An Introduction to Three.js

Joaquim Madeira

February 2024

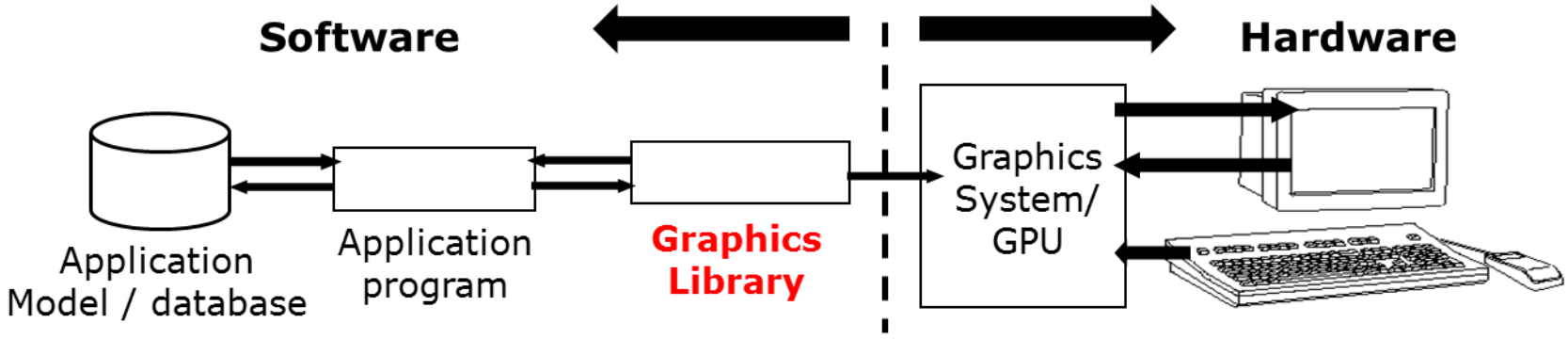
Topics

- Three.js – Main features

RECAP

Interactive Computer Graphics

- Graphics library / package is intermediary between application and display hardware
- Application program maps / renders objects / models to images by calling on the graphics library
- User interaction allows image and / or model modification



[van Dam]

CG Main Tasks

- Modeling
  - Construct individual models / objects
  - Assemble them into a 2D or 3D scene
- Animation
  - Static vs. dynamic scenes
  - Movement and / or deformation
- Rendering
  - Generate final images
  - Where is the viewer / camera ?
  - How is he / she looking at the scene?

Modeling vs Rendering

- Modeling
  - Create models
  - Apply materials to models
  - Place models around scene
  - Place lights in the scene
  - Place the camera

YouTube Demo

- Rendering
  - Take picture with the camera

[van Dam]

