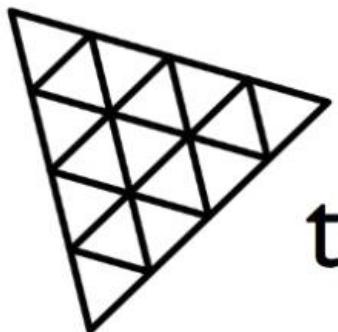


Visualization with Three.js

Benjamin Kenwright



three.js



Content

- Why's Visualization so Important?
- Science behind Visuals
- Why Three.js?
- Basics of 3D Graphics
- Step-by-Step Example
- Interactive Details/Effects
- Pros/Cons
- Questions/Discussion

Why's Visualization so Important?



You Cannot
Communicate Your
Idea Until You Can
Visualize It

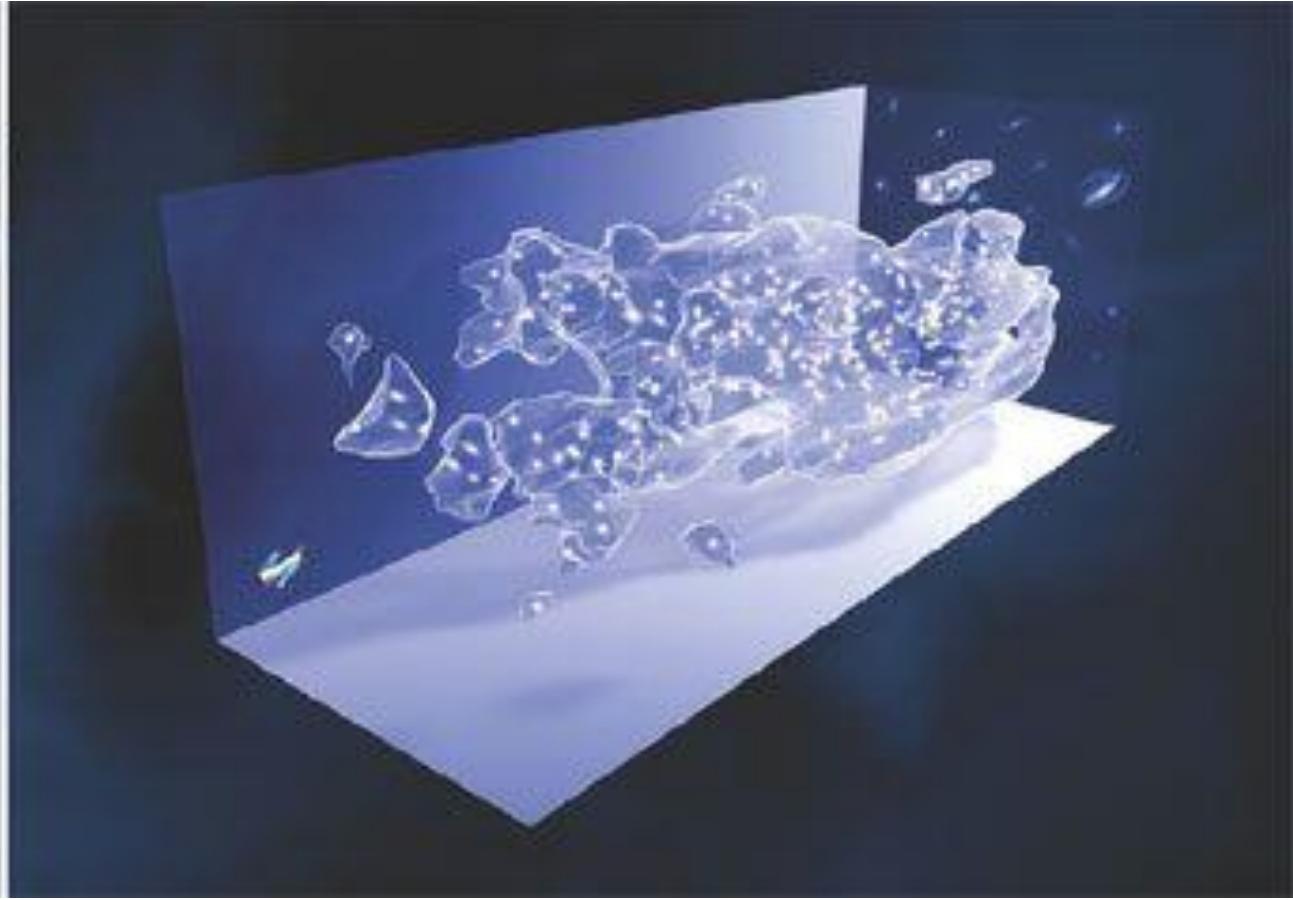
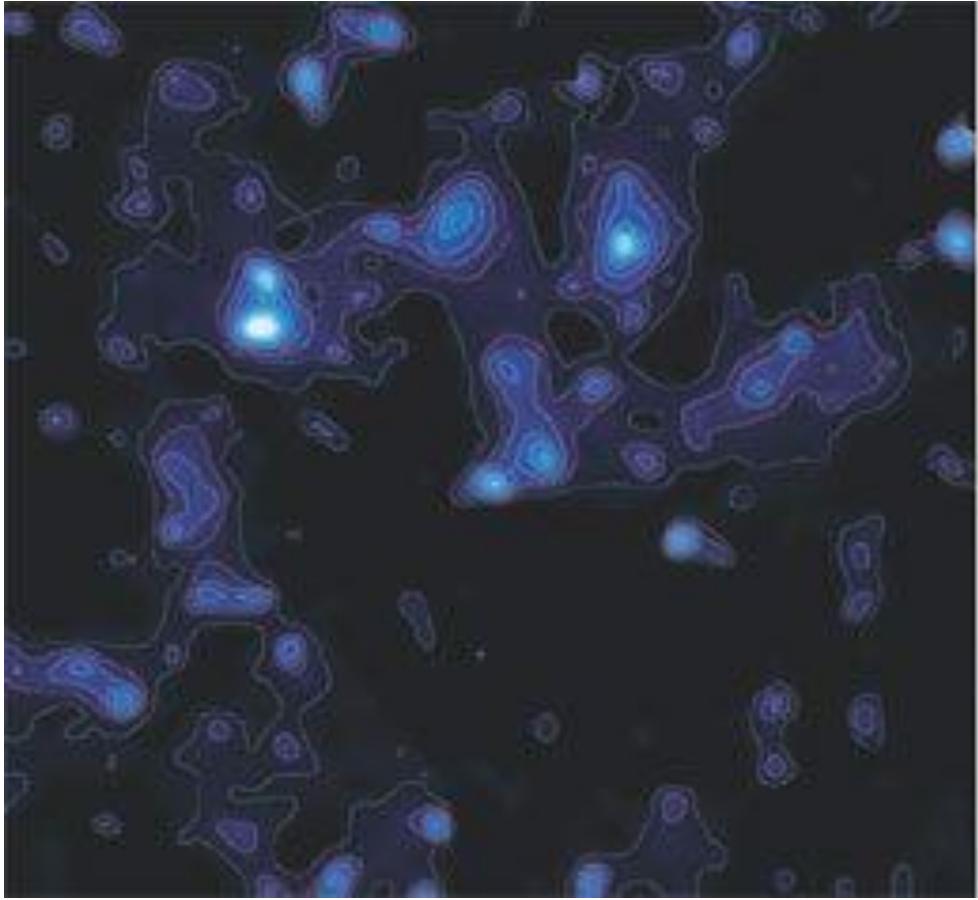
POWER of Visual Communication

(Science behind Visuals)

As humans, we learned to communicate with pictures long before we invented words.

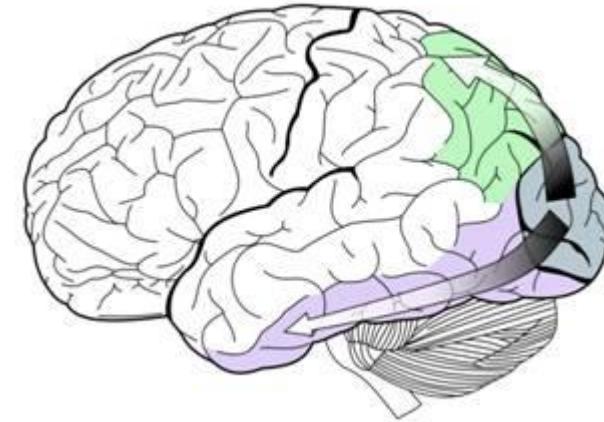
The proof is present in the form of **images in caves**

A two-dimensional map (left) of the Universe's mysterious dark matter can be translated into three dimensions (right) for clarity and impact



How do Visuals Help with Learning & Memory?

Our mind
translates words
into **visuals** before
processing its
meaning



Visuals Helps with Learning & Memory



- Helps process and retain information better
- It is not enough to use 'text'

“Given the **importance** of
imagery in science, it is
surprising that few **scholars**
are trained in graphic design

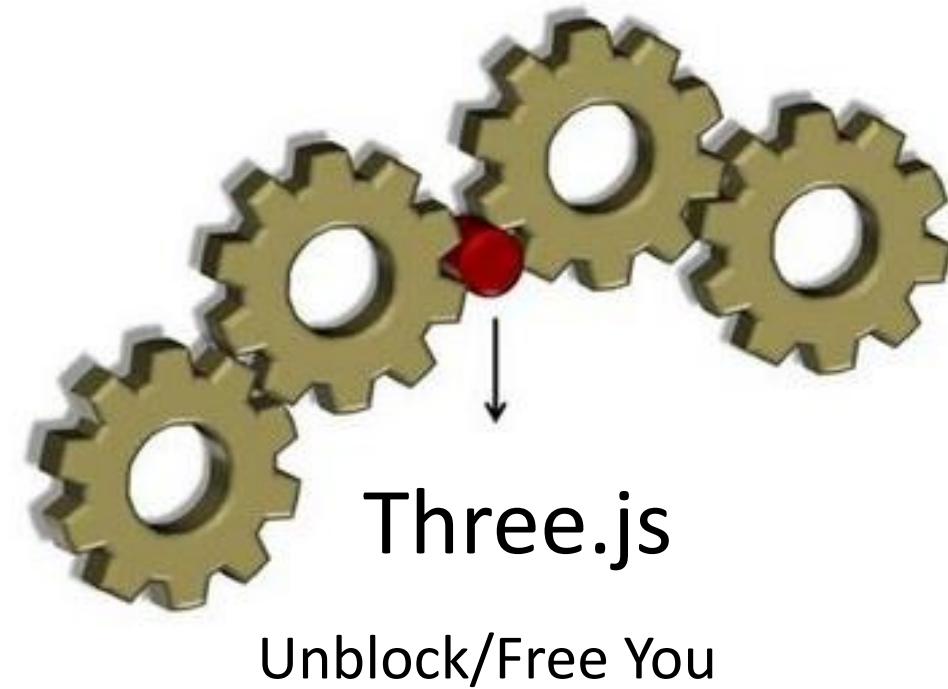


How to Unlock Creative Freedom To Explore & Experiment?

Don't want to be Limited by Software Packages/Tools
Develop Innovative and Bespoke Visual Solutions

Visual Creative Abilities

(Blocked or Limited)



Visuals to impress

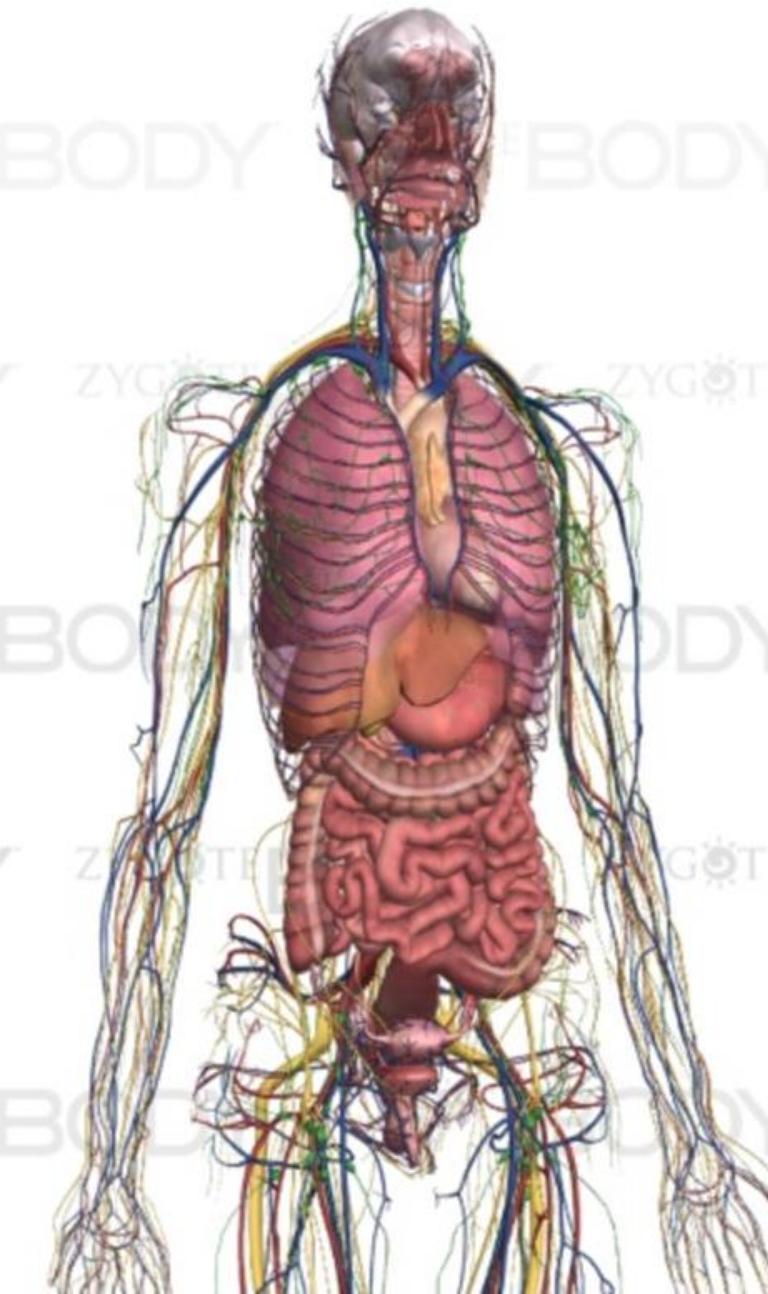


Three.js: Wow Factor

E X A M P L E S



Learn your body



human.biogigital

zygotobody

Small Arms and Ammunition – Imports & Exports

interactive visualization of government-authorized small arms and ammunition transfers from 1992 to 2010.



