three.js

Gerard Antony Caramazza Vilá (gerard.caramazza.vila.15@ull.edu.es)
Edwin Plasencia Hernández (edwin.plasencia.28@ull.edu.es)

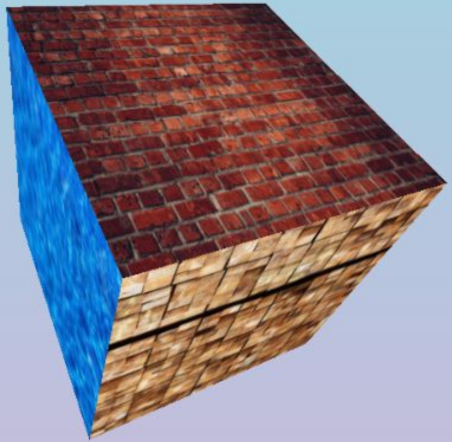
Index

Introduction

What is Three.js?

Three vs WebGL

What can we do with it?



Abstraction

Low level

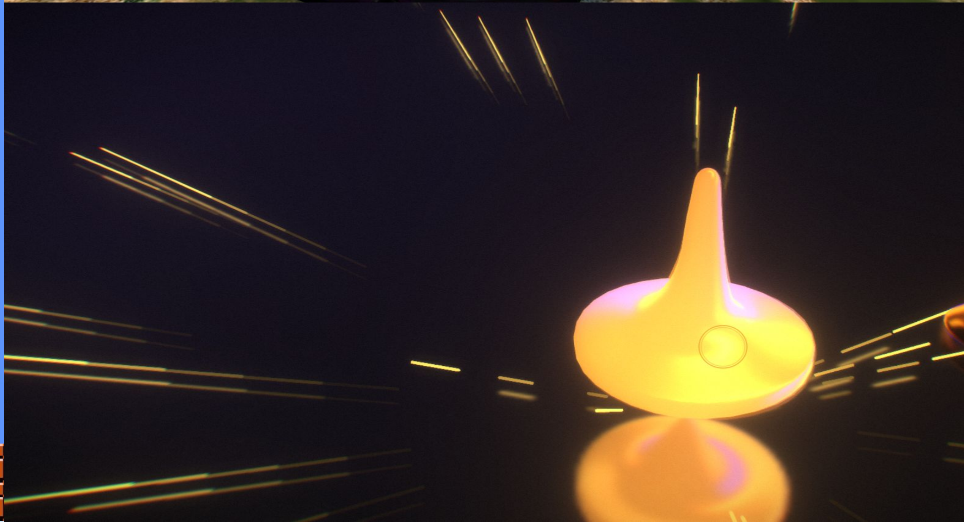
High level

WebGL™

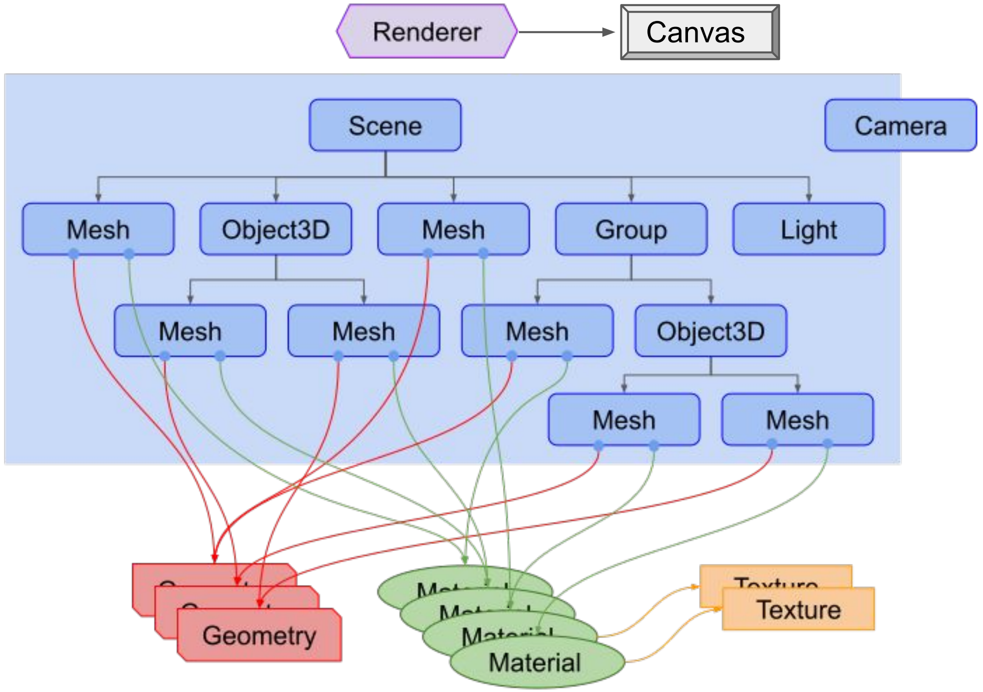
three.js

7 / 19
Trigger Rally
ERM 954
51 km/h
0:37.73

MARIO
000000
×00
WORLD
1-1
TIME
JELLY
MARIO BROS.
USE ARROW KEYS!
EAT TO START!

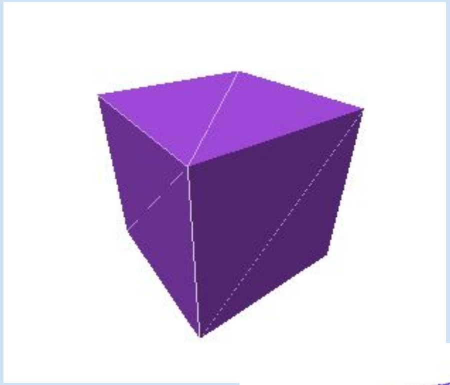
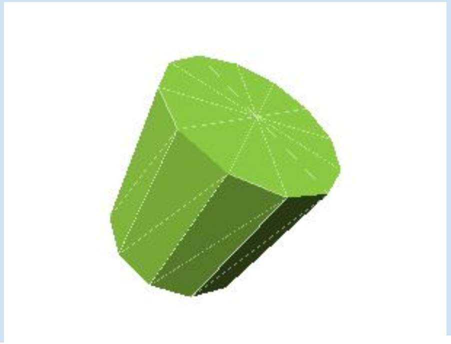
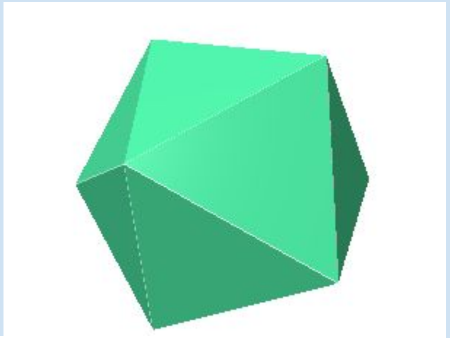


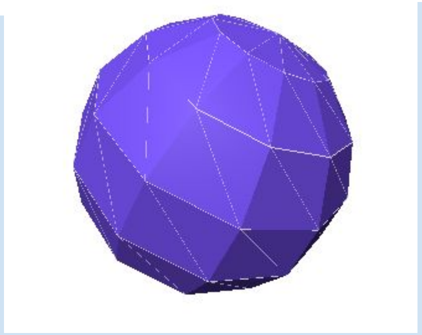
How does it work?