THREE.JS
– MAIN FEATURES

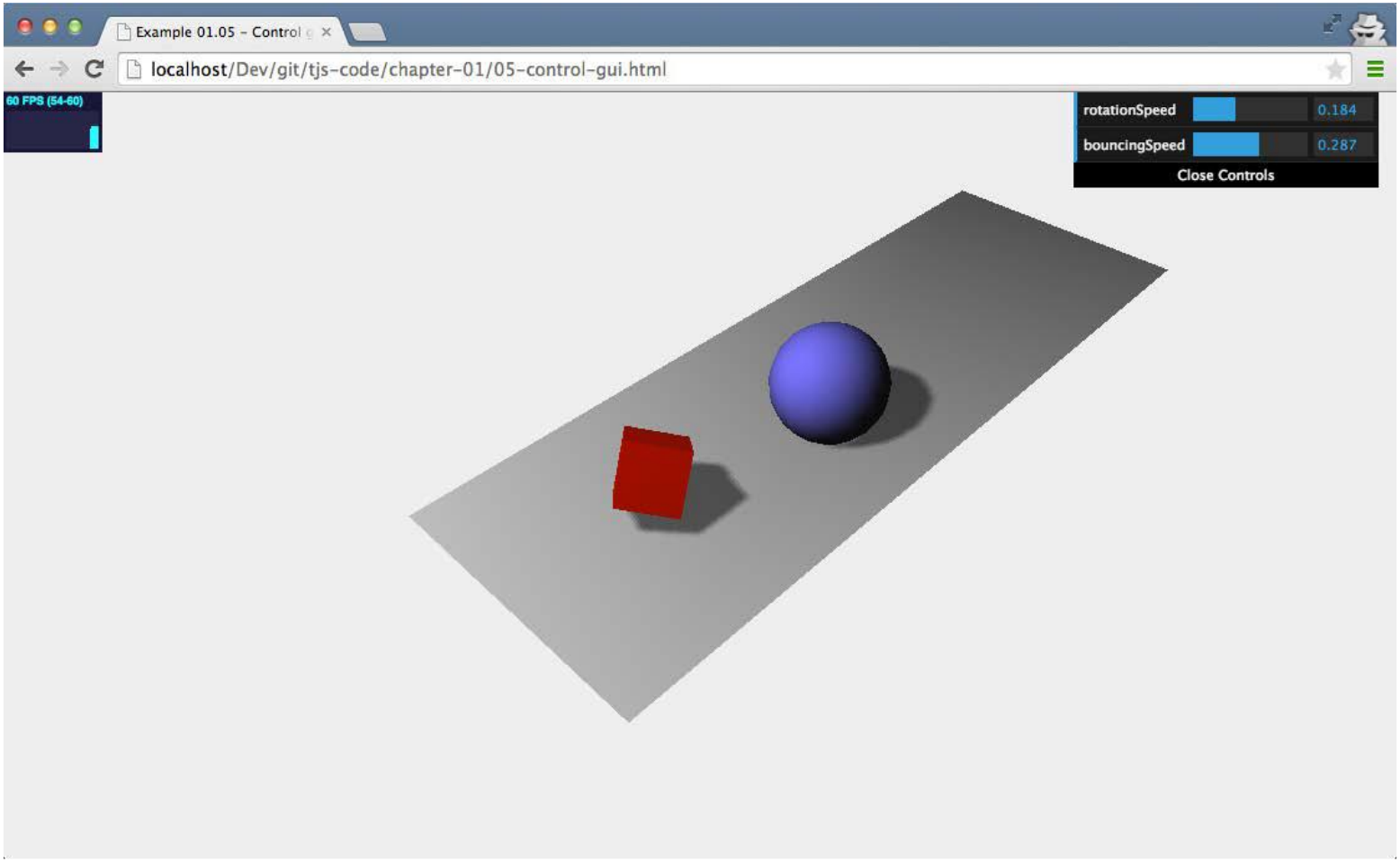
What can we do?

- With a few lines of JavaScript…
- Create anything, from simple 3D models to photorealistic real-time scenes

What can we do?

- Create simple and complex 3D geometries
- Animate and move objects through a 3D scene
- Apply textures and materials to objects
- Make use of different light sources to illuminate the scene