RUSSIAN FEDERATION

Statistics

\$0.11B
AMMUNITION

RUSSIAN FEDERATION

imported: \$36,420,720
exported: \$142,405,200

\$9.76M
AMMUNITION

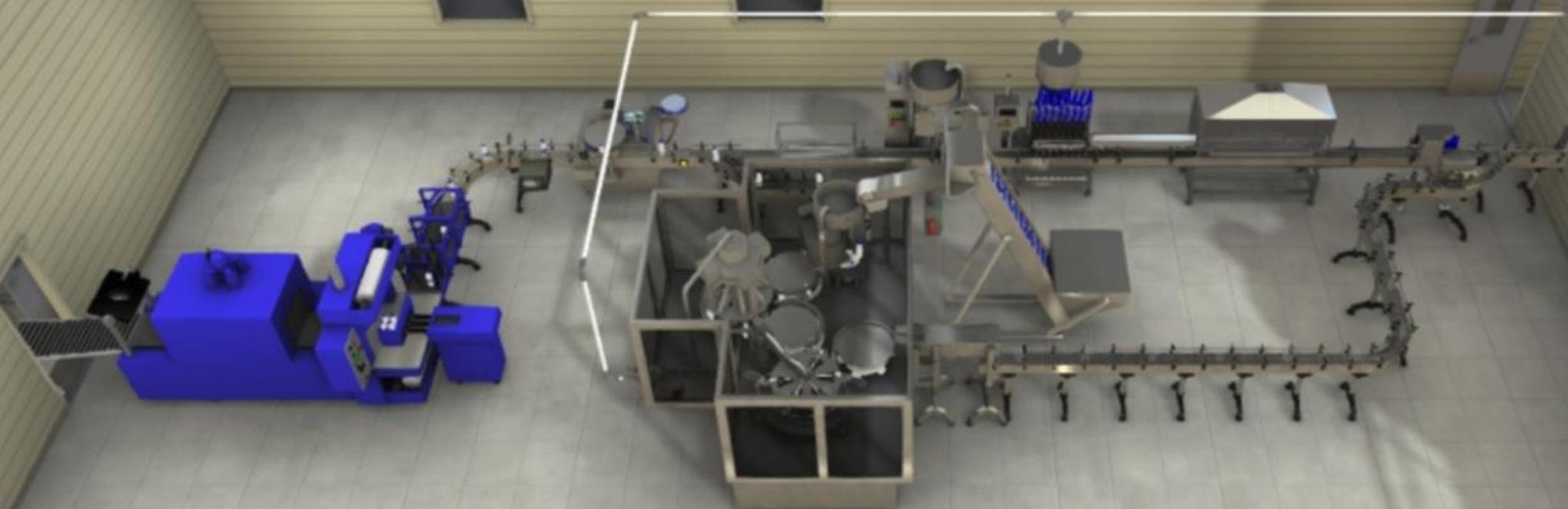
\$26.7M
CIVILIAN
WEAPONS

\$27.7M
CIVILIAN
WEAPONS

chromeexperiments

MILITARY
CIVILIAN
AMMO
IMP
EXP

Blend4Web



blend4web link

WebVR



WebVR link



1-КОМНАТНАЯ КВАРТИРА

40,7 M²

2-КОМНАТНАЯ КВАРТИРА

64,0 M²

3-КОМНАТНАЯ КВАРТИРА

97,6 M²

3D tour

ВИД СВЕРХУ

ГОСТИНАЯ

СПАЛЬНАЯ

ДЕТСКАЯ

КУХНЯ

ВАННАЯ

ТУАЛЕТ

ПРИХОЖАЯ



3D tour link

Panorama



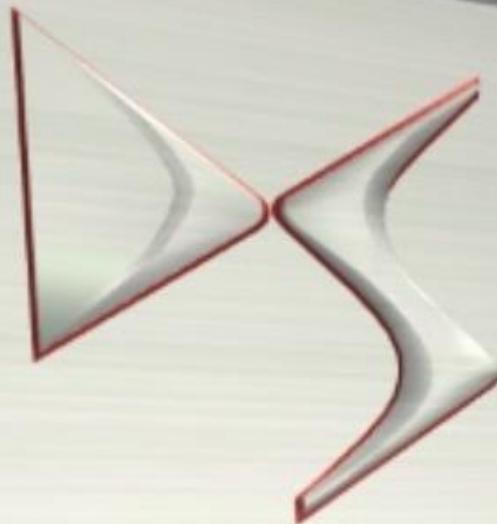
Colosseo link

Book shop

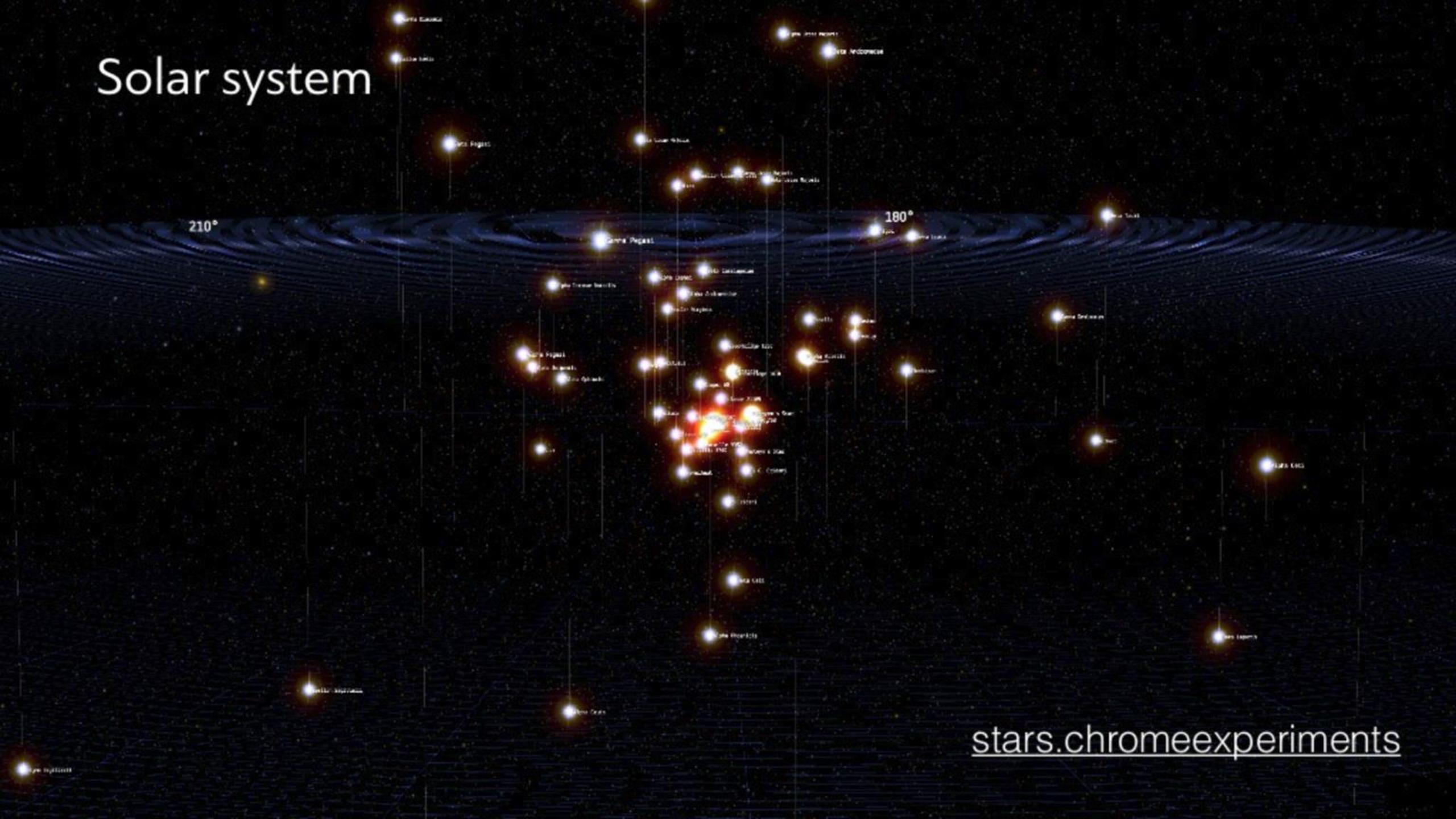
Free Books

bookcase

Car viewer



Solar system



[stars.chromeexperiments](http://stars.chromeexperiments.com)

Three.js

helps you develop your
visually

immersive skills

What is Three.js?

WebGL

Javascript 3D Library **for** WebGL

- HTML5 <canvas>
- OpenGL ES 2.0
- GLSL ES 1.10
- 2D/3D



K H R O N O S™
G R O U P

Components of WebGL



<canvas>



JS



**OpenGL
Shading Language**

Browser Support

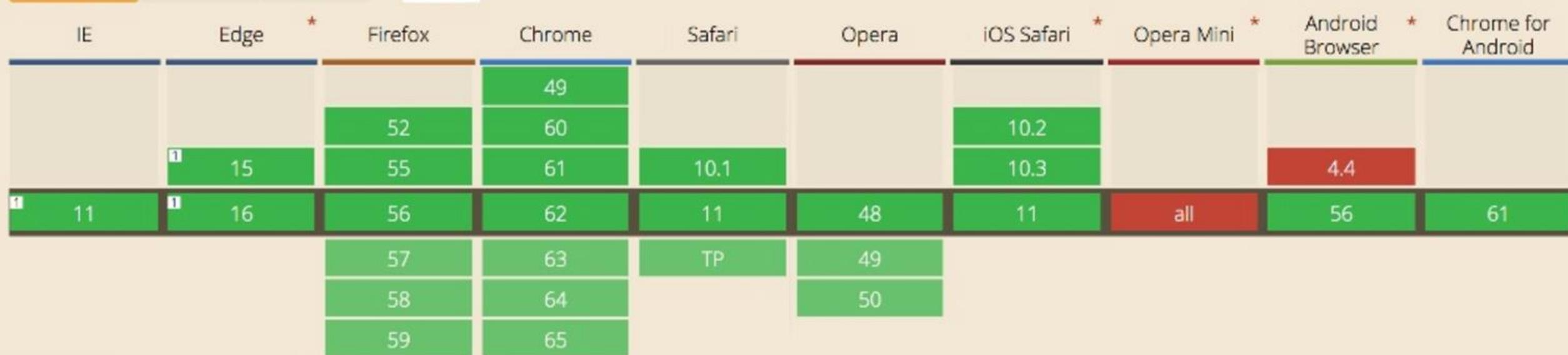
WebGL - 3D Canvas graphics - OTHER

Global

92.88%

Method of generating dynamic 3D graphics using JavaScript,
accelerated through hardware

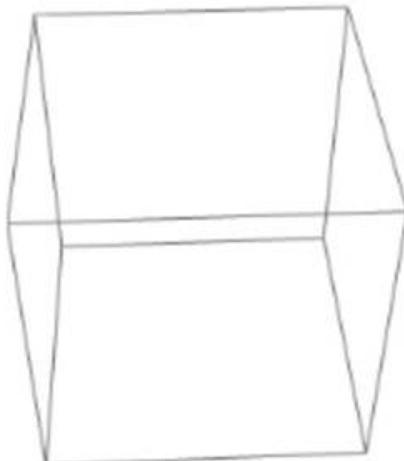
Current aligned Usage relative Date relative Show all



Your browser supports WebGL

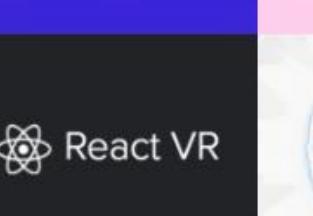
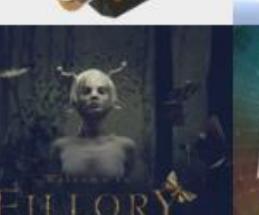
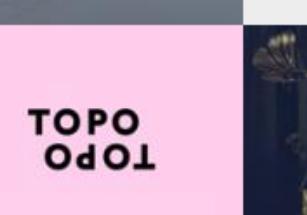
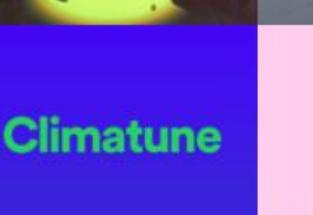
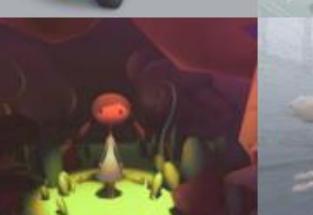
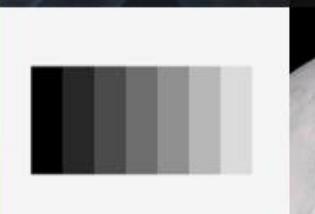
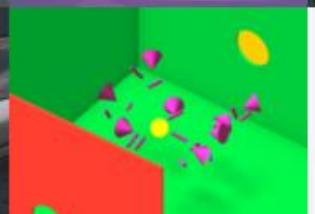
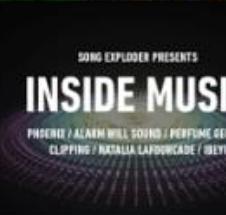
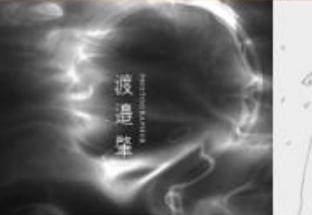
You should see a spinning cube. If you do not, please
[visit the support site for your browser.](#)

get.webgl.org





THREE.JS



Three.js Community

 Watch

2,450

 Star

52,484

 Fork

19,789

 28,394 commits

 4 branches

 97 releases

 1,104 contributors

Babylon.js Community

 Watch

518

 Star

9,524

 Fork

1,780

 19,013 commits

 6 branches

 136 releases

 235 contributors

```
depth / texture
effects / anaglyph
effects / parallaxbarrier
effects / peppersghost
effects / stereo
exporter / obj
geometries
geometries2
geometry / colors
geometry / colors / blender
geometry / colors / lookuptable
geometry / convex
geometry / cube
geometry / dynamic
geometry / extrude / shapes
geometry / extrude / shapes2
geometry / extrude / splines
geometry / hierarchy
geometry / hierarchy2
geometry / minecraft
geometry / minecraft / ao
geometry / normals
geometry / nurbs
geometry / shapes
geometry / spline / editor
geometry / teapot
geometry / terrain
geometry / terrain / fog
geometry / terrain / raycast
geometry / text
geometry / text / earcut
geometry / text / pnlti
gpgpu / birds
gpgpu / water
gpgpu / protoplanet
gpu / particle / system
hdr
```

Move mouse to disturb water.
Press mouse button to orbit around. 'W' key toggles wireframe.