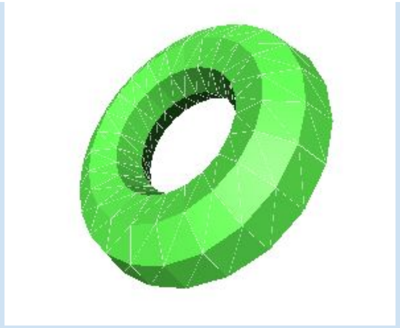


Geometry

Shape of the object

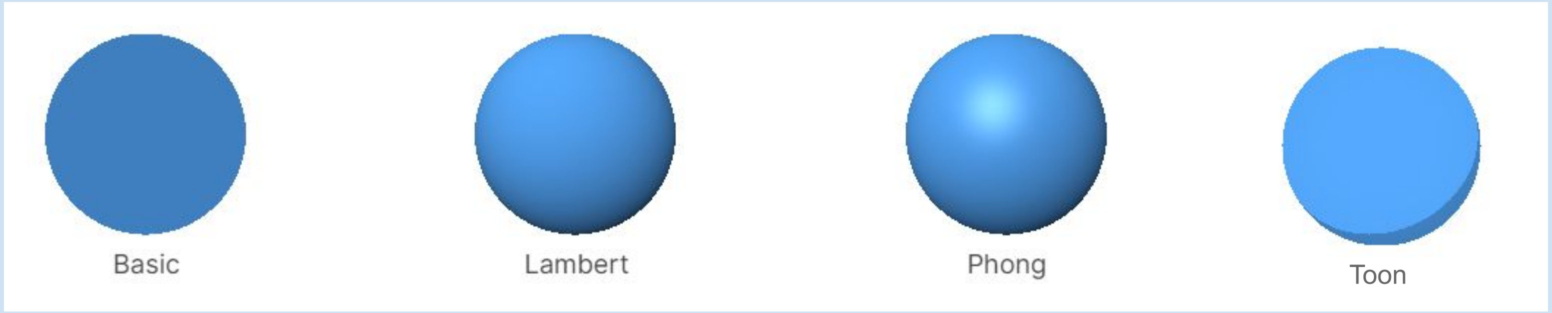






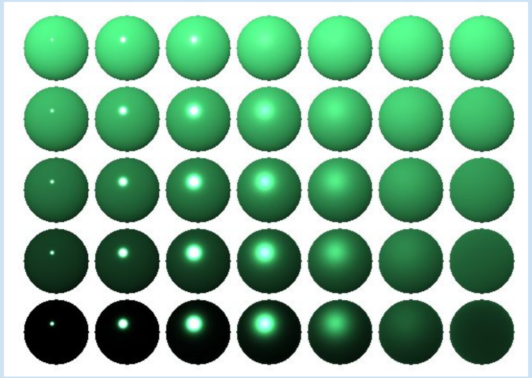


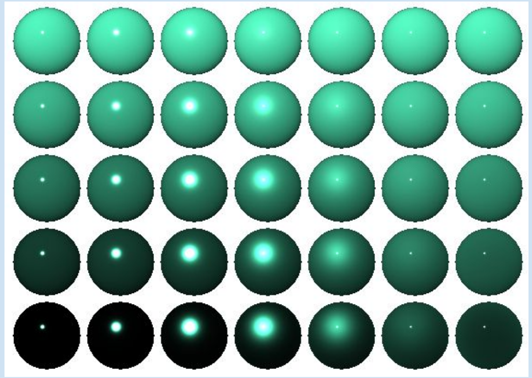
Material

Surface properties of the object



Standard

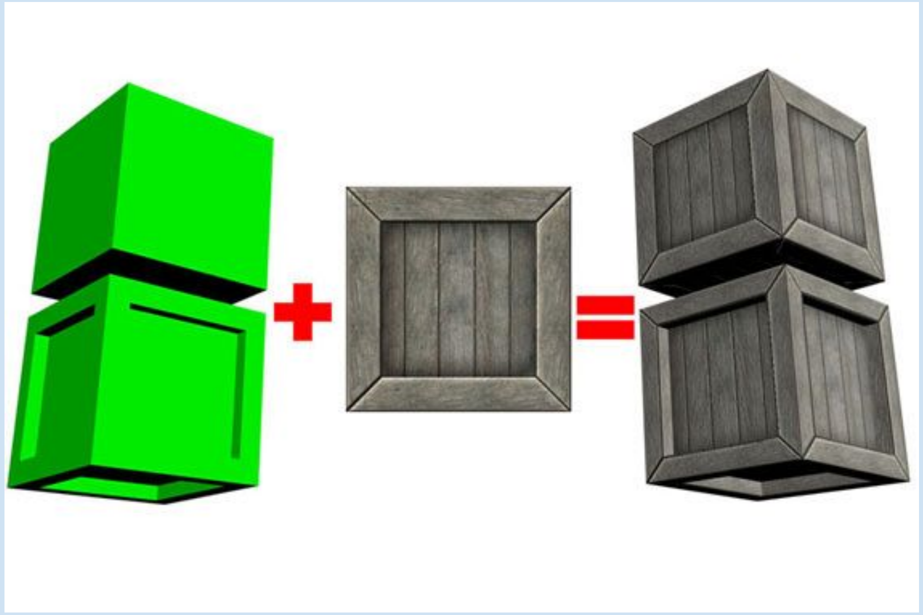




Clear Coat

Textures

Surface appearance of the object



Getting started

Installation

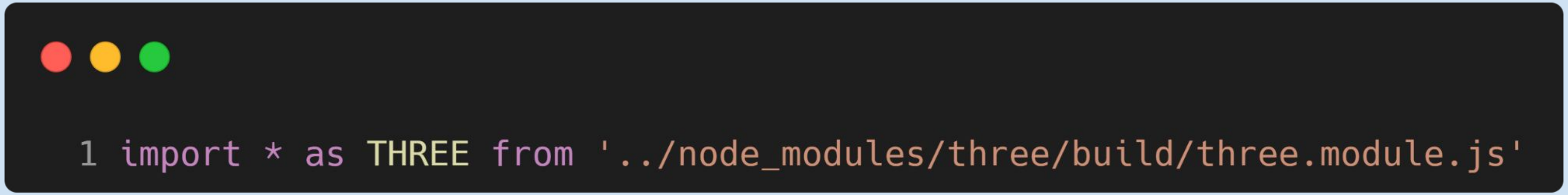
```
npm install three
```

Basic webpage

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Test</title>
    <script type="module" defer src="../src/index.js"></script>
  </head>
  <body>
    <canvas id="canvasBase" width="1920" height="965"></canvas>
  </body
</html>
```

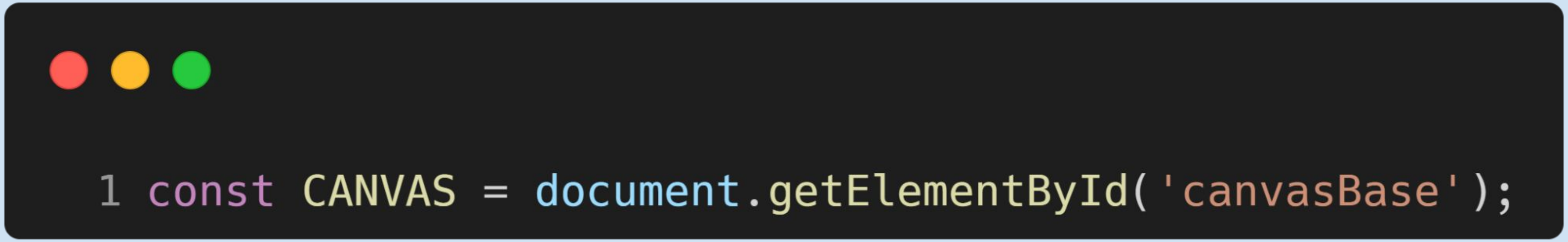
Our script

Import



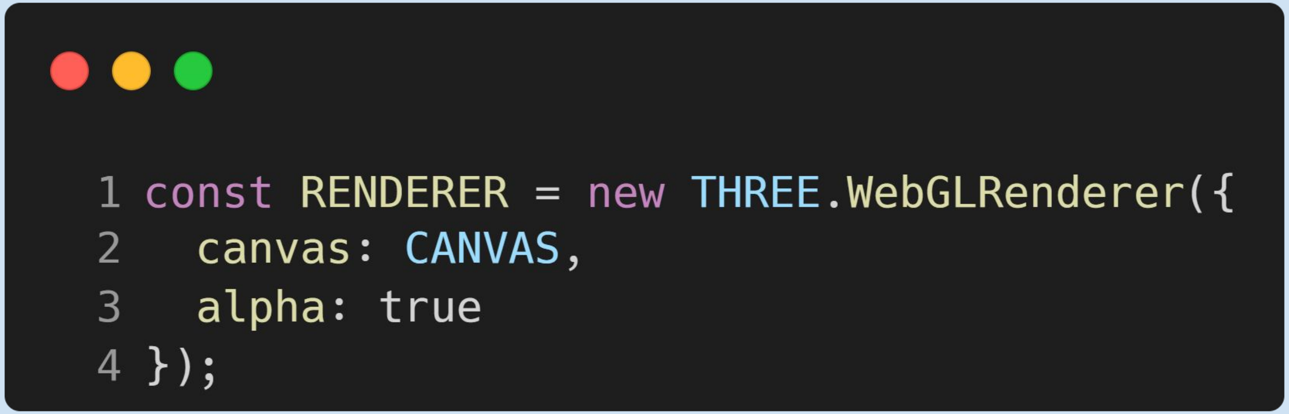
```javascript
import * as THREE from '../node_modules/three/build/three.module.js'
```

Getting our canvas

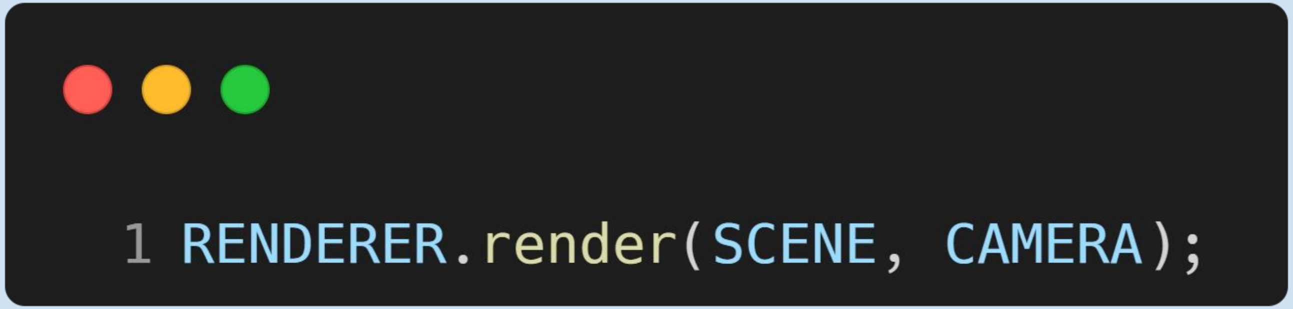


```javascript
const CANVAS = document.getElementById('canvasBase');
```

Creating our renderer



