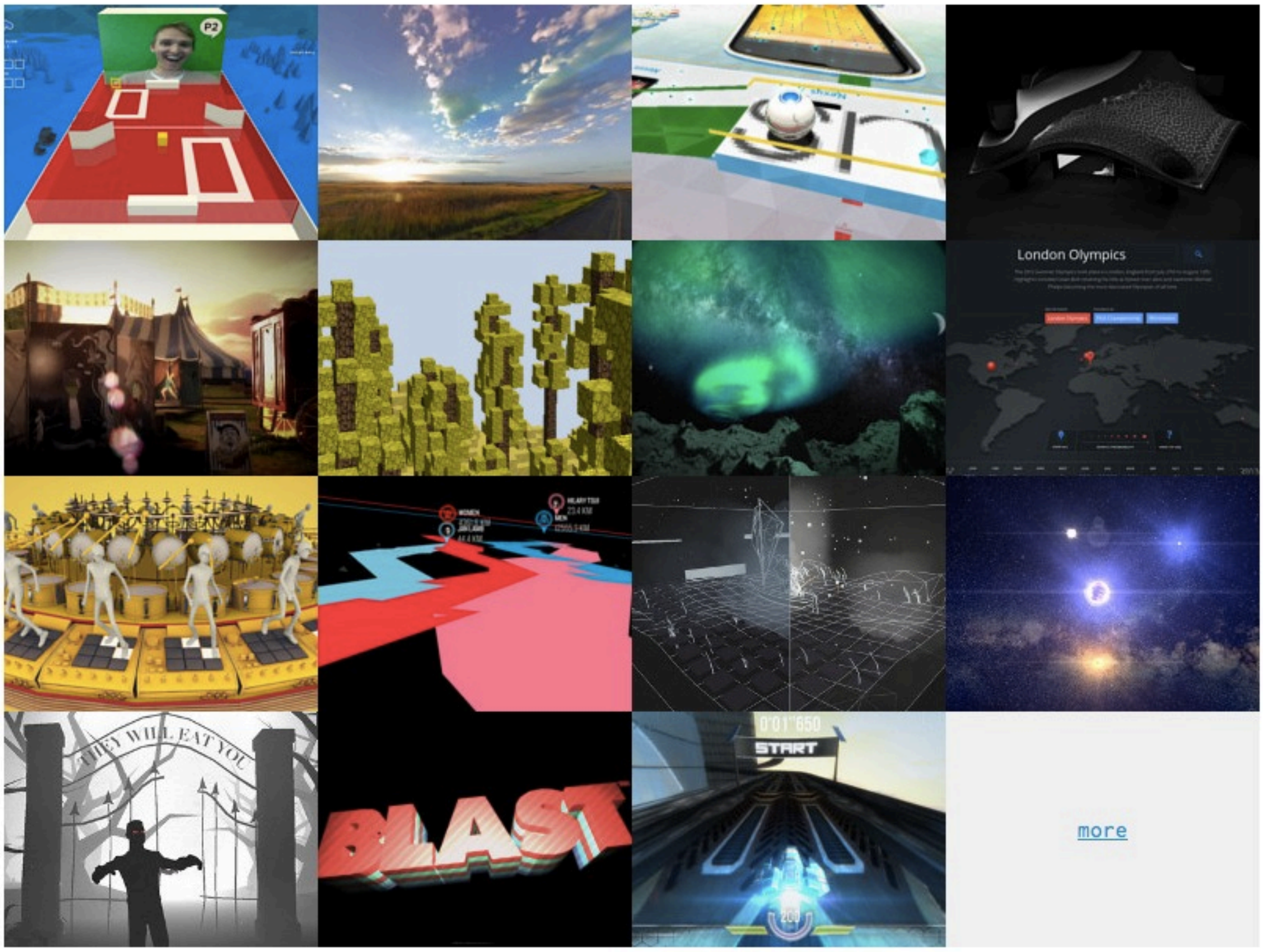
featured projects

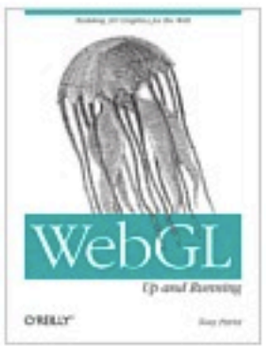
download

getting started
documentation
google+
chat
help

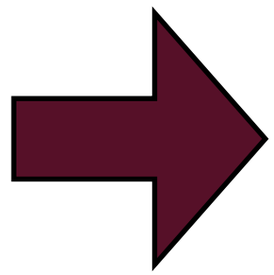
github
contributors
wiki
issues

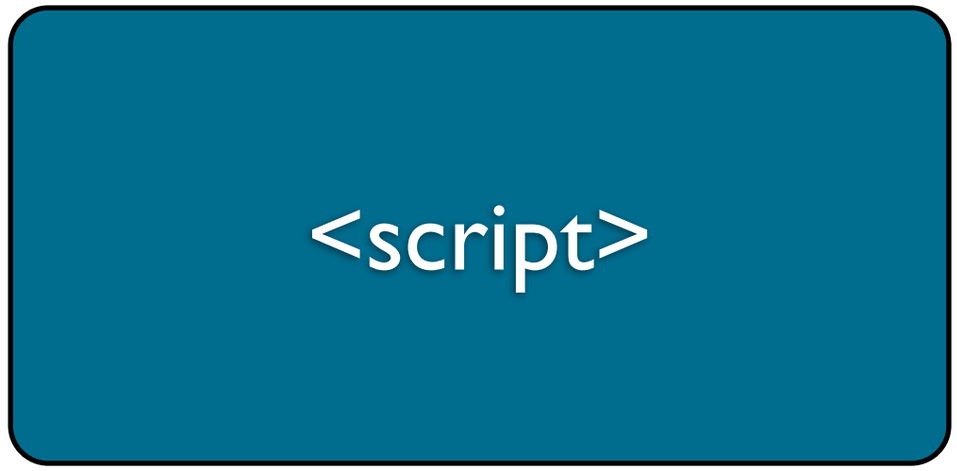
editor (beta)





three.min.js





THREE

Camera

Scene

Lights

Renderer

Shapes

Materials

Mesh

Object3D →

Edit this page

Camera

Abstract base class for cameras. This class should always be inherited when you build a new camera.

Constructor

Camera()

This constructor sets following properties to the correct type: matrixWorldInverse, projectionMatrix and projectionMatrixInverse.

Properties

.matrixWorldInverse

This is the inverse of matrixWorld. MatrixWorld contains the Matrix which has the world transform of the Camera.

.projectionMatrix

This is the matrix which contains the projection.

.projectionMatrixInverse

This is the inverse of projectionMatrix.

Methods

.lookAt(vector)

Basic Commands

```
// create scene
scene  = new THREE.Scene();
scene.add(object);