Three.js Examples

threejs/examples

Export files

PLYLoader.js	FBXLoader2.js	collada
PVRLoader.js	GLTF2Loader.js	ctm
PlayCanvasLoader.js	GLTFLoader.js	deprecated
RGBELoader.js	HDRCubeTextureLoader.js	sea3d
STLLoader.js	KMZLoader.js	3MFLoader.js
SVGLoader.js	MD2Loader.js	AMFLoader.js
TGALoader.js	MMDLoader.js	AWDLoader.js
TTFLoader.js	MTLLoader.js	AssimpJSONLoader.js
UTF8Loader.js	NRRDLoader.js	AssimpLoader.js
VRMLLoader.js	OBJLoader.js	BVHLoader.js
VTKLoader.js	PCDLoader.js	BabylonLoader.js
XLoader.js	PDBLoader.js	BinaryLoader.js
draco_decoder.js	PLYLoader.js	ColladaLoader.js
	PVRLoader.js	ColladaLoader2.js
	PlayCanvasLoader.js	DDSLoader.js
	RGBELoader.js	DRACOLOader.js
	STLLoader.js	FBXLoader.js



Manual

[Getting Started](#)

[Creating a scene](#)

[Import via modules](#)

[Browser support](#)

[WebGL compatibility check](#)

[How to run things locally](#)

[How to use WebGL 2](#)

[Drawing lines](#)

[Creating text](#)

[Loading 3D models](#)

[FAQ](#)

[Useful links](#)

[Next Steps](#)

[How to update things](#)

[How to dispose of objects](#)

[How to create VR content](#)

[How to use post-processing](#)

[Matrix transformations](#)

[Animation system](#)

[Build with ESM](#)

[Testing with NPM](#)

Reference

[Animation](#)

[AnimationAction](#)

CubeTextureLoader

Class for loading a [CubeTexture](#). This uses the [ImageLoader](#) internally for loading files.

Example

[materials / cubemap](#)

[materials / cubemap / balls / reflection](#)

[materials / cubemap / balls / refraction](#)

[materials / cubemap / dynamic](#)

[materials / cubemap / refraction](#)

```
var scene = new THREE.Scene();
scene.background = new THREE.CubeTextureLoader()
  .setPath( 'textures/cubeMaps/' )
  .load([
    'px.png',
    'nx.png',
    'py.png',
    'ny.png',
    'pz.png',
    'nz.png'
  ]);
```

Constructor

`CubeTextureLoader(manager : LoadingManager)`

`manager` — The [loadingManager](#) for the loader to use. Default is [THREE.DefaultLoadingManager](#).

Creates a new CubeTextureLoader.

Great Documentation

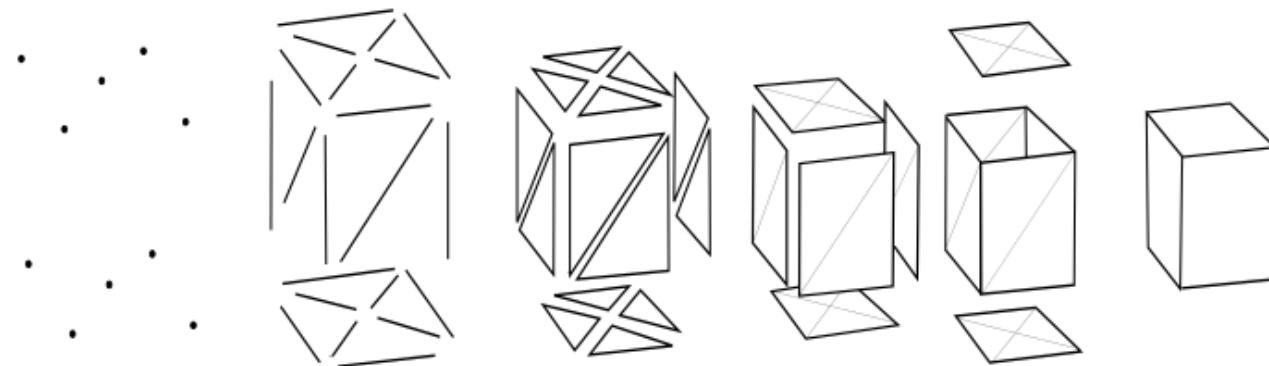
Most important **file** (and only
essential file for three.js)

three.js

three.js

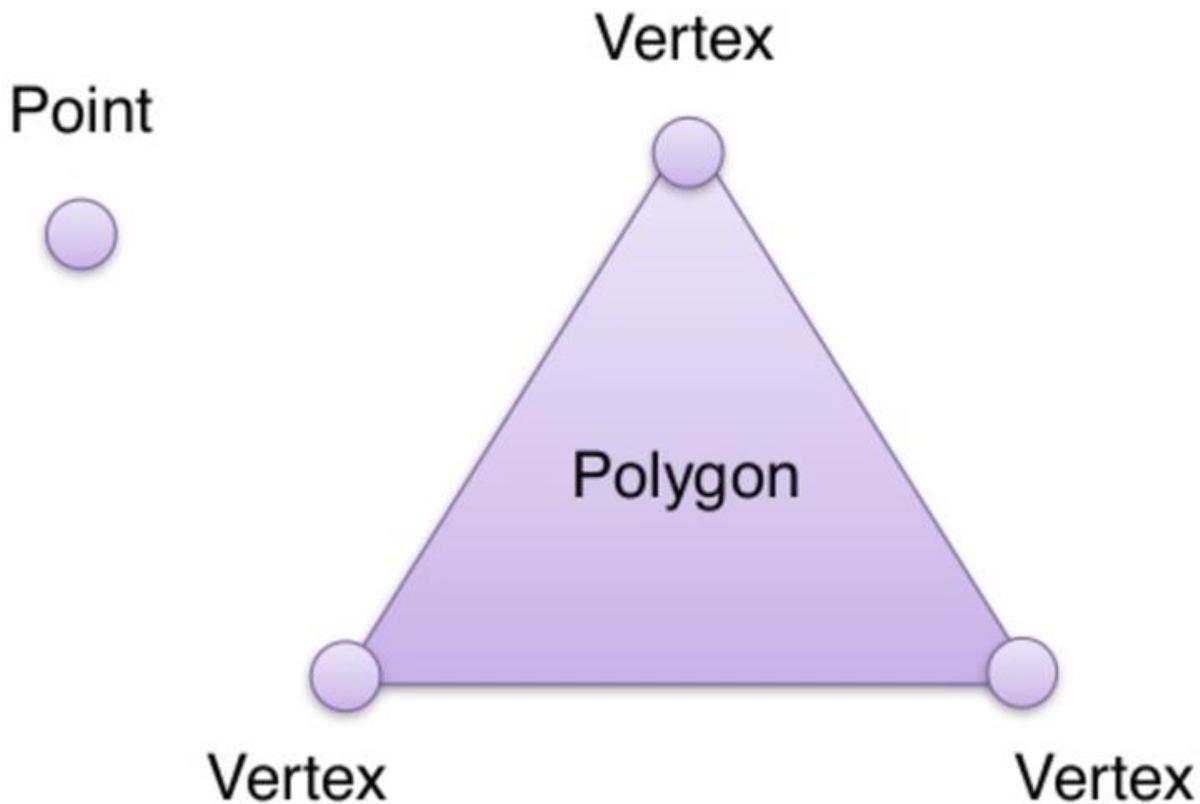
three.min.js is a minified version of three.js. It is totally identical code-wise but less human-readable

3D Basics



3D basics

- Point
- Polygon
- Object (mesh)
- Geometry
- Material
- Texture



Object Data

Example model.obj

Vertex list with coords (x,y,z)

```
v 13.825026512145996 -96.140419006347656 3.6714630126953125
v 15.503727912902832 -96.428817749023438 2.1252975463867187
v 14.864977836608887 -95.269874572753906 1.5220794677734375
v 12.35379695892334 -91.997489929199219 1.42974853515625
v 13.748141288757324 -93.280204772949219 1.0182418823242187
v 7.9108209609985352 -85.125518798828125 3.9087066650390625
v 11.508156776428223 -94.102958679199219 4.5504951477050781
v 13.997708320617676 -95.774742126464844 2.4739608764648437
v 13.337004661560059 -94.254402160644531 1.780059814453125
v 10.512875556945801 -90.527717590332031 2.5589218139648438
v 9.9423093795776367 -91.974296569824219 4.4449958801269531
v 12.121970176696777 -93.814460754394531 2.7707862854003906
v 8.8889608383178711 -89.88848876953125 4.7834281921386719
v 8.5577688217163086 -87.575393676757813 3.7520294189453125
v 65.255134582519531 50.347309112548828 32.405288696289063
v 66.766136169433594 50.430992126464844 33.477485656738281
v 63.850997924804687 51.162227630615234 31.309993743896484
v 61.49749755859375 53.767066955566406 30.037763595581055
v 62.612274169921875 55.586418151855469 32.590259552001953
v 63.765449523925781 56.516639709472656 34.983329772949219
v 86.712333679199219 -29.699516296386719 67.651824951171875
v 86.298973083496094 -27.467063903808594 67.311981201171875
v 87.931144714355469 -26.754478454589844 69.544113159179688
v 87.254051208496094 -28.534954071044922 69.252769470214844
v 54.208457946777344 52.926681518554688 17.700454711914063
v 55.330917358398438 51.7030029296875 17.893693923950195
```

Texture coordinates (u,v)

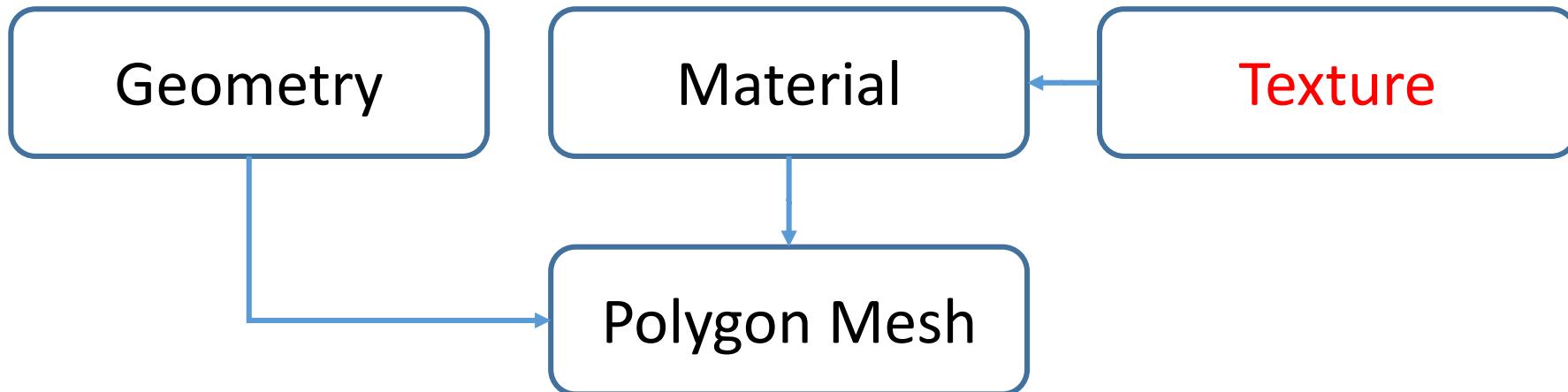
```
vt 0.056318659335374832 0.62613308429718018
vt 0.061030514538288116 0.6244666576385498
vt 0.05973927304148674 0.62342357635498047
vt 0.060924597084522247 0.62328207492828369
vt 0.058572947978973389 0.62352561950683594
vt 0.056074466556310654 0.62256932258605957
vt 0.044292308390140533 0.48838406801223755
vt 0.04412538930773735 0.48736923933029175
vt 0.043935131281614304 0.48618751764297485
```

Faces (v/vt)

```
f 70220/29439 91099/29452 70267/29440
f 91099/29452 71051/29454 71061/29455
f 70265/29449 70272/29442 70288/29456
f 70262/29434 70272/29442 70265/29449
f 91099/29452 71061/29455 70299/29453
f 70265/29449 70288/29456 98620/29450
f 70267/29440 70299/29453 70363/29448
```

Object

More than just ‘Triangles’



Textures

texture.jpg

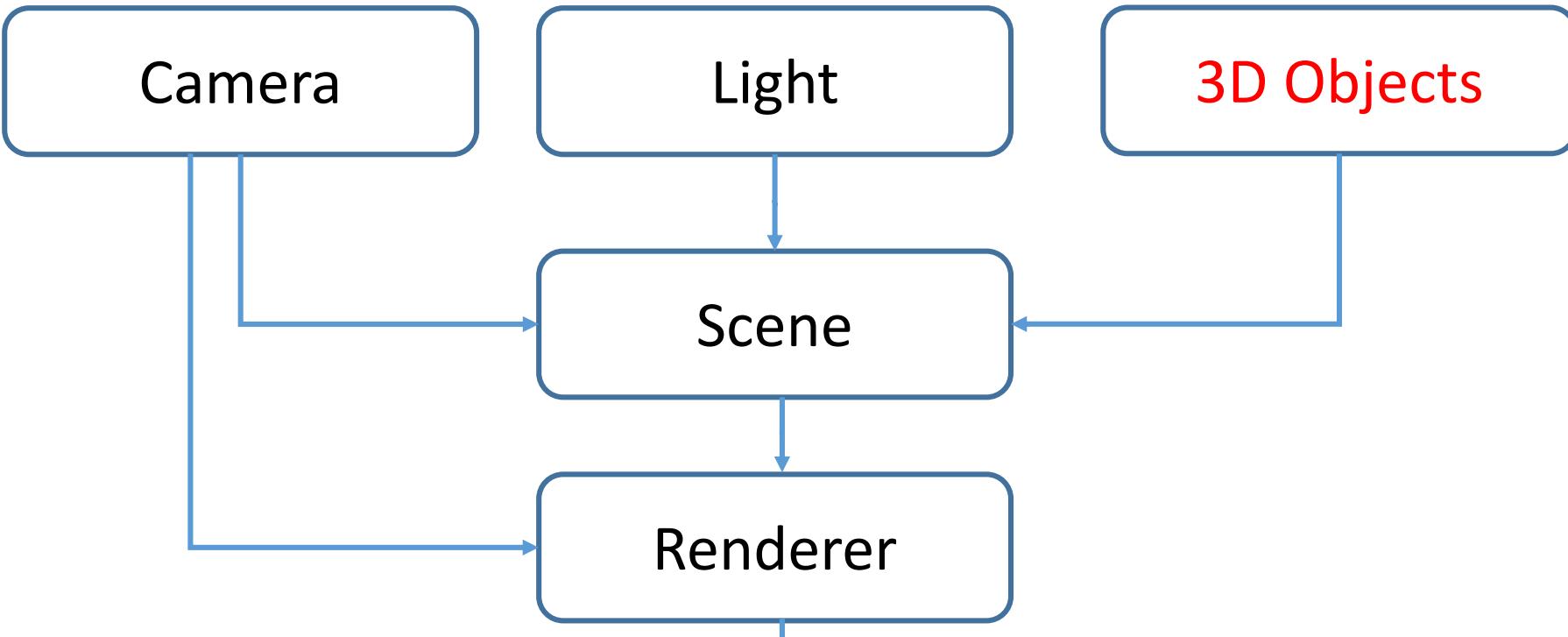


3D Terms

- Scene
- Light
- Camera
- Renderer



How everything connects



Example

(step by step)