```js
const RENDERER = new THREE.WebGLRenderer({
  canvas: CANVAS,
  alpha: true
});
```



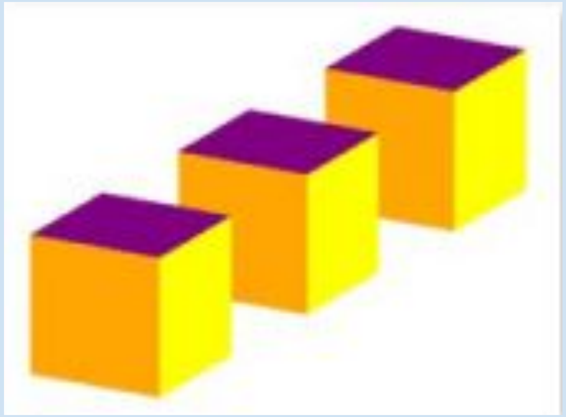
```js
RENDERER.render(SCENE, CAMERA);
```

Parameters[1]:

- canvas
- alpha
- antialias
- precision
- and more…

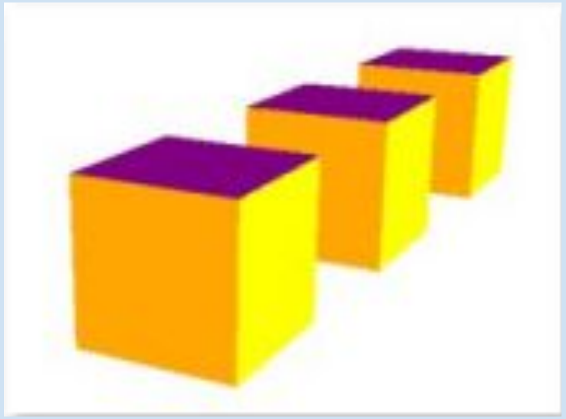
Creating our camera

Orthographic camera



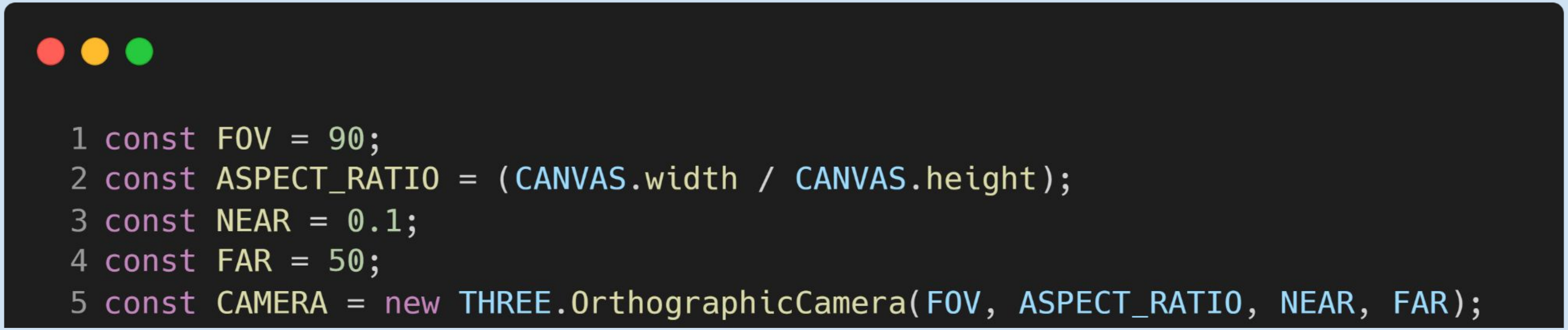
`OrthographicCamera()`

Perspective camera



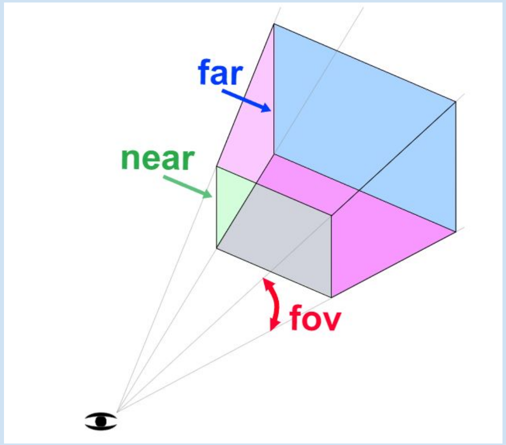
`PerspectiveCamera()`

Creating our camera



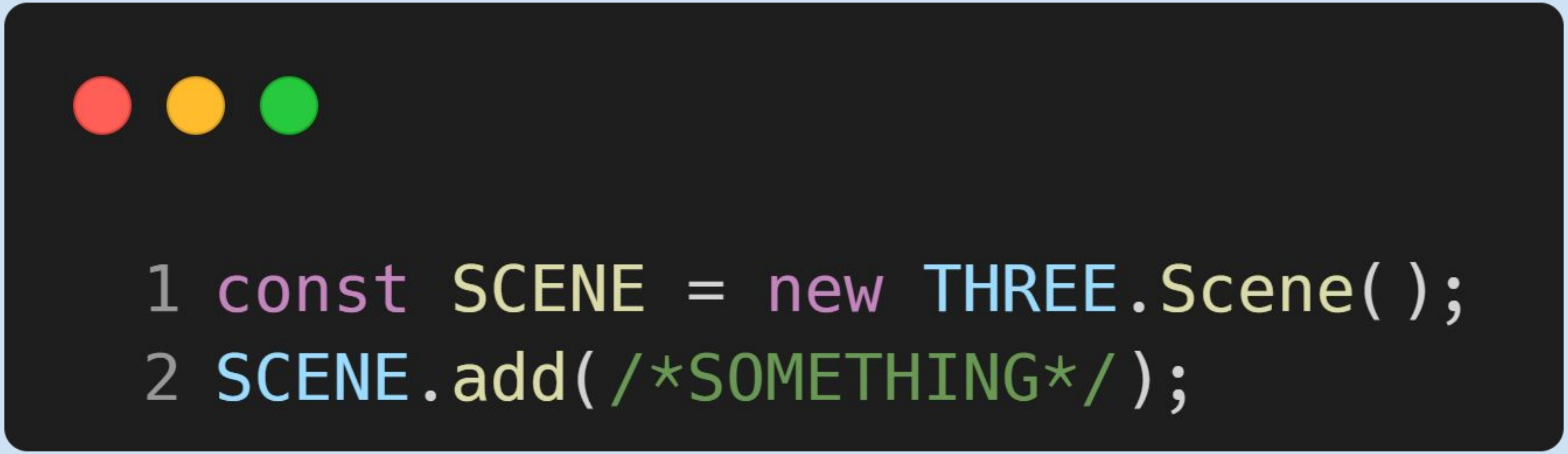
```
const FOV = 90;
const ASPECT_RATIO = (CANVAS.width / CANVAS.height);
const NEAR = 0.1;
const FAR = 50;
const CAMERA = new THREE.OrthographicCamera(FOV, ASPECT_RATIO, NEAR, FAR);
```

FOV
ASPECT_RATIO
NEAR
FAR



Creating a scene

What will be rendered by the renderer

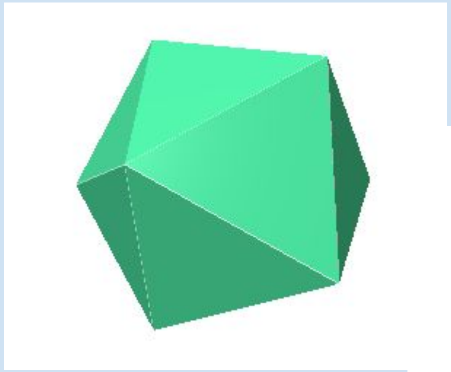


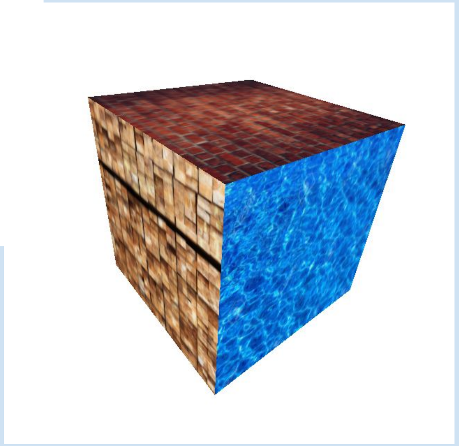
```
const SCENE = new THREE.Scene();
SCENE.add(/*SOMETHING*/);
```

SOMETHING = Objects and/or lights

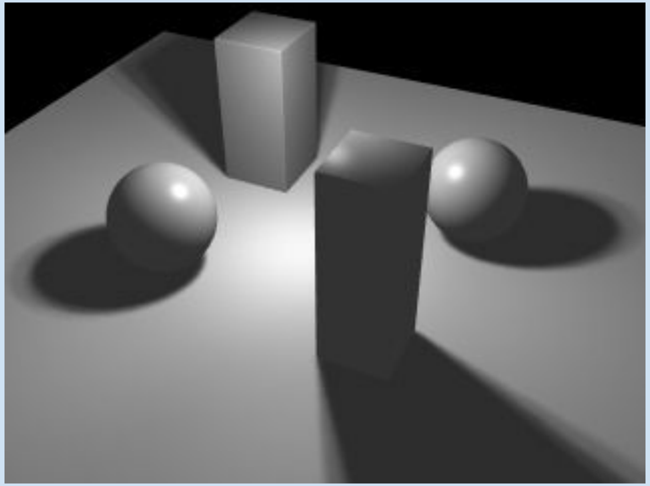
Adding things to our scene

Objects





Lights



Objects – Geometry

SphereGeometry(radius, widthSegments, heightSegments)

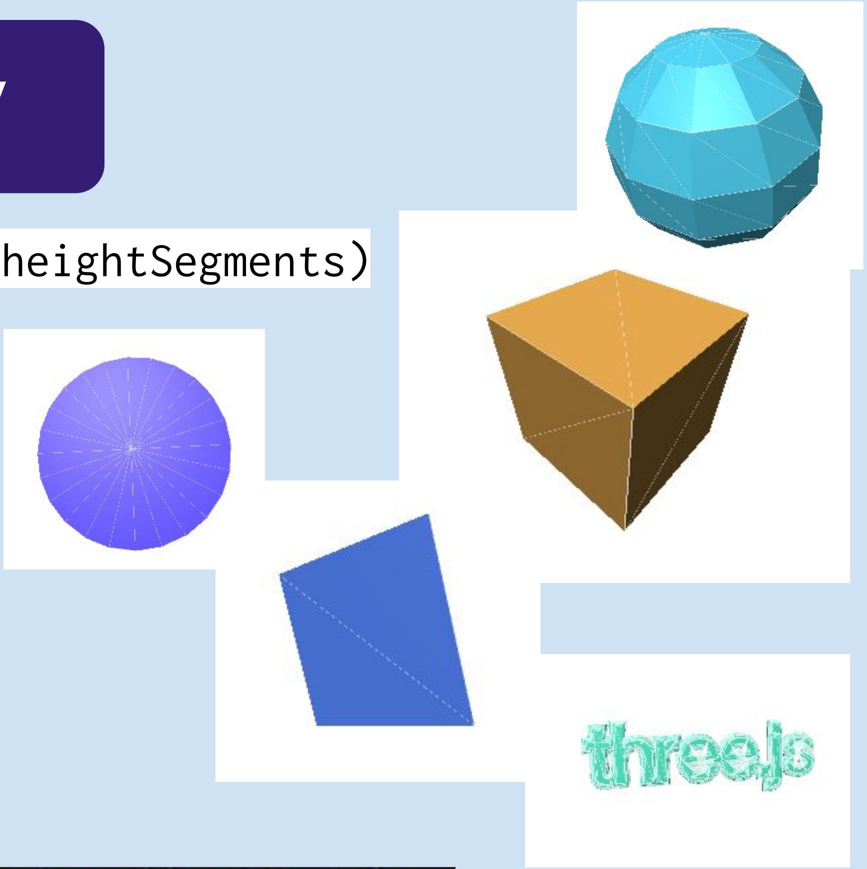
PlaneGeometry(width, height)

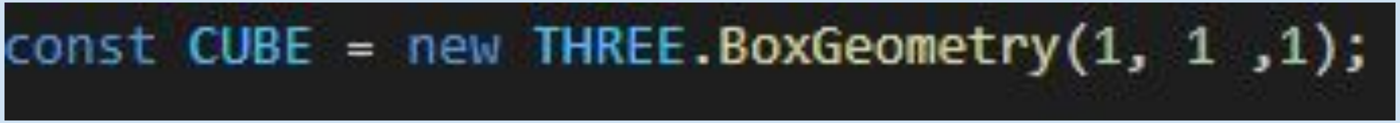
CircleGeometry(radius, segments)

BoxGeometry(width, height, depth)

TextGeometry(text, {font, size, etc})

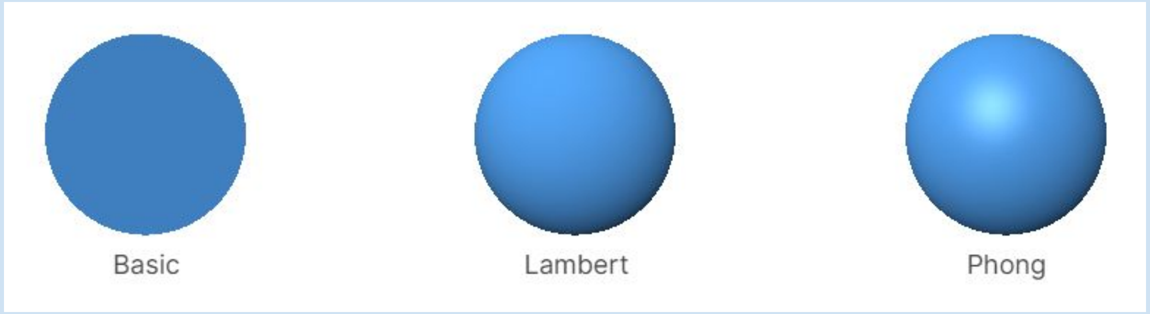
and more…