```

Cameras

Camera
OrthographicCamera
PerspectiveCamera

```
// create and position camera
camera = new THREE.PerspectiveCamera(
  FOV, aspect ratio, near clipping, far clipping
);
```

```
position, rotation, matrix
```

```
SUPER CLASS - Object3D
```

Basic Commands

```
// create Geometry (SHAPE)
```

Extras / Geometries

CircleGeometry
ConvexGeometry
CubeGeometry
CylinderGeometry
ExtrudeGeometry
IcosahedronGeometry
LatheGeometry
OctahedronGeometry
ParametricGeometry
PlaneGeometry
PolyhedronGeometry
ShapeGeometry
SphereGeometry
TetrahedronGeometry
TextGeometry
TorusGeometry
TorusKnotGeometry
TubeGeometry

Materials

Material
LineBasicMaterial
LineDashedMaterial
MeshBasicMaterial
MeshDepthMaterial
MeshFaceMaterial
MeshLambertMaterial
MeshNormalMaterial
MeshPhongMaterial
ParticleBasicMaterial
ParticleCanvasMaterial
ParticleDOMMaterial
ShaderMaterial
SpriteMaterial

```
cube = new THREE.CubeGeometry(width, height, depth);

// create material
material = new THREE.MeshBasicMaterial( {prop} );
material = new THREE.MeshLambertMaterial( {prop} );
```