Practical aspect

Simple (but not too simple example)

Example will need these files

Most important

three.min.js

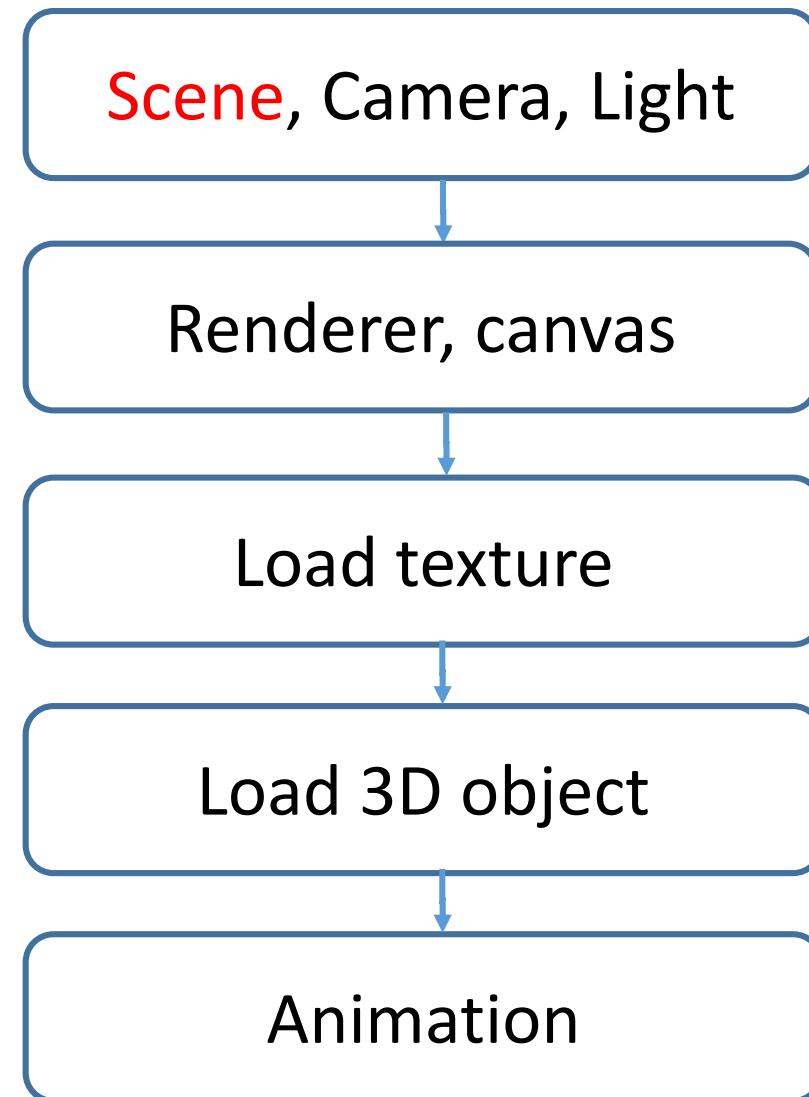
OBJLoader.js

TrackballControls.js

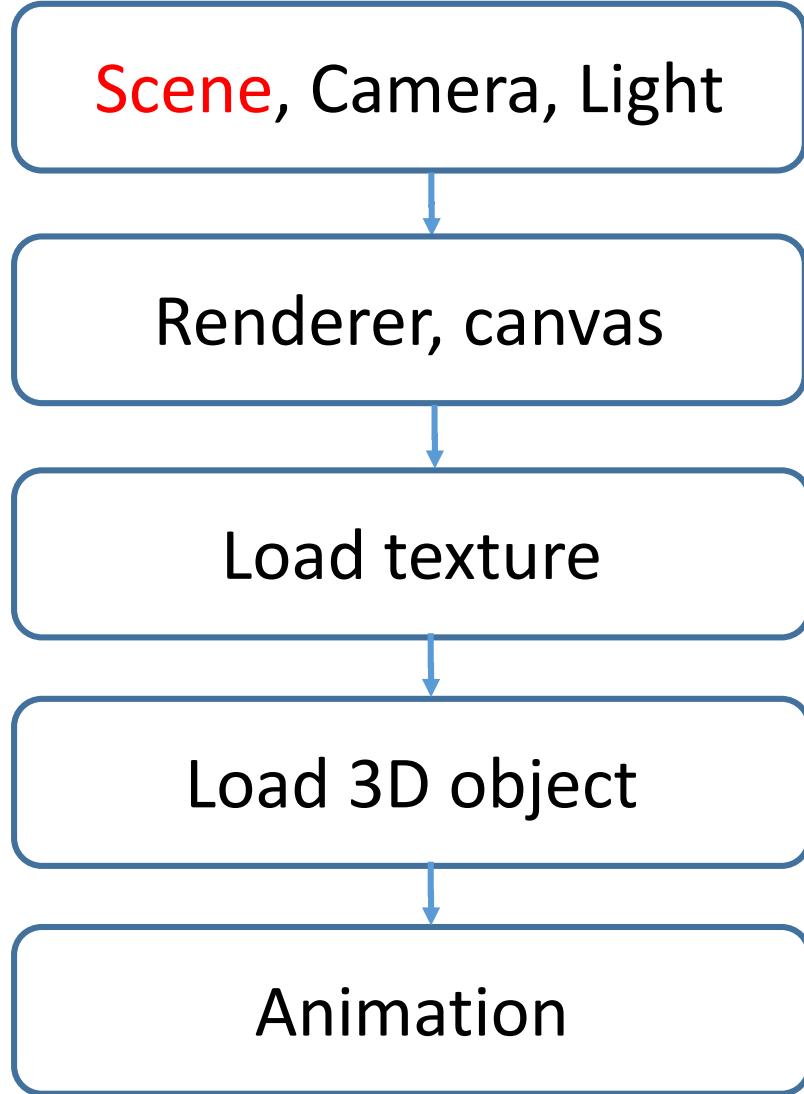
Helpers

Example

How many lines of code?



Example



100 Lines of Code
(with comments &
spacing)

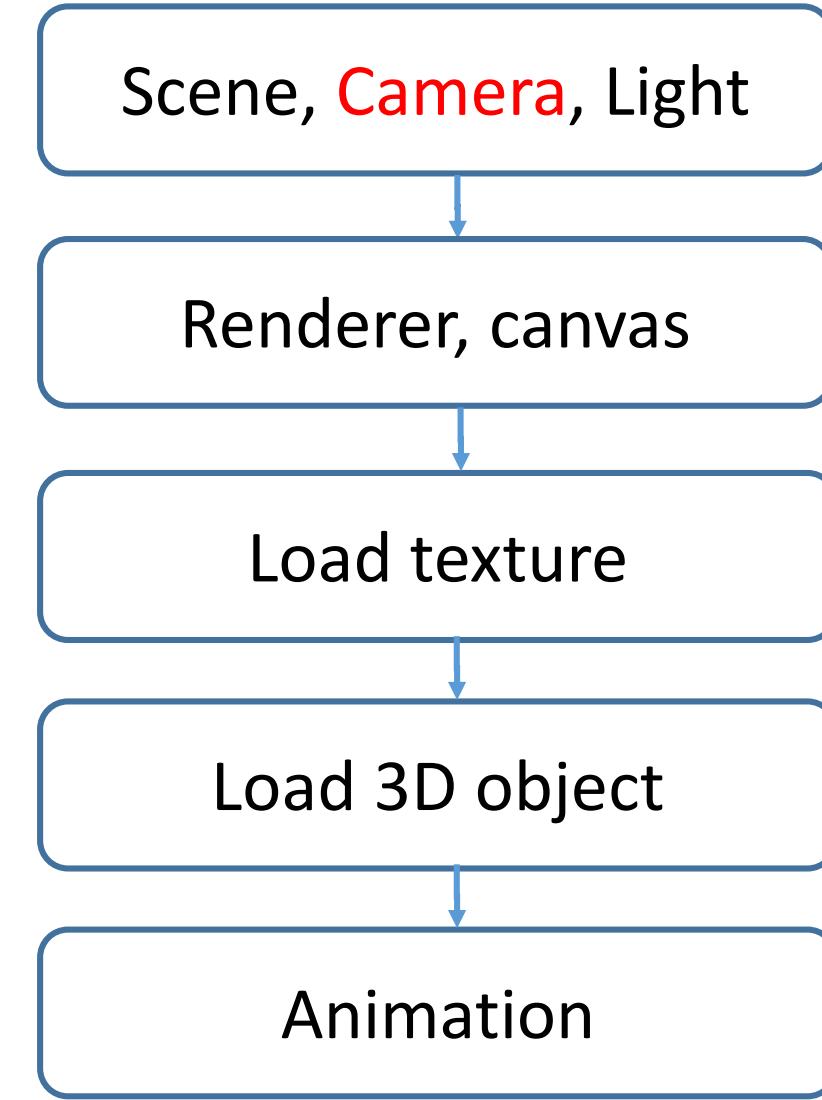


Scene

```
Player.container = document.getElementById("webgl-player");

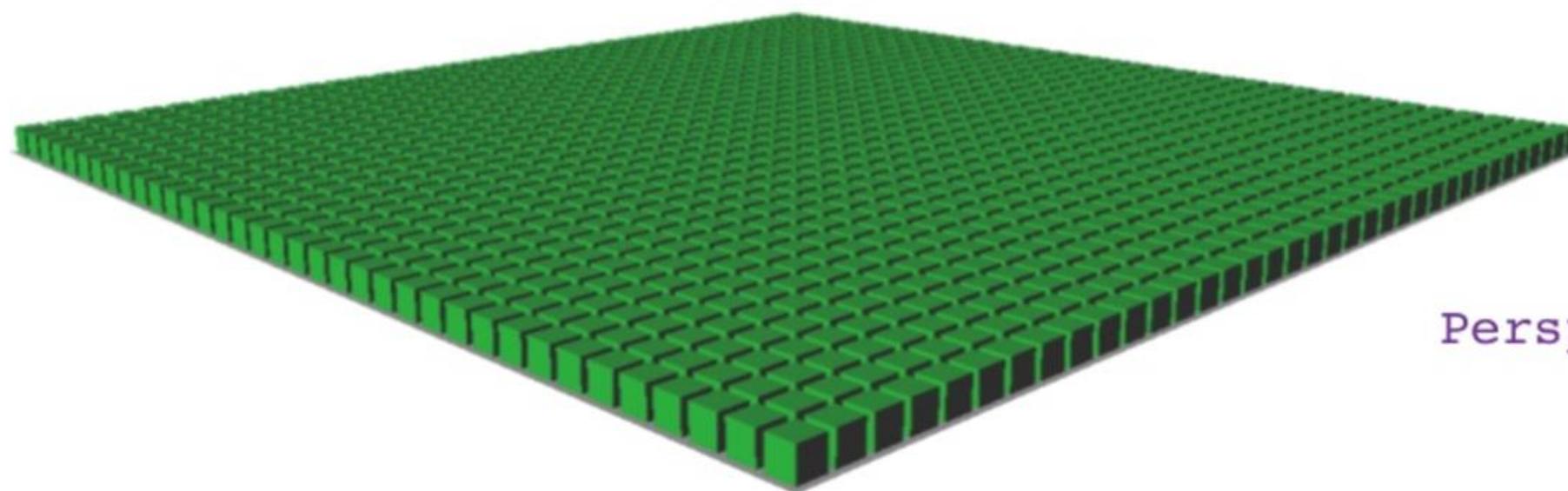
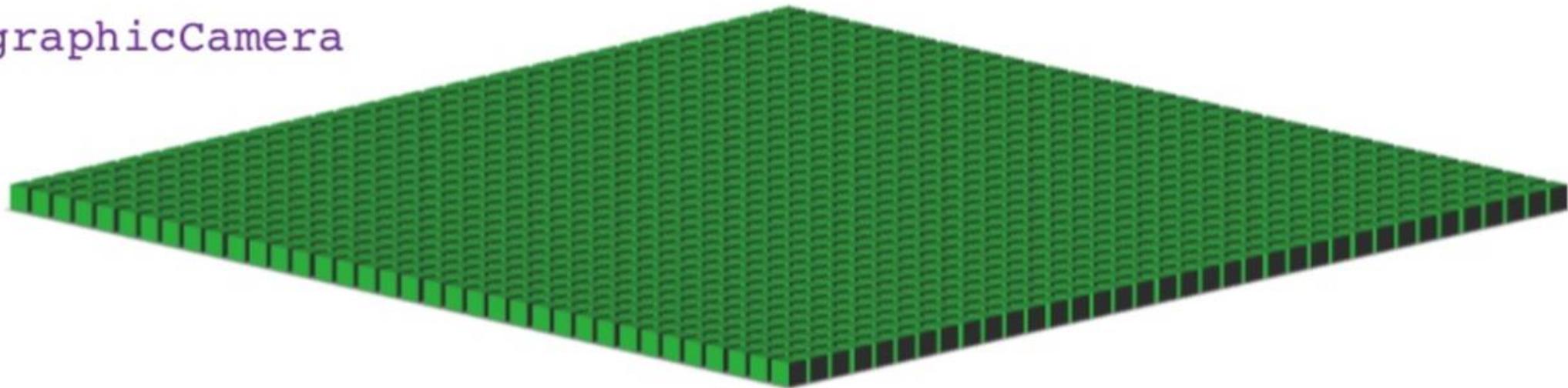
Player.size = {
    width: Player.container.offsetWidth,
    height: Player.container.offsetHeight
};

Player.scene = new THREE.Scene();
```