```
const CUBE = new THREE.BoxGeometry(1, 1 ,1);
```

Objects - Materials



`MeshBasicMaterial()`

No light effects

`MeshLambertMaterial()`

Light effects

only on vertices

`MeshPhongMaterial()`

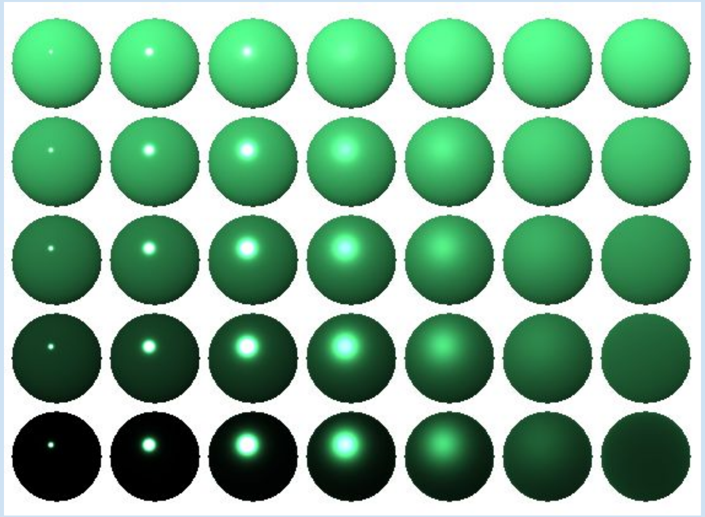
Light effects

everywhere

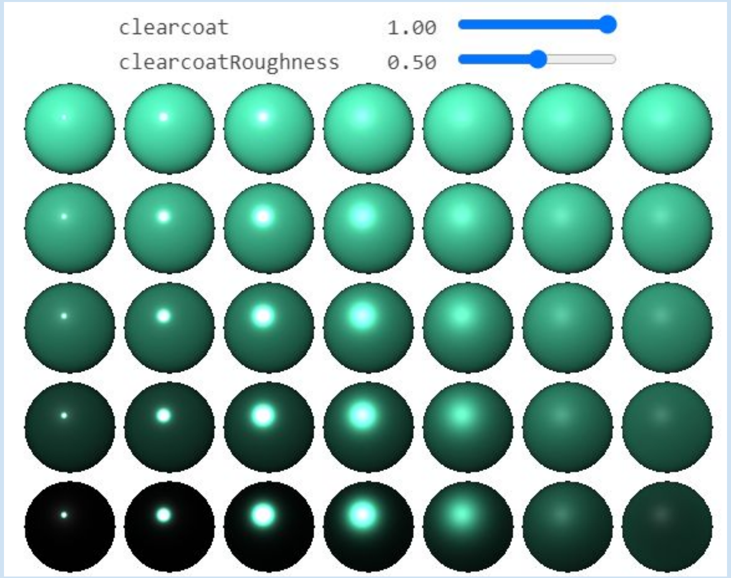
Objects – PBR Materials

Roughness

Metalness



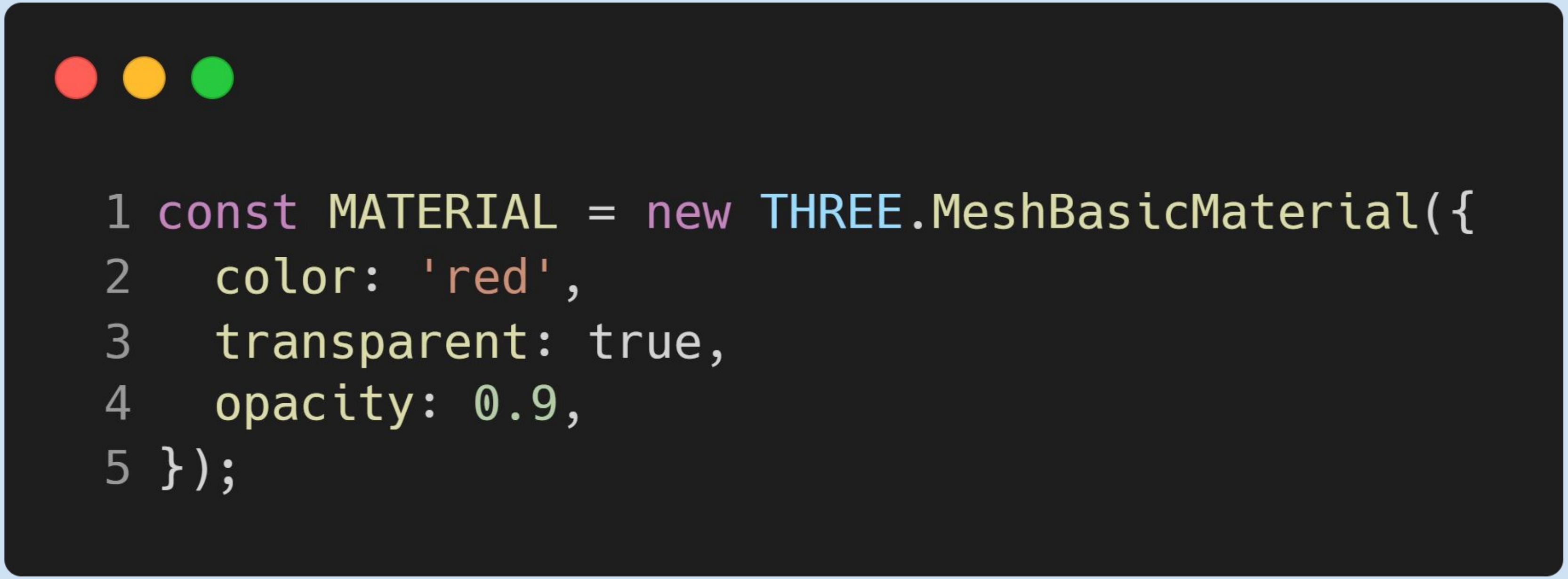
MeshStandardMaterial()



MeshClearCoatMaterial()

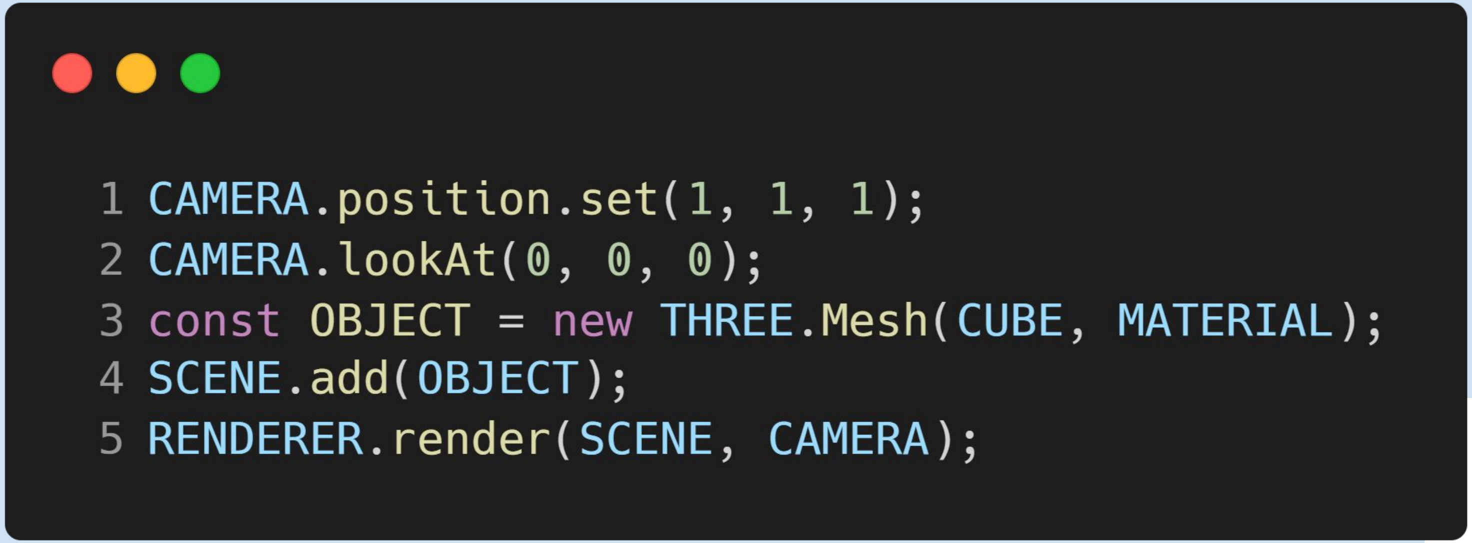
Objects – Materials

Our example:

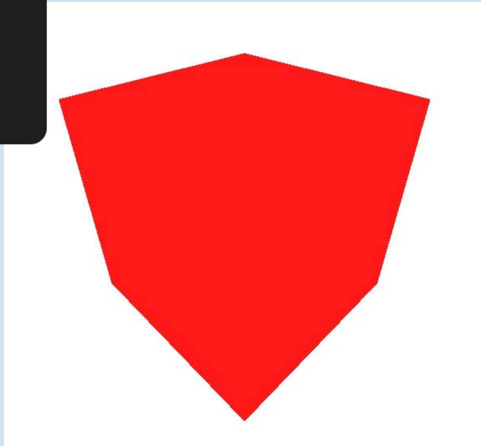