A simple scene

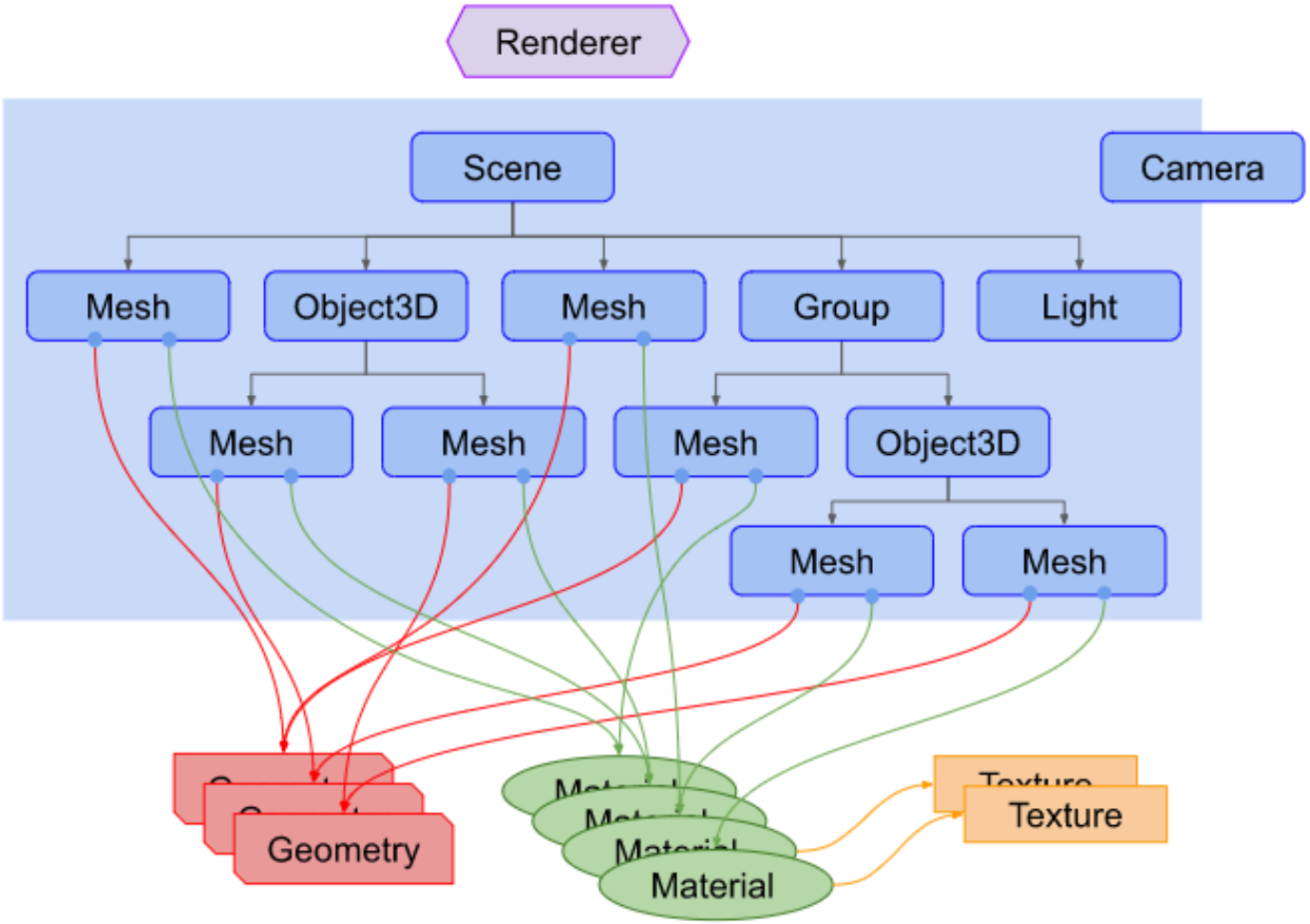


What can we do?

- Load objects from 3D-modeling software
- Add advanced postprocessing effects to a 3D scene
- Work with custom shaders
- Create point clouds