Camera types

OrthographicCamera

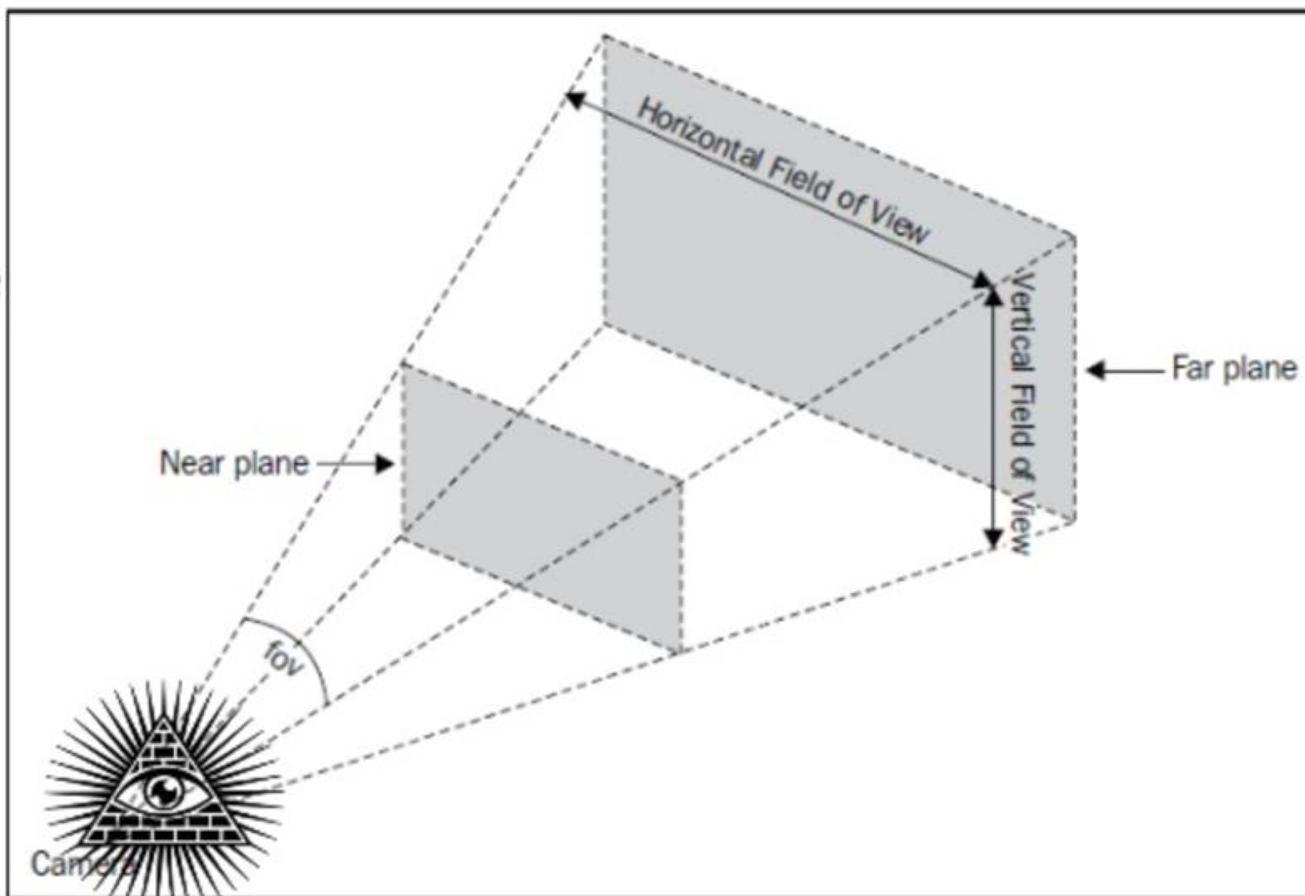


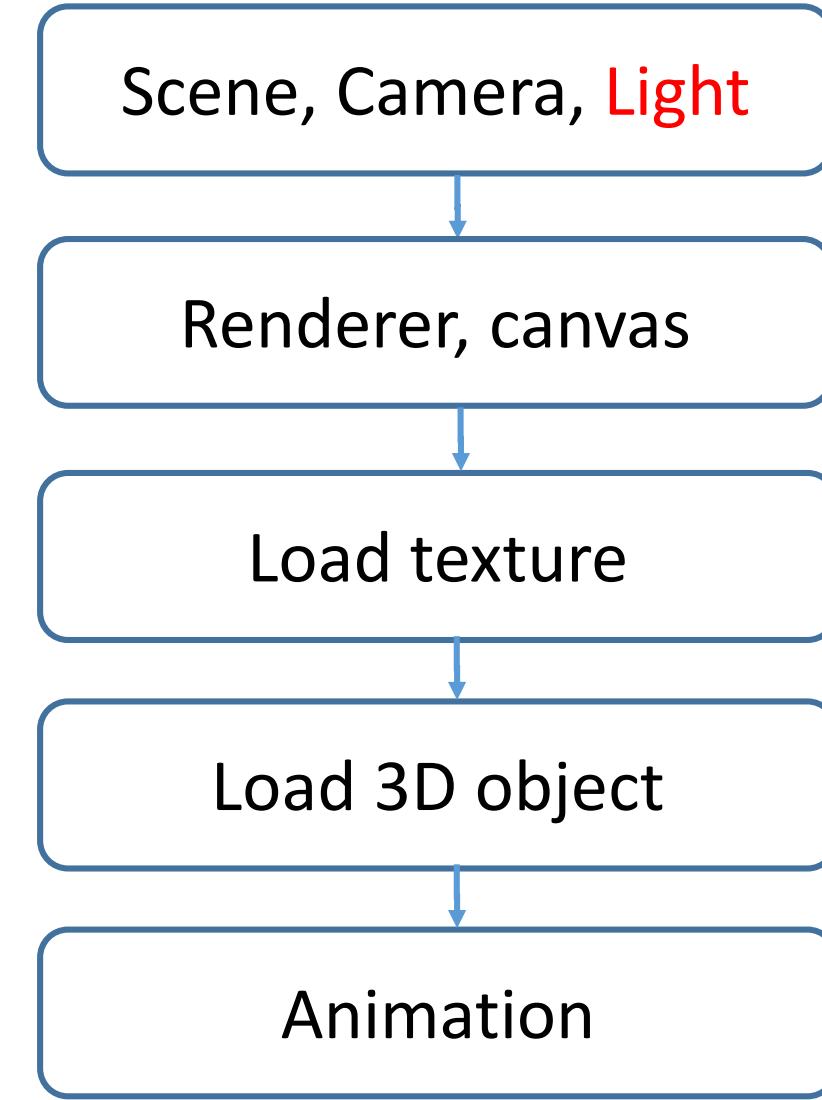
PerspectiveCamera

Camera

```
// PerspectiveCamera( fov, aspect, near, far )
aspect = Player.size.width / Player.size.height;
Player.camera = new THREE.PerspectiveCamera(45.0, aspect, 2, 8000);

Player.camera.position.z = 300;
Player.scene.add(Player.camera);
```





Light

```
Player.light = new THREE.AmbientLight();
Player.scene.add(Player.light);
```

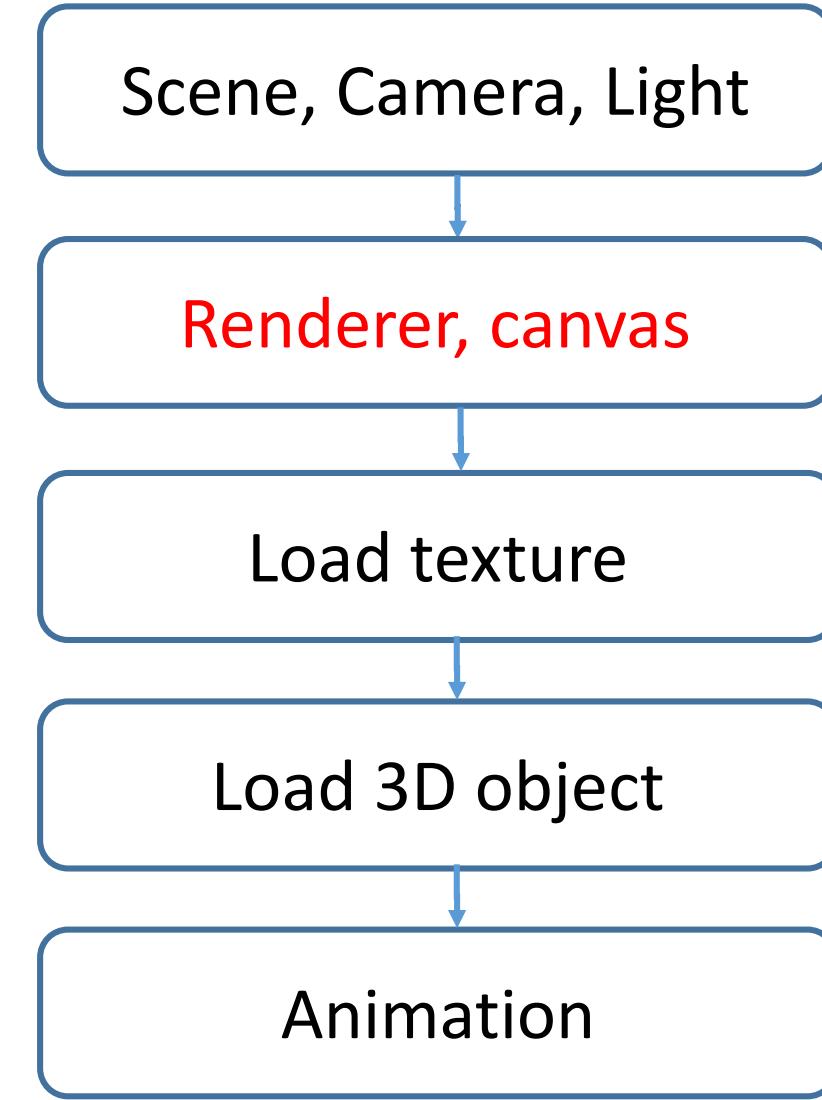
Light



```
// Player.scene.add(Player.light)
```



```
Player.scene.add(Player.light)
```



Renderer

```
Player.renderer = new THREE.WebGLRenderer({alpha: true});  
Player.renderer.setSize(Player.size.width, Player.size.height);  
  
// canvas  
Player.container.appendChild(Player.renderer.domElement);
```

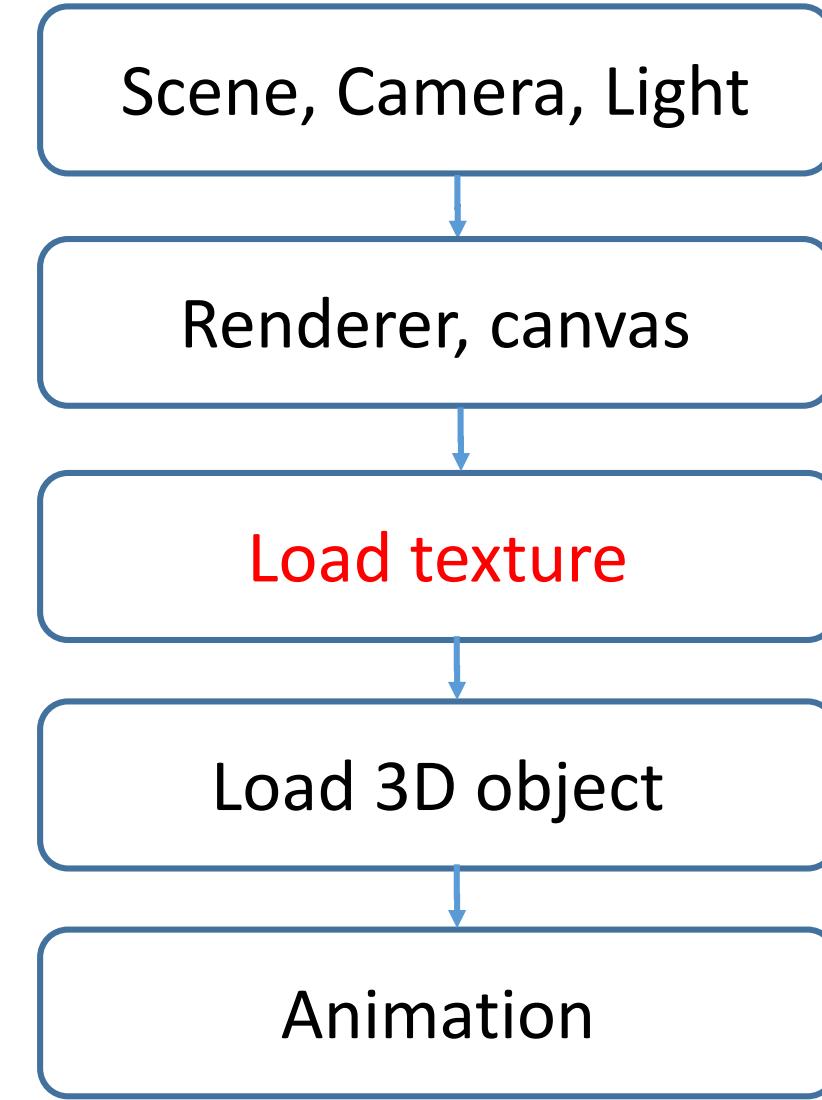
Renderer



```
THREE.WebGLRenderer()
```