```javascript
const MATERIAL = new THREE.MeshBasicMaterial({
  color: 'red',
  transparent: true,
  opacity: 0.9,
});
```

Rendering the scene

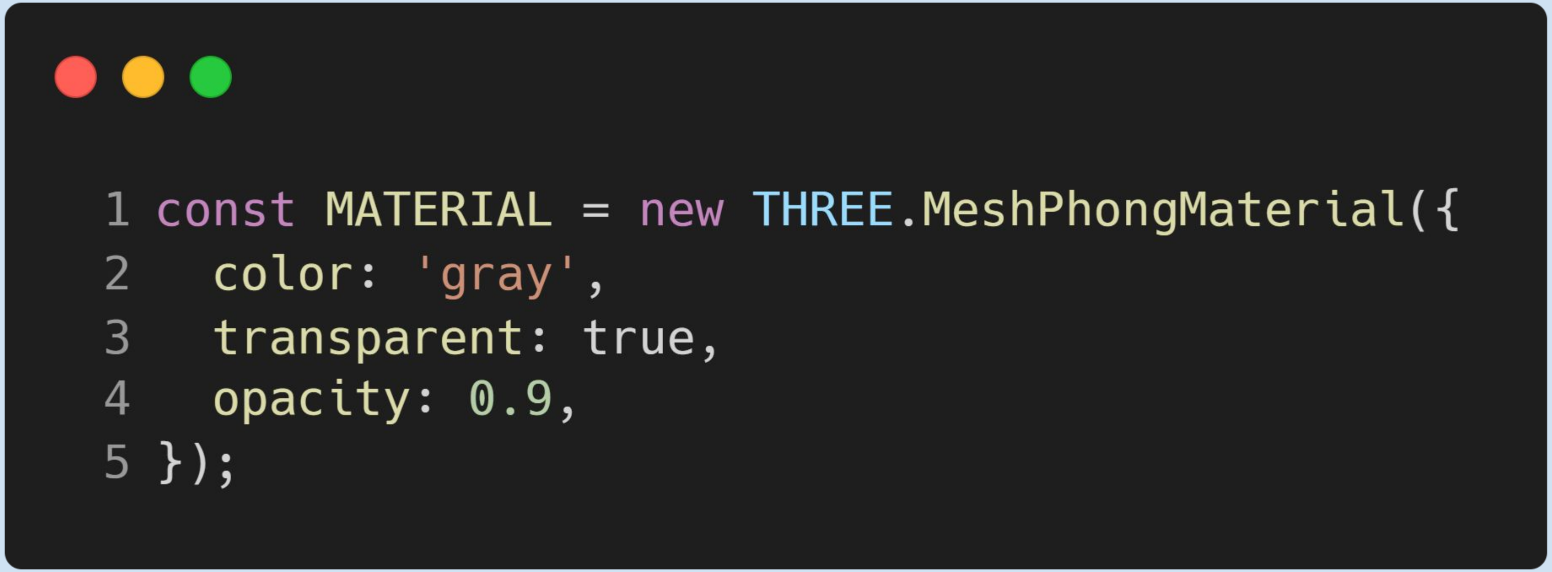


```javascript
CAMERA.position.set(1, 1, 1);
CAMERA.lookAt(0, 0, 0);
const OBJECT = new THREE.Mesh(CUBE, MATERIAL);
SCENE.add(OBJECT);
RENDERER.render(SCENE, CAMERA);
```



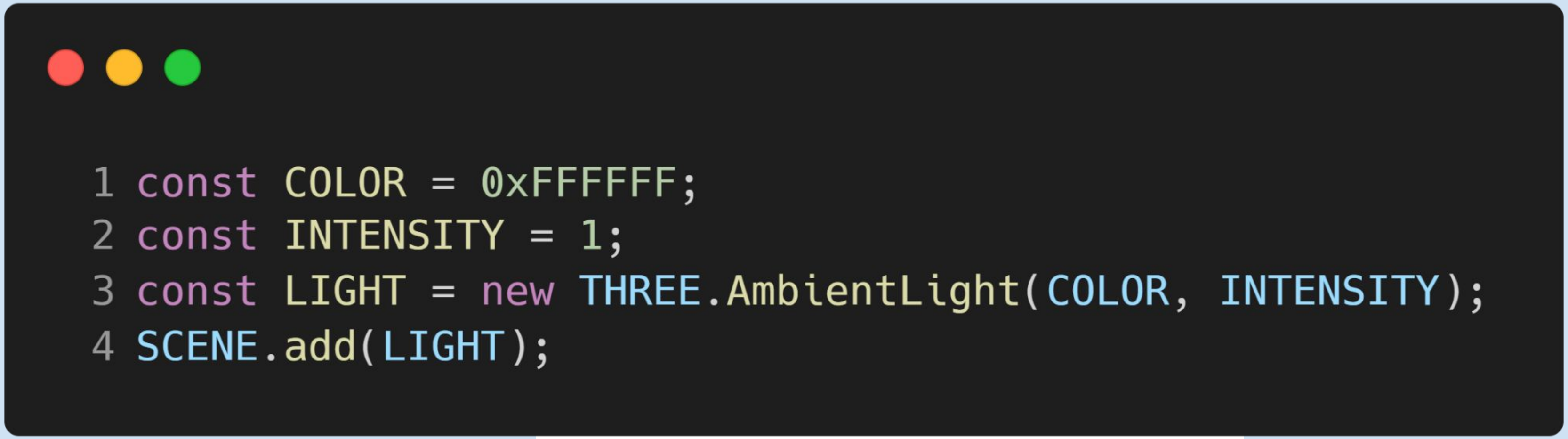
Let’s spice it up - Lights

First let’s change the material

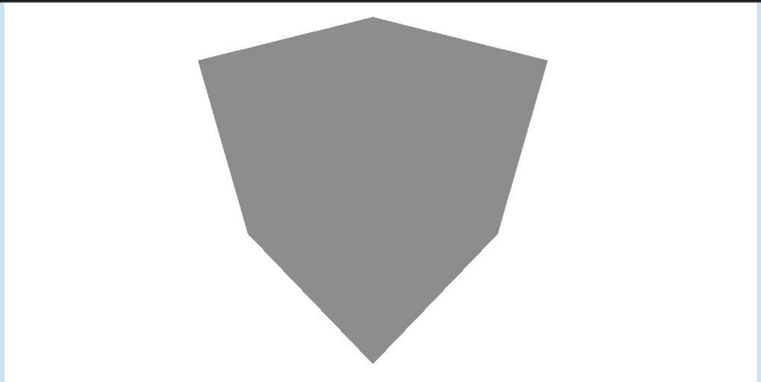