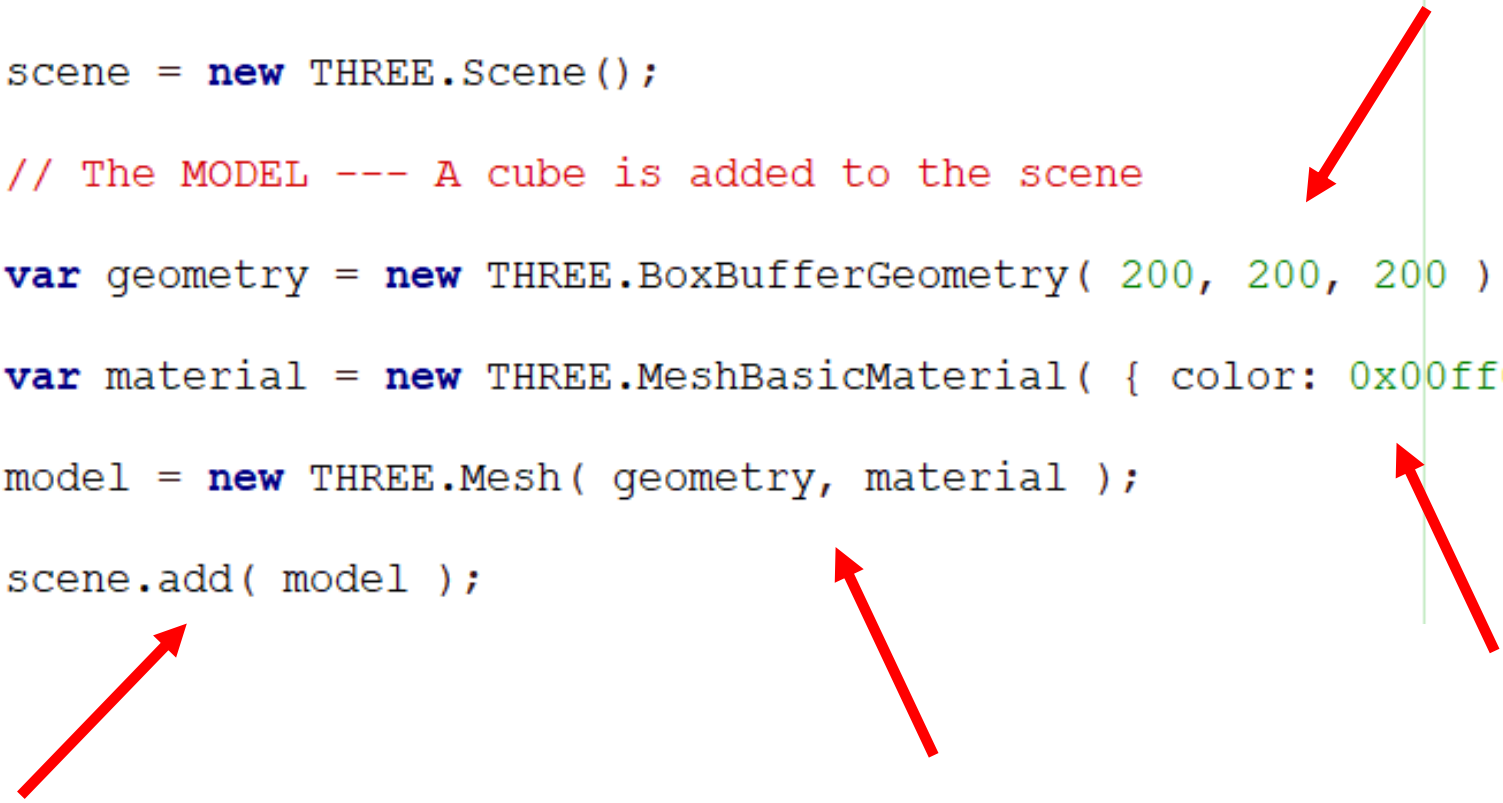
THE BASICS

Three.js – App structure



Three.js – A scene with a cube

```
// The SCENE

scene = new THREE.Scene();

// The MODEL --- A cube is added to the scene

var geometry = new THREE.BoxBufferGeometry( 200, 200, 200 );

var material = new THREE.MeshBasicMaterial( { color: 0x00ff00 } );

model = new THREE.Mesh( geometry, material );

scene.add( model );
```



Three.js – The camera

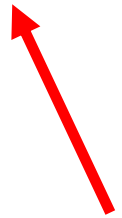
```
// The CAMERA

// --- Where the viewer is and how he is looking at the scene

camera = new THREE.PerspectiveCamera( 70,

                    window.innerWidth / window.innerHeight, 1, 1000 );

camera.position.z = 400;
```

Three.js – The renderer

```
// The RENDERER --- To dispplay the scene on the Web page

renderer = new THREE.WebGLRenderer( { antialias: true } );

renderer.setPixelRatio( window.devicePixelRatio );

renderer.setSize( window.innerWidth, window.innerHeight );

document.body.appendChild( renderer.domElement );
```

Three.js – Animation

```
function animate() {
    requestAnimationFrame( animate );
    model.rotation.x += 0.005;
    model.rotation.y += 0.01;
    renderer.render( scene, camera );
}
```