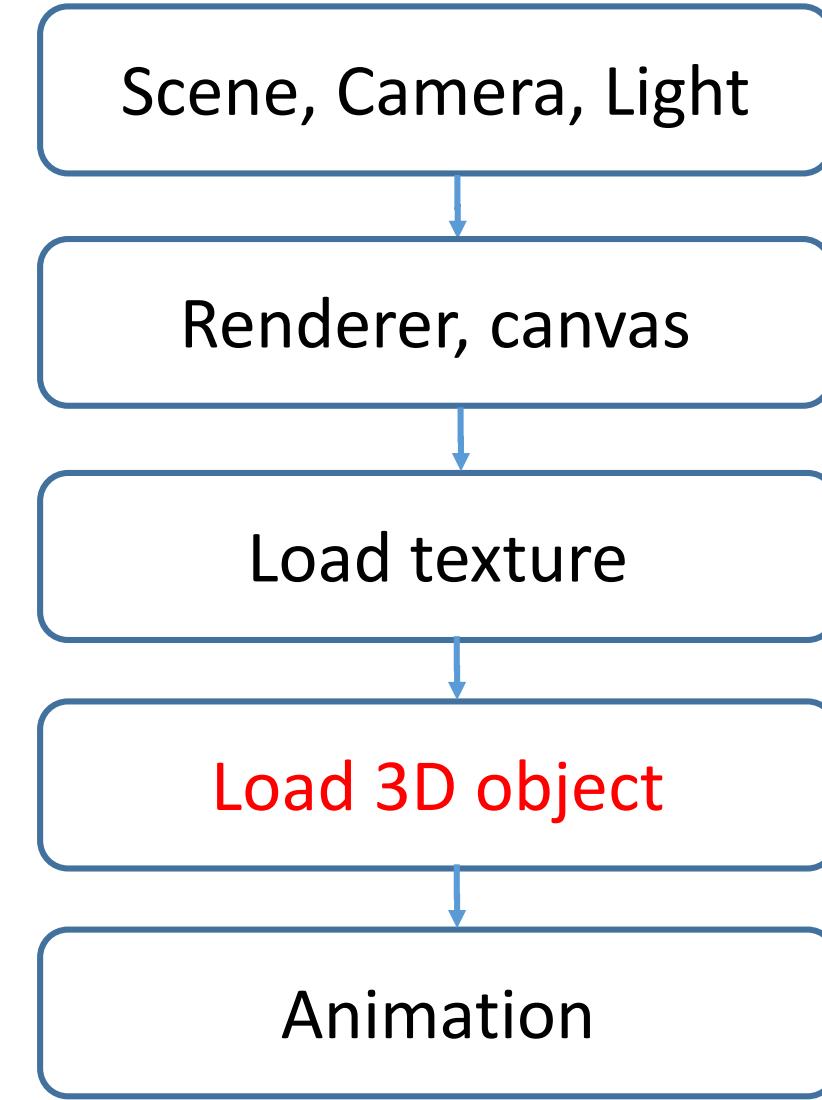


```
THREE.WebGLRenderer({alpha: true})
```



Load texture

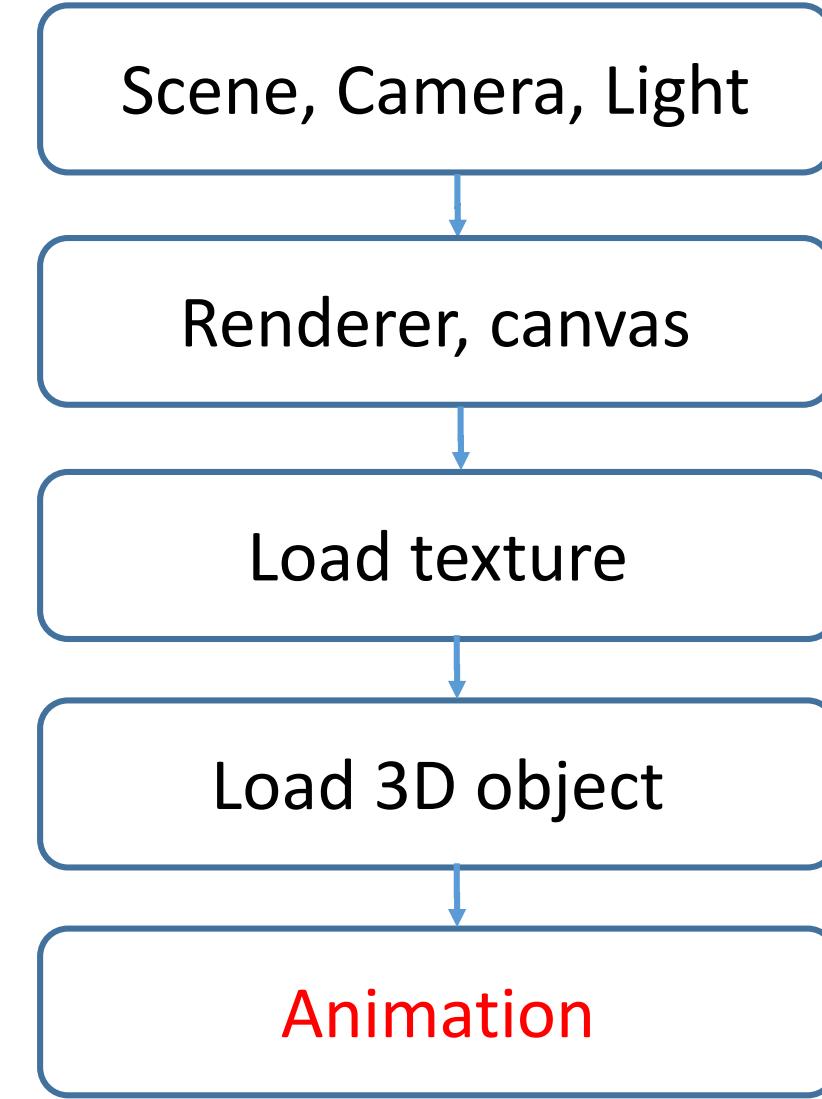
```
Player.textureLoader = new THREE.TextureLoader();
Player.textureLoader.load("texture.jpg", function(texture)
{
    Player.texture = texture;
    Player.loadModel();
});
```



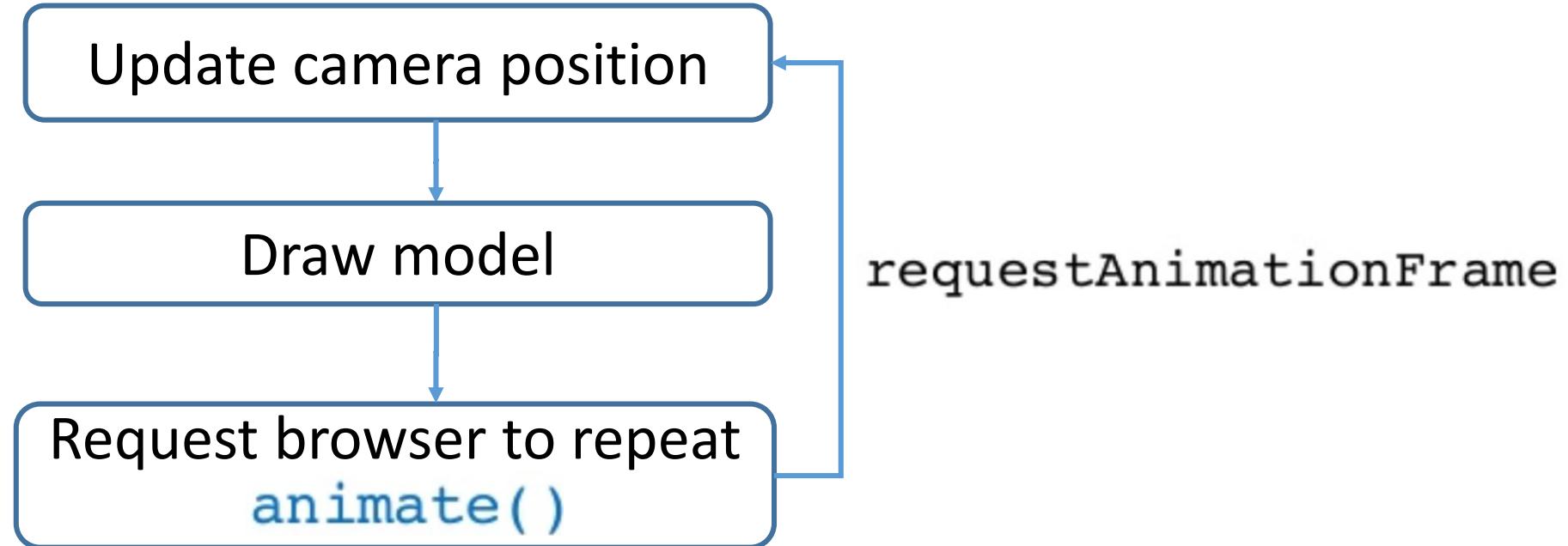
Loading model

```
loadModel: function() {  
  
    objectLoader = new THREE.OBJLoader();  
  
    objectLoader.load("object.obj", function(object) {  
        object.traverse(function(child) {  
            if (child instanceof THREE.Mesh) {  
                child.material.map = Player.texture;  
            }  
        });  
        Player.scene.add(object);  
    });  
}
```

You may also want to consider JSONLoader



Animation loop



```
animate: function() {  
    requestAnimationFrame(Player.animate);  
    Player.controls.update();  
    Player.renderer.render(Player.scene, Player.camera);  
}
```

Controls

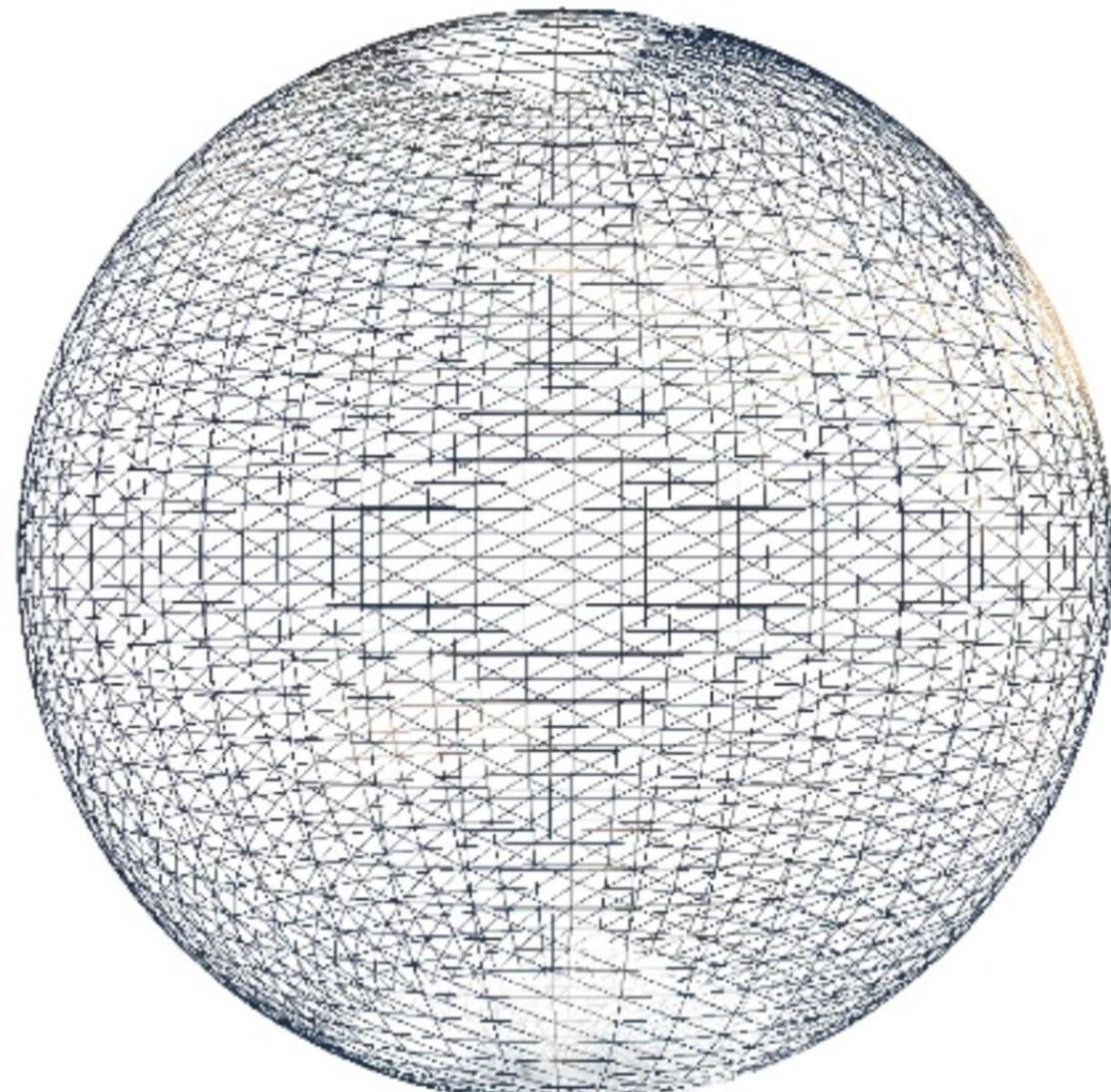
```
[Player.controls] = new THREE.TrackballControls(Player.camera,  
                                              Player.container);  
  
Player.animate();  
  
animate: function() {  
    requestAnimationFrame(Player.animate);  
    [Player.controls.update();]  
    Player.renderer.render(Player.scene, Player.camera);  
}
```

You may also want to consider OrbitControls

How to Cancel (Stop) Animation?

```
requestId = requestAnimationFrame/animate)  
cancelAnimationFrame(requestId)
```

What about Build in Geometry (Shapes)?