```
const MATERIAL = new THREE.MeshPhongMaterial({
  color: 'gray',
  transparent: true,
  opacity: 0.9,
});
```

Lights - Ambient light

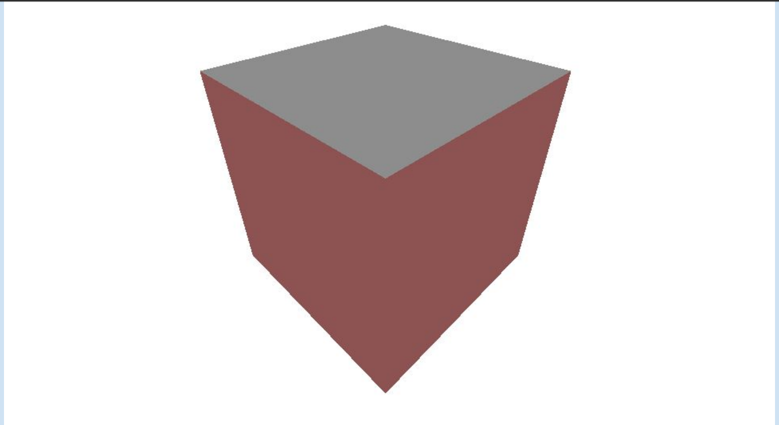


```javascript
const COLOR = 0xFFFFFF;
const INTENSITY = 1;
const LIGHT = new THREE.AmbientLight(COLOR, INTENSITY);
SCENE.add(LIGHT);
```

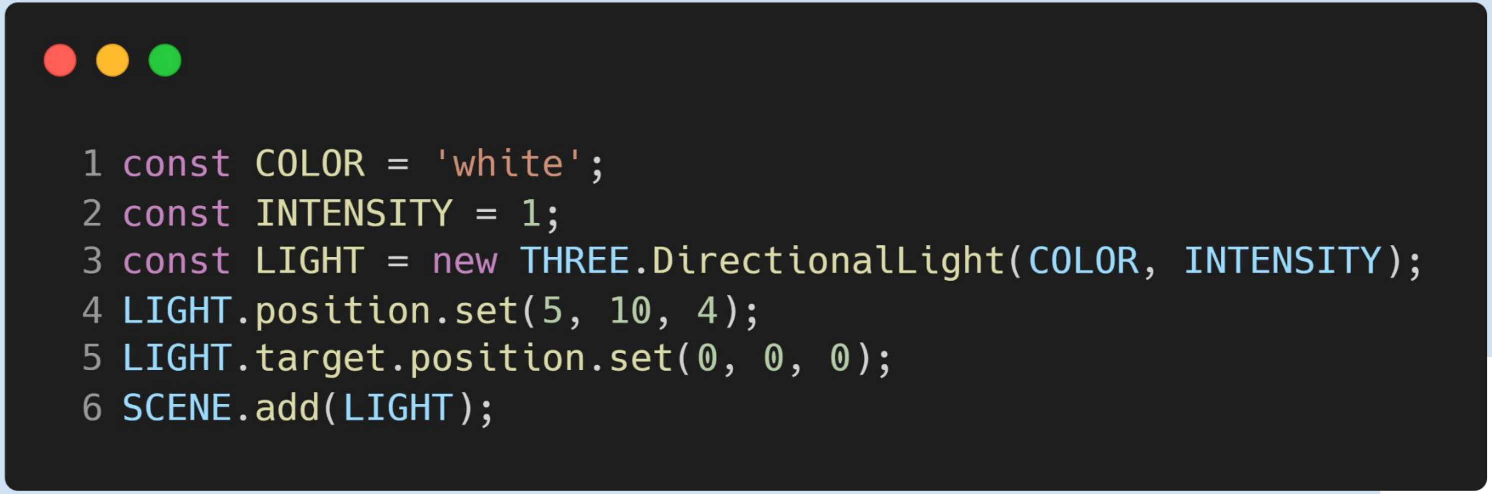


Lights - Hemisphere light

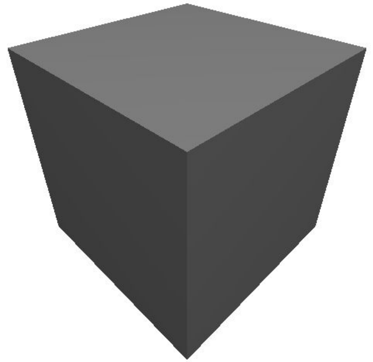
```javascript
const COLOR_SKY = 'white';
const COLOR_GROUND = 'red';
const INTENSITY = 1;
const LIGHT = new THREE.HemisphereLight(COLOR_SKY, COLOR_GROUND, INTENSITY);
SCENE.add(LIGHT);
```



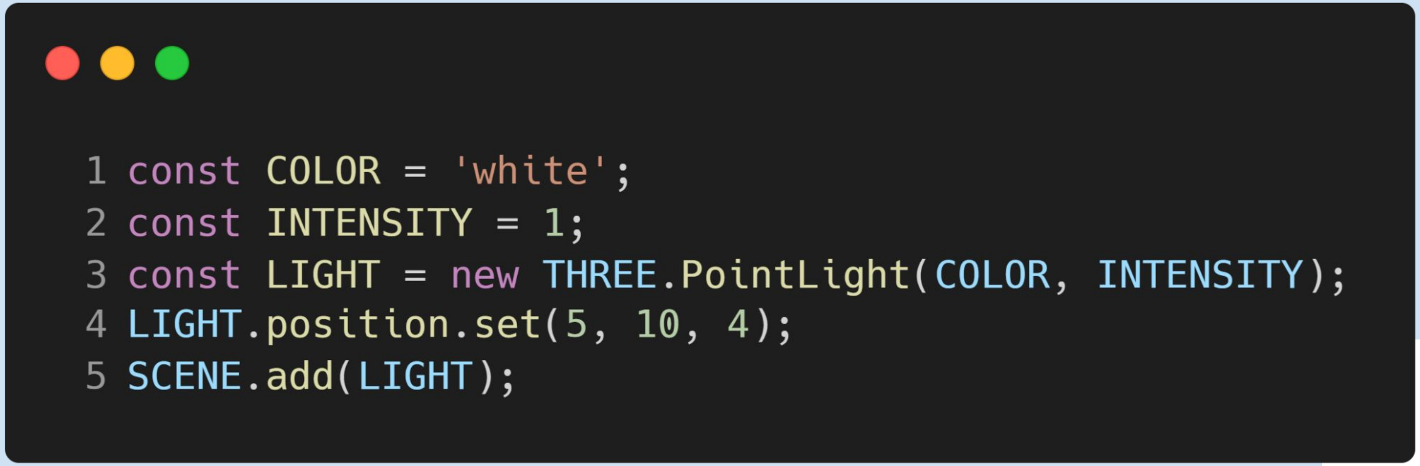
Lights - Directional light



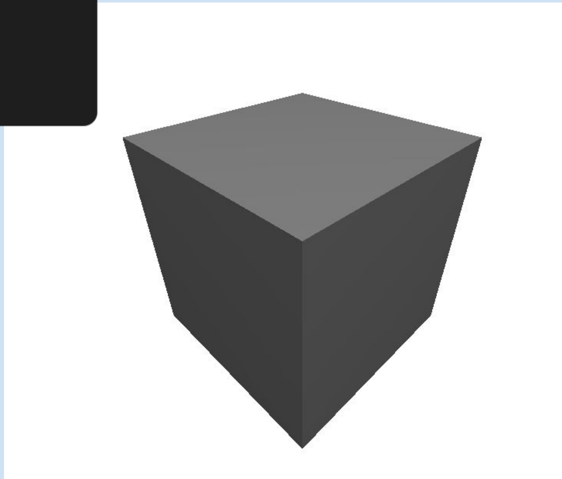
```
const COLOR = 'white';
const INTENSITY = 1;
const LIGHT = new THREE.DirectionalLight(COLOR, INTENSITY);
LIGHT.position.set(5, 10, 4);
LIGHT.target.position.set(0, 0, 0);
SCENE.add(LIGHT);
```



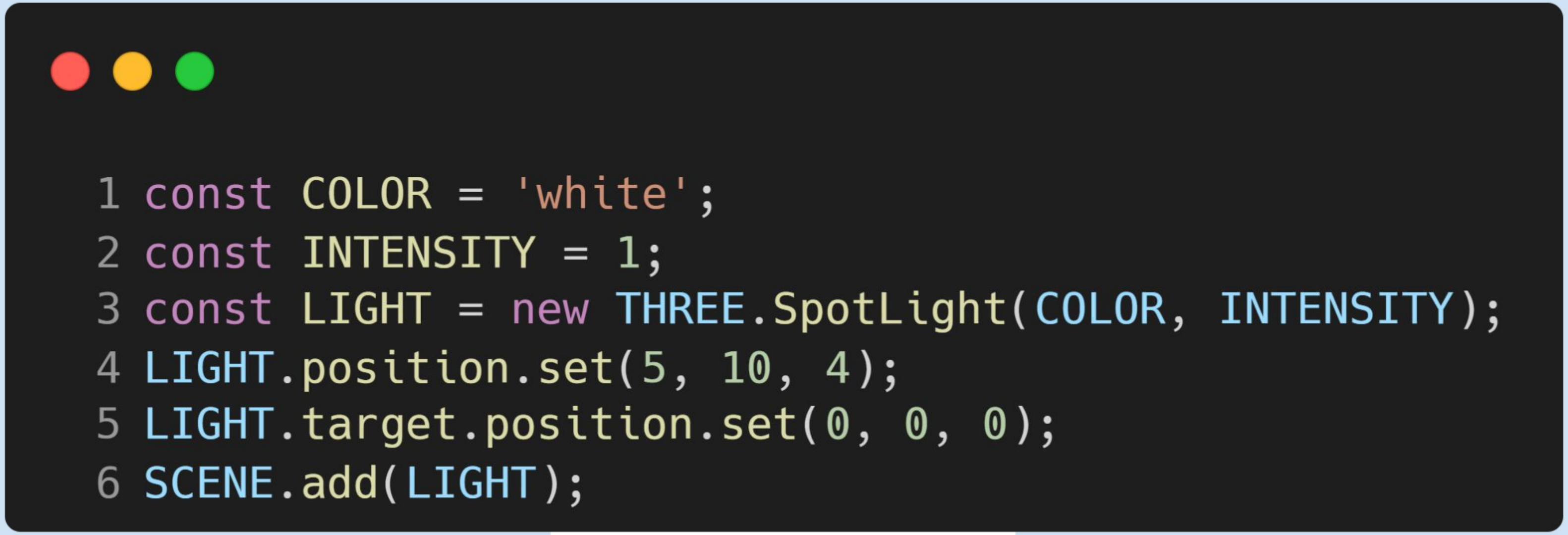
Lights - Point light



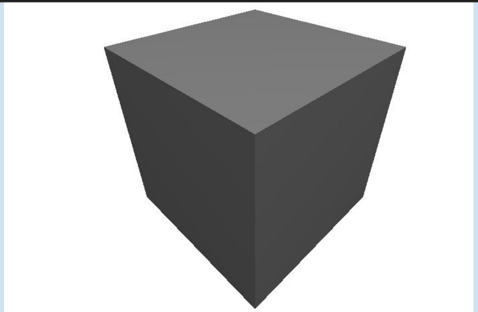
```
const COLOR = 'white';
const INTENSITY = 1;
const LIGHT = new THREE.PointLight(COLOR, INTENSITY);
LIGHT.position.set(5, 10, 4);
SCENE.add(LIGHT);
```



Lights - Spot light