Basic Commands

```
// create Geometry (SHAPE)
```

Extras / Geometries

CircleGeometry
ConvexGeometry
CubeGeometry
CylinderGeometry
ExtrudeGeometry
IcosahedronGeometry
LatheGeometry
OctahedronGeometry
ParametricGeometry
PlaneGeometry
PolyhedronGeometry
ShapeGeometry
SphereGeometry
TetrahedronGeometry
TextGeometry
TorusGeometry
TorusKnotGeometry
TubeGeometry

Materials

Material
LineBasicMaterial
LineDashedMaterial
MeshBasicMaterial
MeshDepthMaterial
MeshFaceMaterial
MeshLambertMaterial
MeshNormalMaterial
MeshPhongMaterial
ParticleBasicMaterial
ParticleCanvasMaterial
ParticleDOMMaterial
ShaderMaterial
SpriteMaterial

Lights

Light
AmbientLight
AreaLight
DirectionalLight
HemisphereLight
PointLight
SpotLight

```
// Create Light
var pointLight = new THREE.PointLight(0xHEX Colour);
```

Object3D →

Edit this page

Mesh

Base class for Mesh objects, such as MorphAnimMesh and SkinnedMesh.

Constructor

Mesh(geometry, material)

geometry -- todo
material -- todo

todo

Mesh(geometry, material)

geometry — An instance of Geometry.
material — An instance of Material (optional).

Properties

.geometry

An instance of Geometry, defining the object's structure.

.material

An instance of Material, defining the object's appearance. Default is a MeshBasicMaterial with wireframe mode enabled and randomised colour.

Renderers

CanvasRenderer
WebGLRenderer
WebGLRenderTarget
WebGLRenderTargetCube
WebGLShaders

CanvasRenderer

Edit this page

The Canvas renderer displays your beautifully crafted scenes *not* using WebGL, but draws it using the (slower) Canvas 2D Context API.

This renderer can be a nice fallback from WebGLRenderer for simple scenes:

```
if (window.WebGLRenderingContext)
        renderer = new THREE.WebGLRenderer();
else
        renderer = new THREE.CanvasRenderer();
```

Note: both WebGLRenderer and CanvasRenderer are embedded in the web page using an HTML5 <canvas> tag.
The "Canvas" in CanvasRenderer means it uses Canvas 2D instead of WebGL.
Don't confuse either CanvasRenderer with the SoftwareRenderer example, which simulates a screen buffer in a Javascript array.

Renderers

CanvasRenderer
WebGLRenderer
WebGLRenderTarget
WebGLRenderTargetCube
WebGLShaders

WebGLRenderer

Edit this page

The WebGL renderer displays your beautifully crafted scenes using WebGL, if your device supports it.

This renderer has way better performance than CanvasRenderer.

Constructor

WebGLRenderer(parameters)

parameters is an optional object with properties defining the renderer's behaviour. The constructor also accepts no parameters at all. In all cases, it will assume sane defaults when parameters are missing.

canvas — A Canvas where the renderer draws its output.
precision — shader precision. Can be "highp", "mediump" or "lowp".
alpha — Boolean, default is true.
premultipliedAlpha — Boolean, default is true.
antialias — Boolean, default is false.
stencil — Boolean, default is true.
preserveDrawingBuffer — Boolean, default is false.
clearColor — Integer, default is 0x000000.
clearAlpha — Float, default is 0.
maxLights — Integer, default is 4.

Renderers

CanvasRenderer
WebGLRenderer
WebGLRenderTarget
WebGLRenderTargetCube
WebGLShaders

```
// create renderer
renderer = new THREE.CanvasRenderer();
renderer.setSize(600, 400);
document.getElementById(divID).appendChild(
  renderer.domElement
);

renderer.render(scene, camera);
```

Animating WebGL

Browser Timer

```
window.setInterval( function() {}, interval);
```

```
window.requestAnimationFrame( function() {} );
```

To loop animation,
request frame within loop function.

Basic Workflow

Build a scene and camera, position camera.

Add camera to scene.

Create Shapes, Materials and Lights, position all.

Group Shapes and Materials into Meshes.

Add Meshes and Lights to Scene.

Create Render

Request Animation Function

Render Scene within Animation Function