Example (Sphere)

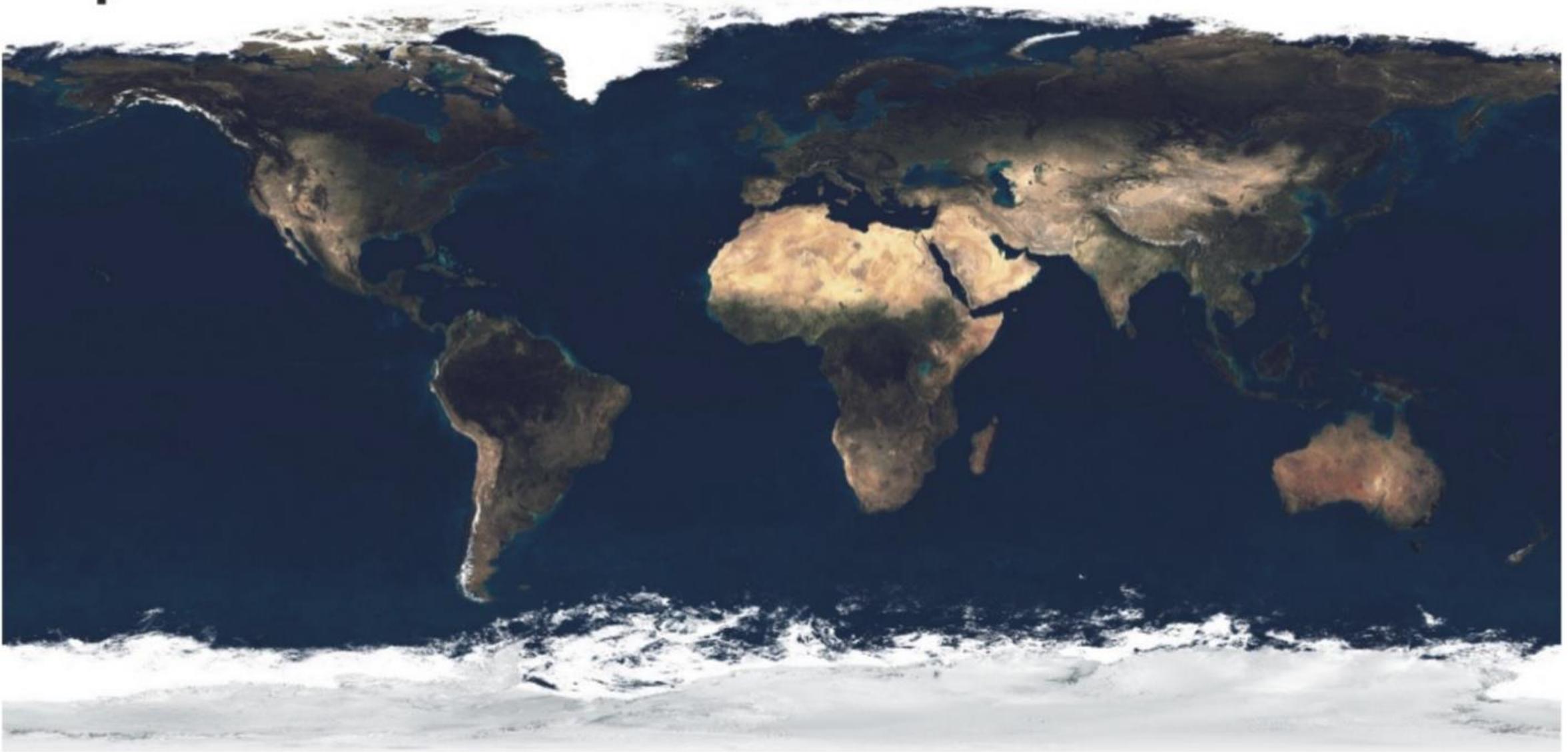
```
geometry = new THREE.SphereGeometry(100, 50, 50);

material = new THREE.MeshPhongMaterial({map: texture});

Player.mesh = new THREE.Mesh(geometry, material);

Player.scene.add(Player.mesh);
```

Sphere texture



Materials

Materials

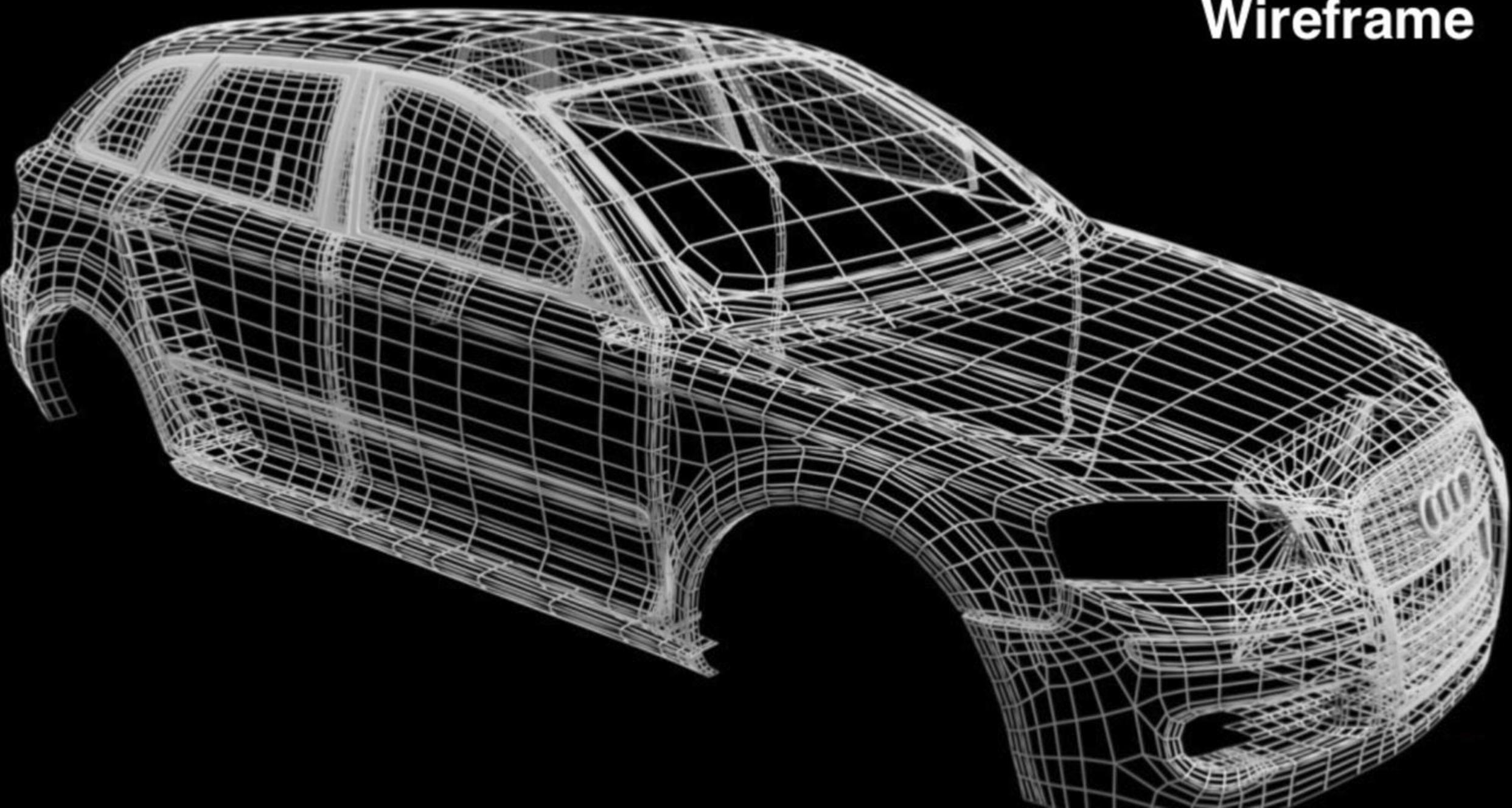


Materials

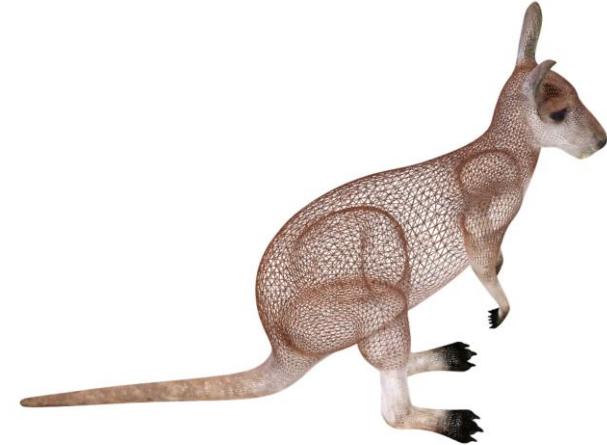
```
new THREE.MeshPhongMaterial({  
    color: option,  
    map: texture,  
    side: THREE.DoubleSide,  
});
```

```
object.material = new THREE.ShaderMaterial(  
    uniforms: {  
        u_Sampler: {type: "t", value: texture}  
    },  
    vertexShader: vShader,  
    fragmentShader: fShader,  
} );
```

Wireframe



Wireframe Code



```
const material = new THREE.MeshPhongMaterial({  
  map: texture,  
  wireframe: true  
});
```

Environment Map



Environment Map (Cube) Skybox

```
const urls = [
    require('../resources/textures/skybox/glass/posx.jpg'),
    require('../resources/textures/skybox/glass/negx.jpg'),
    require('../resources/textures/skybox/glass/posy.jpg'),
    require('../resources/textures/skybox/glass/negy.jpg'),
    require('../resources/textures/skybox/glass/posz.jpg'),
    require('../resources/textures/skybox/glass/negz.jpg'),
];
```

```
let materials = [];

for (let i = 0, length = urls.length; i < length; i++) {
    materials.push(new THREE.MeshBasicMaterial({
        map: new THREE.TextureLoader().load(urls[i]),
        side: THREE.BackSide
    }));
}

const skybox = new THREE.Mesh(
    new THREE.CubeGeometry(7000, 7000, 7000),
    new THREE.MultiMaterial(materials)
);
```

Environment Map (Cube) Model

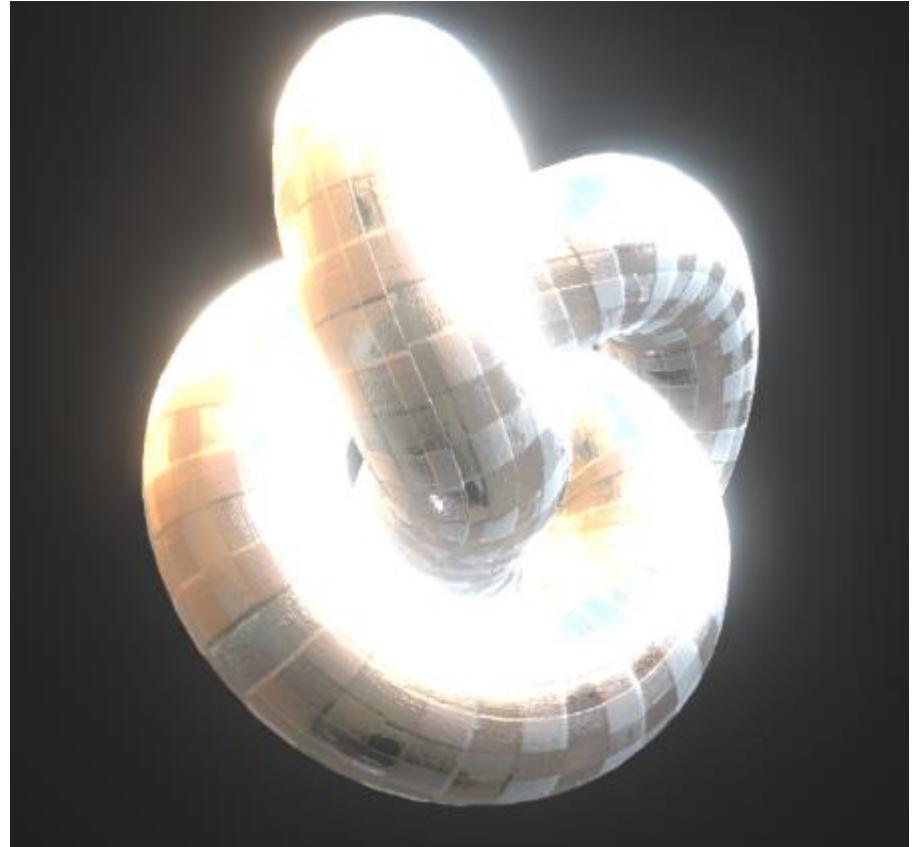
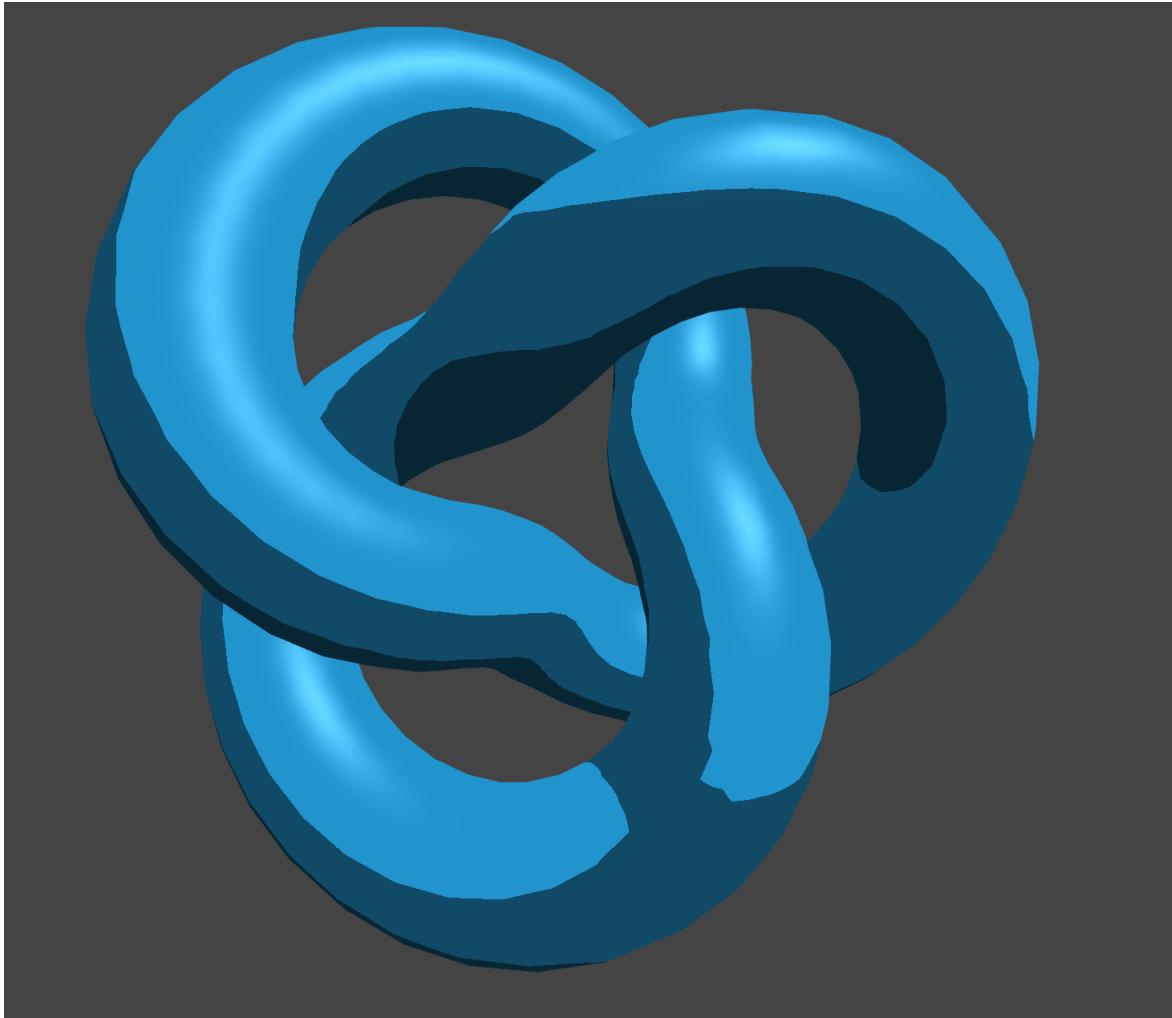


```
const urls = [
  require('../resources/textures/skybox/glass/posx.jpg'),
  require('../resources/textures/skybox/glass/negx.jpg'),
  require('../resources/textures/skybox/glass/posy.jpg'),
  require('../resources/textures/skybox/glass/negy.jpg'),
  require('../resources/textures/skybox/glass/posz.jpg'),
  require('../resources/textures/skybox/glass/negz.jpg'),
];

const textureCube = new THREE.CubeTextureLoader().load(urls);
textureCube.mapping = THREE.CubeRefractionMapping;
textureCube.format = THREE.RGBFormat;

return new THREE.MeshPhongMaterial({
  color: option === 'white' ? '#e6ffcc' : option,
  shininess: 35,
  emissive: 0x000000,
  specular: 0x333333,
  envMap: textureCube,
  reflectivity: 0.95,
  refractionRatio: 0.95,
  side: THREE.DoubleSide,
  polygonOffset: true,
  polygonOffsetFactor: 1,
  polygonOffsetUnits: 1,
  transparent: true,
  opacity: 1
});
```