```javascript
const COLOR = 'white';
const INTENSITY = 1;
const LIGHT = new THREE.SpotLight(COLOR, INTENSITY);
LIGHT.position.set(5, 10, 4);
LIGHT.target.position.set(0, 0, 0);
SCENE.add(LIGHT);
```

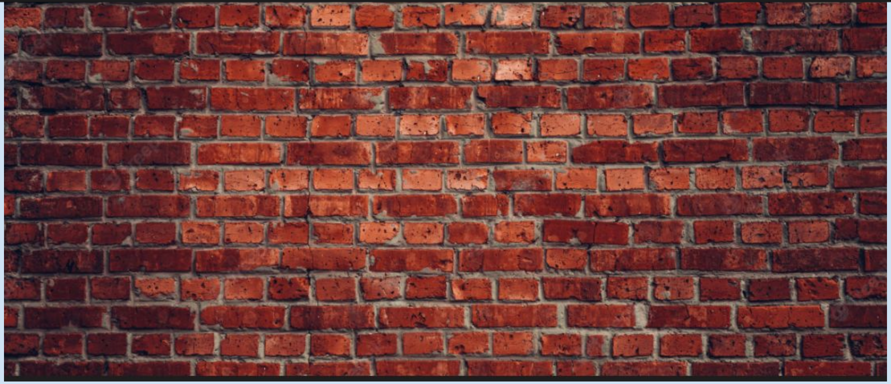


Textures

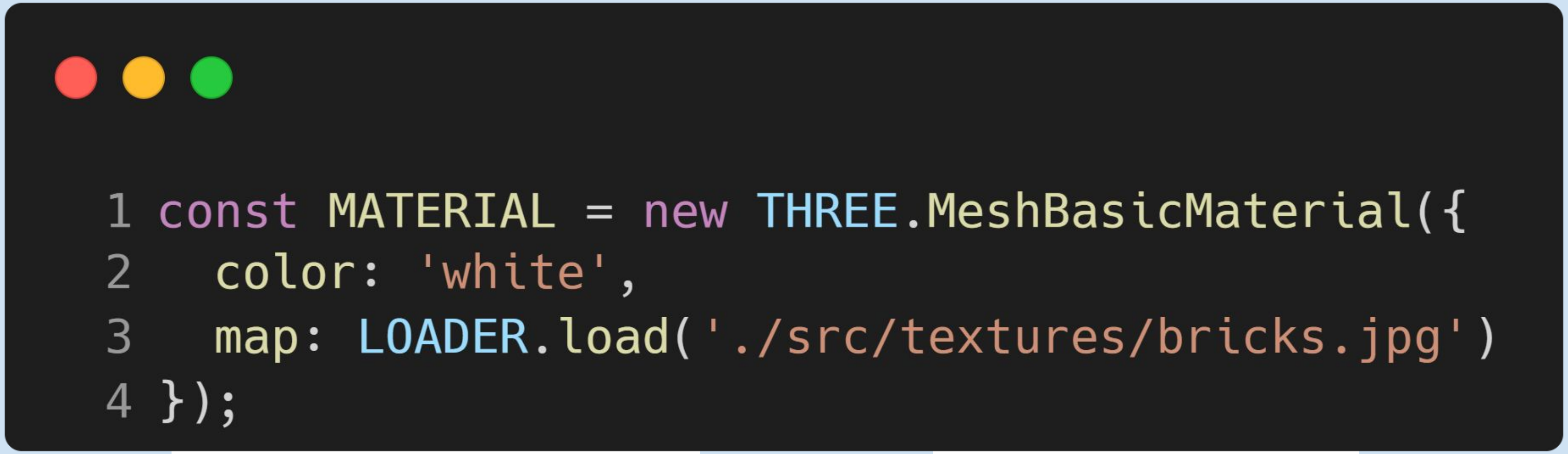
Texture loader