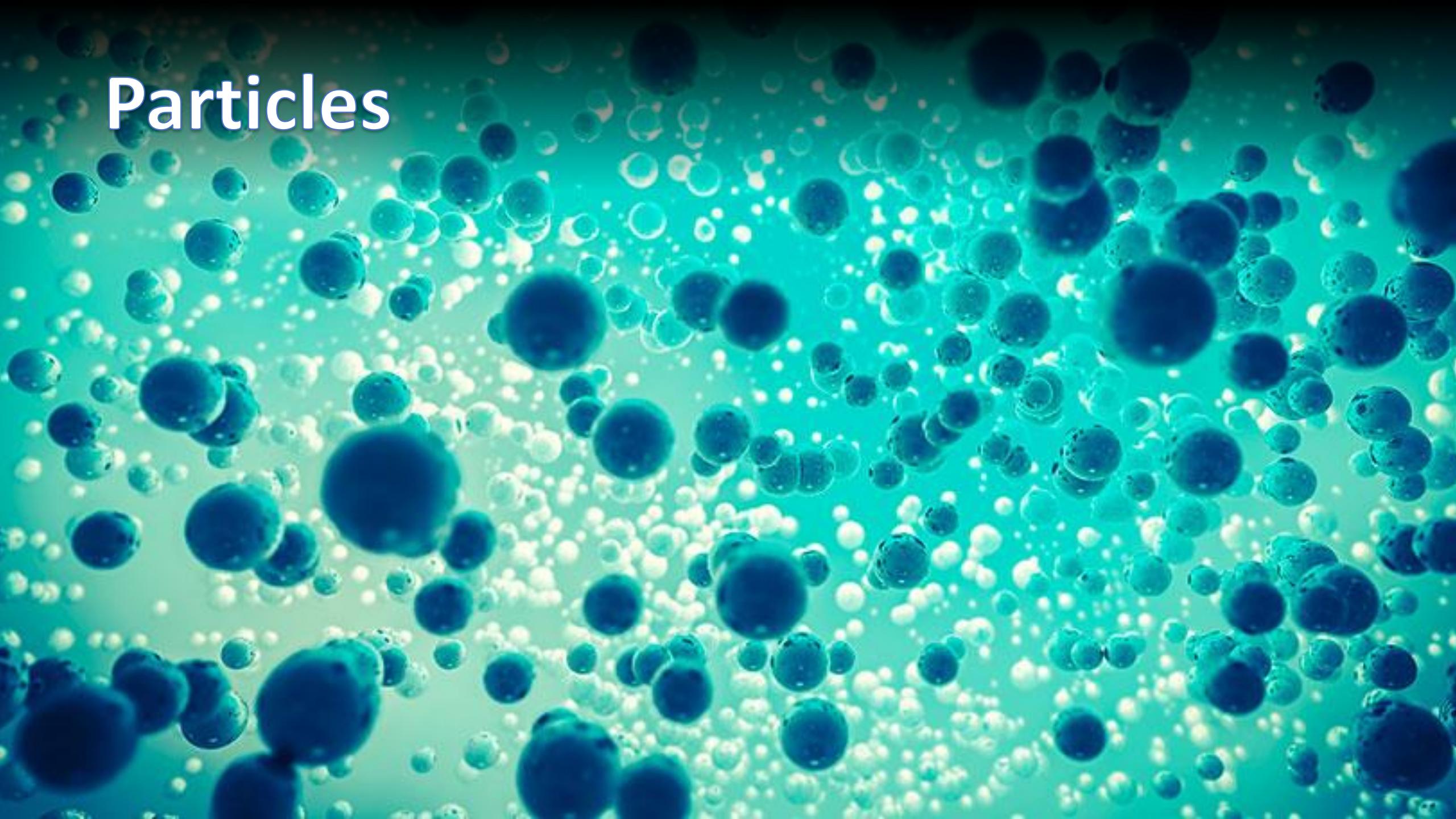
Post-Processing



Post-Processing

```
composer = new THREE.EffectComposer(renderer);
composer.addPass( new THREE.RenderPass(scene, camera) );
Toon = { uniforms: uniforms,
          vertexShader: vertexShader,
          fragmentShader: fragmentShader,
        };
effect = new THREE.ShaderPass(Toon);
effect.renderToScreen = true;
composer.addPass(effect);
```

Particles



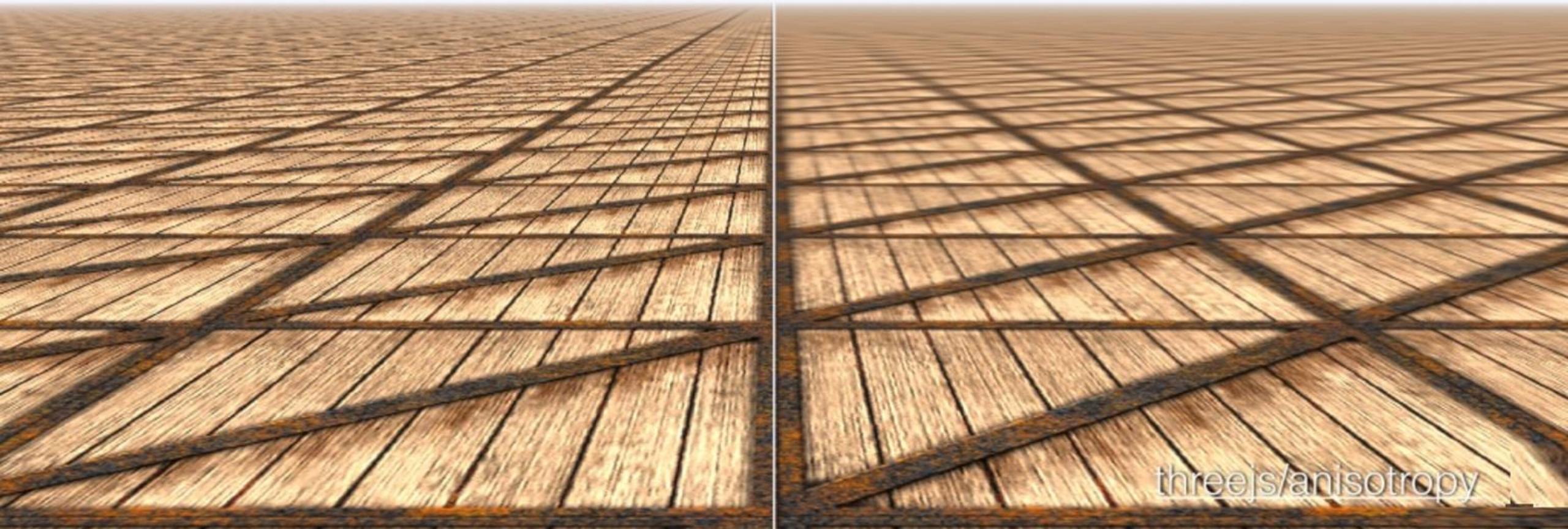
Particles

```
particleObj = new THREE.ParticleSystem(geometry, material);
particleObj.position.set(0, 50, 0);
particleObj.dynamic = true;
particleObj.sortParticles = true;
scene.add( particleObj );
```

Particles

oos.moxiecode.com

Texture Quality/Filtering



threejs/anisotropy

Anisotropic Filtering

A method for enhancing the **image quality** of textures on surfaces that are at oblique viewing angles with respect to the camera

```
const renderer = new THREE.WebGLRenderer();
const maxAnisotropy = renderer.getMaxAnisotropy();

const textureLoader = new THREE.TextureLoader();
const texture1 = textureLoader.load('texture1.png');
texture1.anisotropy = maxAnisotropy;
```

Review

Pros

No compilation

High performance (GPU)

Cross-platform

Open standard

Automatic memory management

Cons

Poly-polygonal scenes

Optimization required

Raster graphics

Interactivity



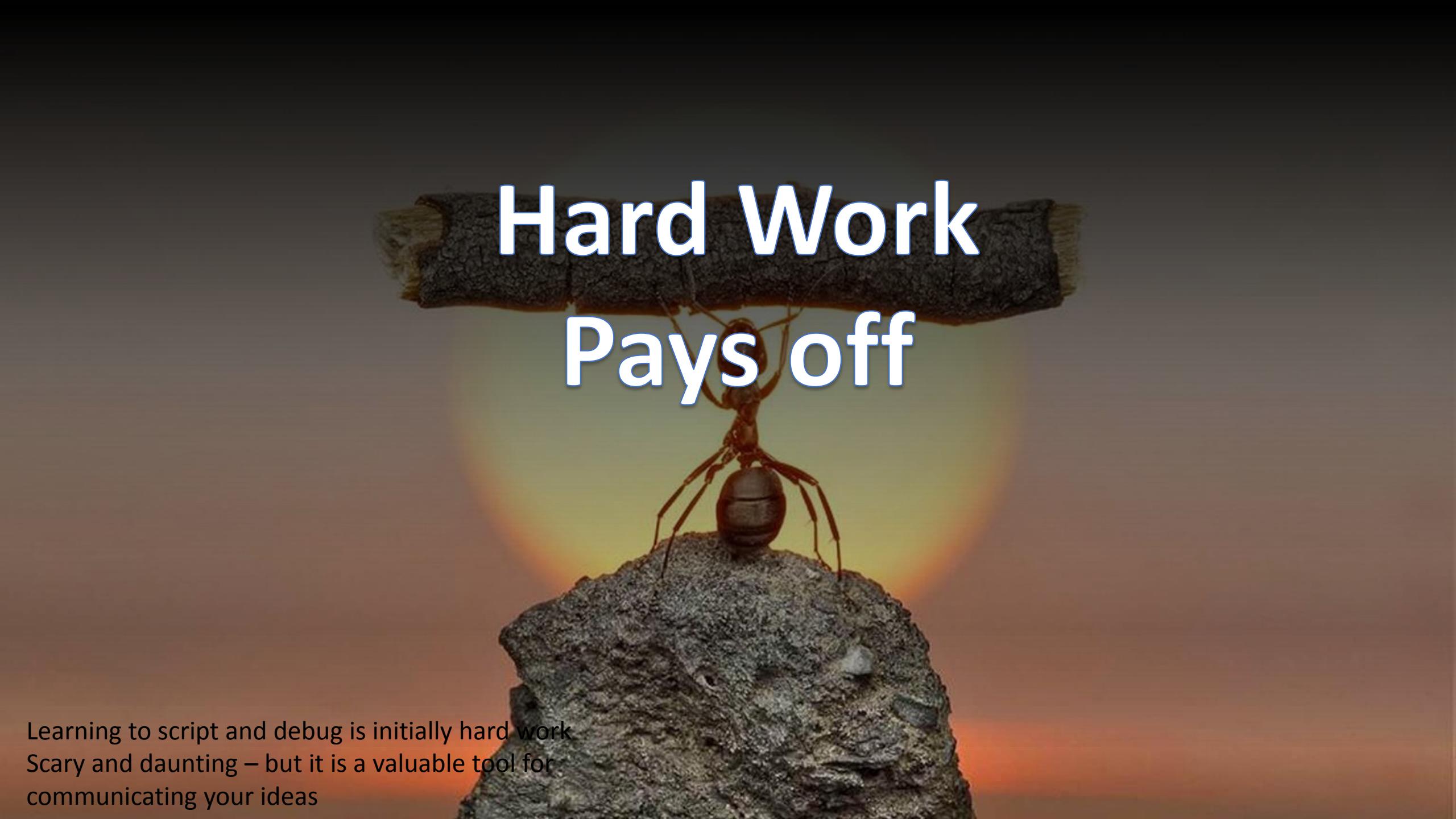
Useful Links/Resources

Three.js

- <https://threejs.org/>

Visualization

- <http://mariovalle.name/visualization/VizArt.html>
- <https://www.psychologytoday.com/us/blog/flourish/200912/seeing-is-believing-the-power-visualization>
- <https://www.nature.com/articles/487430a>



Hard Work Pays off

Learning to script and debug is initially hard work
Scary and daunting – but it is a valuable tool for
communicating your ideas



Questions & Discussion

Q&A

- Do you need to be connected to the internet to use Three.js

No, Three.js runs in the browser

- Should you learn WebGL before you learn Three.js?

Learn Three.js, and then if your interest wants to go deeper, the internals of Three.js are a fantastic learning resource for how they accomplish everything on top of raw WebGL

How to run things locally (browser/debugging)

File access blocked by CORS policy

One solution (Python)

```
/*
// Workaround for local testing/file loading in browser
/> python -m http.server 8000 --bind 127.0.0.1
```

Run this command from inside the directory, see the results by accessing
<http://localhost:8000/>

```
*/
```