```
const LOADER = new THREE.TextureLoader();
```

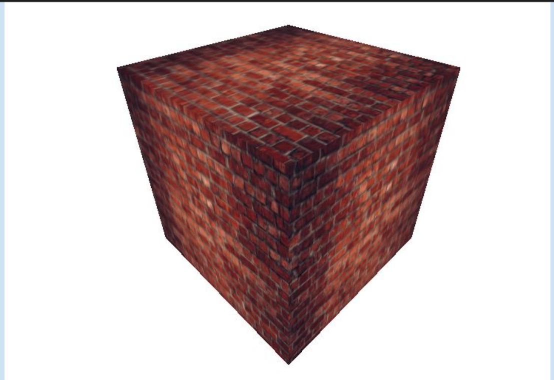
Our texture:



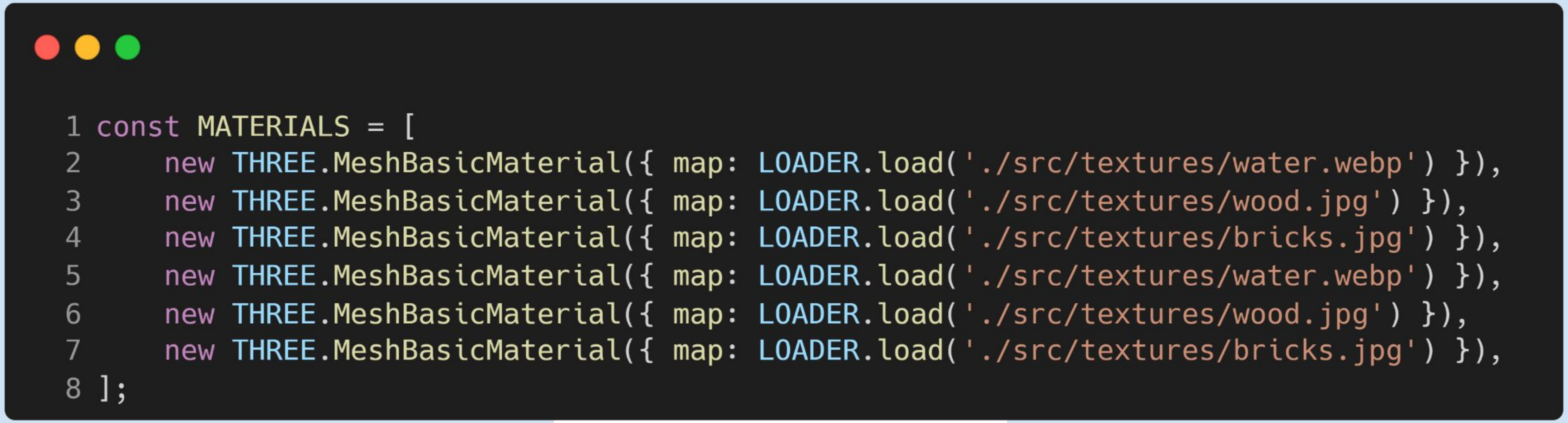
Applying a texture to a material



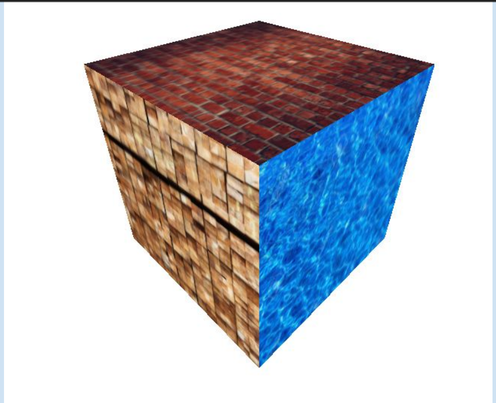
```javascript
const MATERIAL = new THREE.MeshBasicMaterial({
  color: 'white',
  map: LOADER.load('./src/textures/bricks.jpg')
});
```



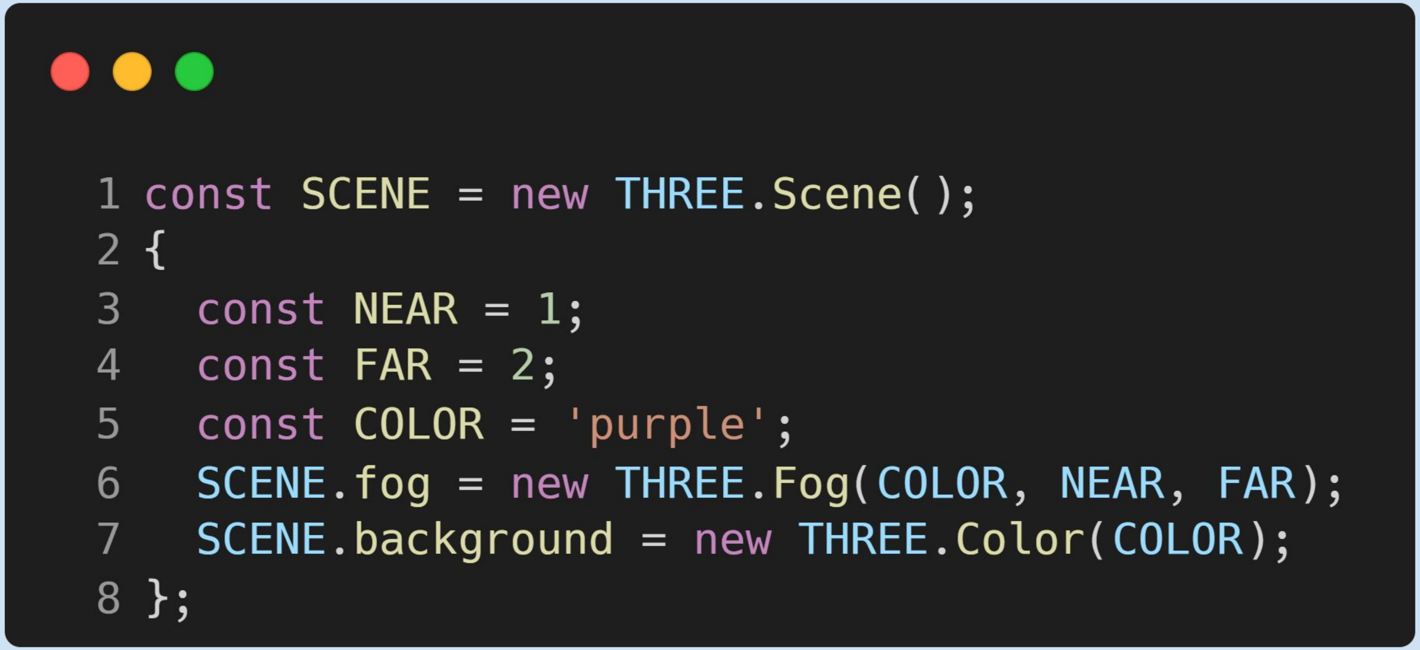
Applying a texture to a material



```
const MATERIALS = [
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/water.webp') }),
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/wood.jpg') }),
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/bricks.jpg') }),
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/water.webp') }),
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/wood.jpg') }),
    new THREE.MeshBasicMaterial({ map: LOADER.load('./src/textures/bricks.jpg') }),
];
```



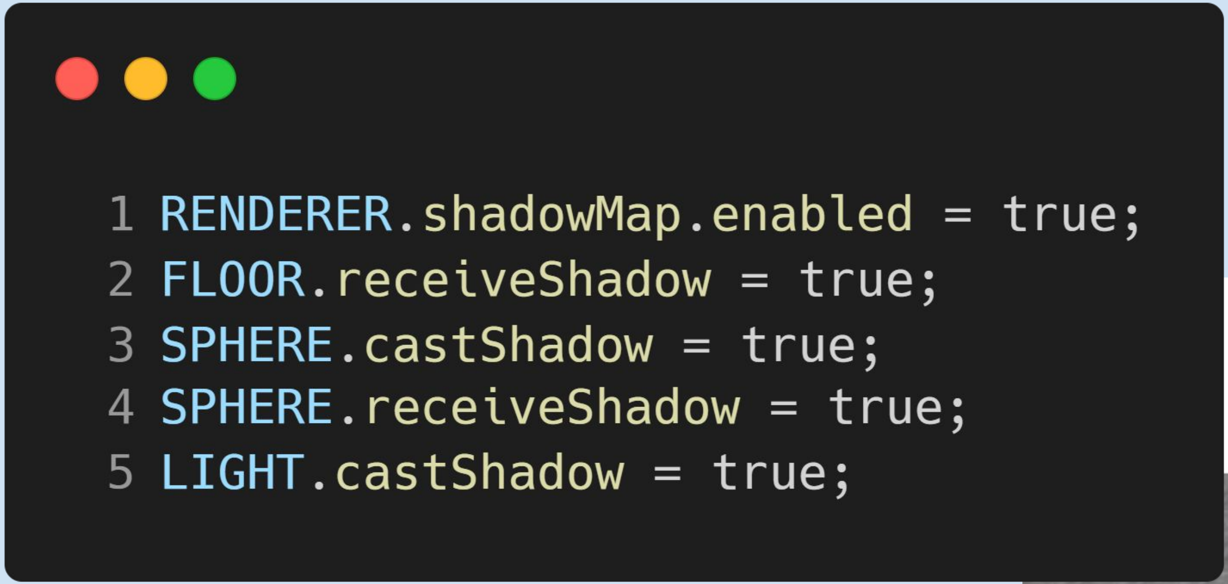
More things - Fog



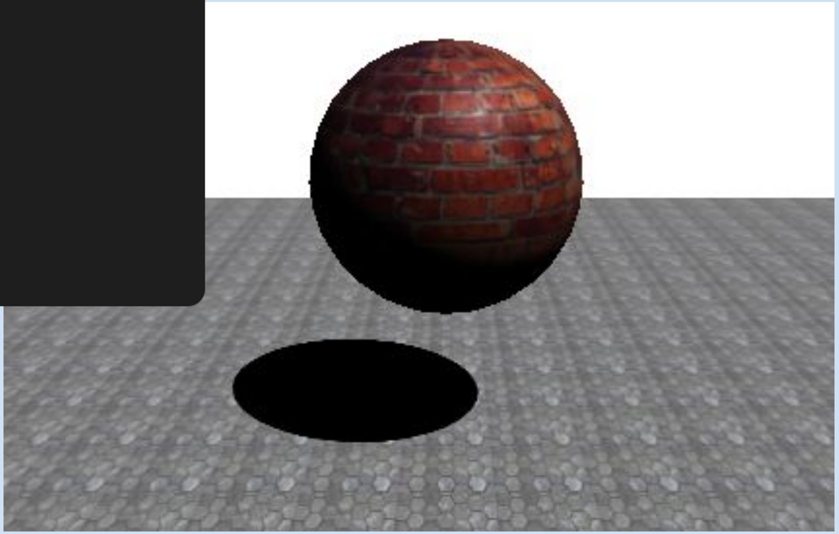
```
const SCENE = new THREE.Scene( );
{
  const NEAR = 1;
  const FAR = 2;
  const COLOR = 'purple';
  SCENE.fog = new THREE.Fog(COLOR, NEAR, FAR);
  SCENE.background = new THREE.Color(COLOR);
};
```

Fog with color and vision parameter

Shadows



```
RENDERER.shadowMap.enabled = true;
FLOOR.receiveShadow = true;
SPHERE.castShadow = true;
SPHERE.receiveShadow = true;
LIGHT.castShadow = true;
```



Bibliography

https://threejs.org/docs/index.html#api/en/renderers/WebGLRenderer

https://threejs.org/manual/

https://threejs.org/docs/index.html#manual/en/introduction/Creating-a-scene

https://github.com/josdirksen/threejs-cookbook

https://davidlyons.dev/threejs-intro/#slide-0

https://riptutorial.com/three-js