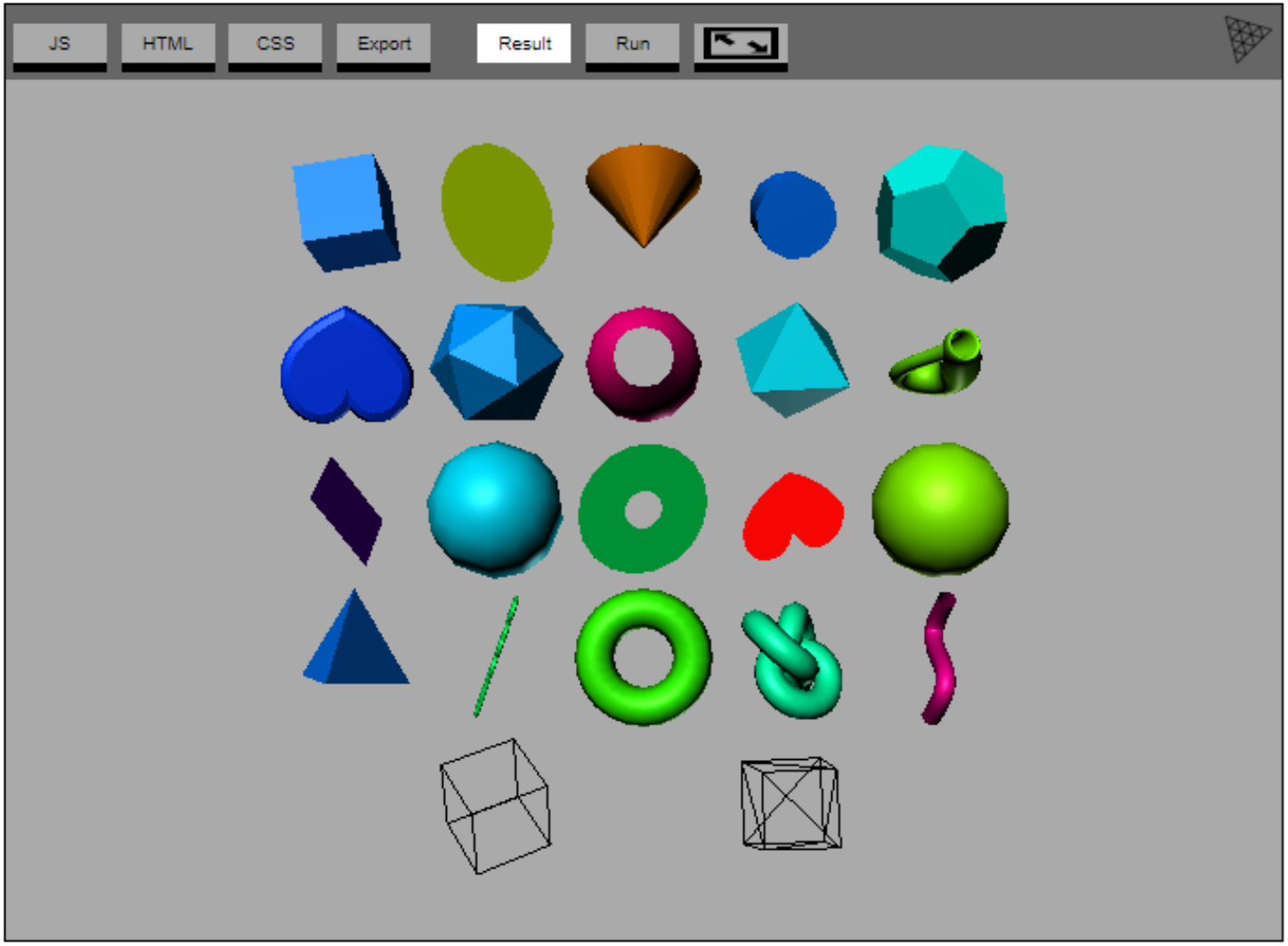


Three.js – Interaction

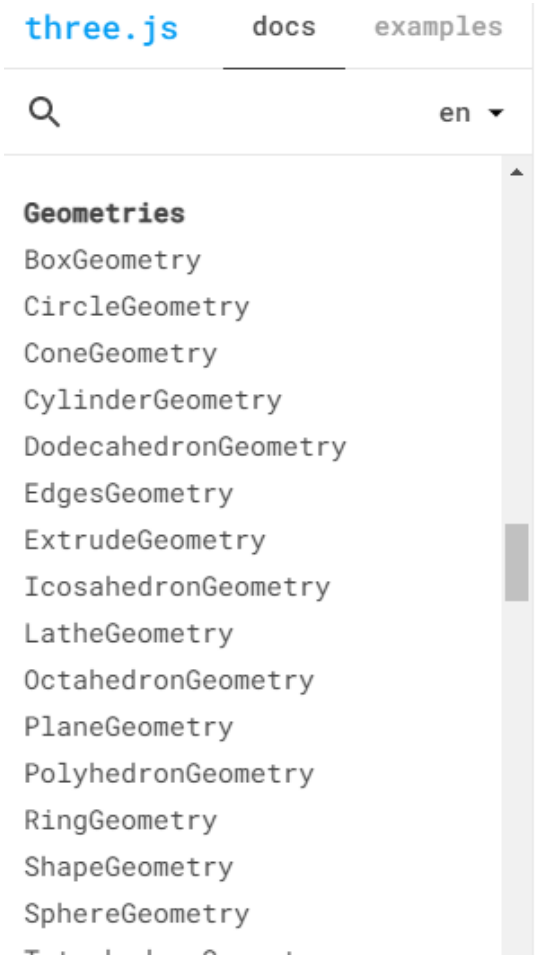
```
// What to do if resize occurs ?

window.addEventListener( 'resize', onWindowResize, false );


 function onWindowResize() {

     // Adjusting the renderer and camera features

     renderer.setSize( window.innerWidth, window.innerHeight );

     camera.aspect = window.innerWidth / window.innerHeight;

     camera.updateProjectionMatrix();
 }
```

3D GEOMETRIES

Three.js – 3D Geometries

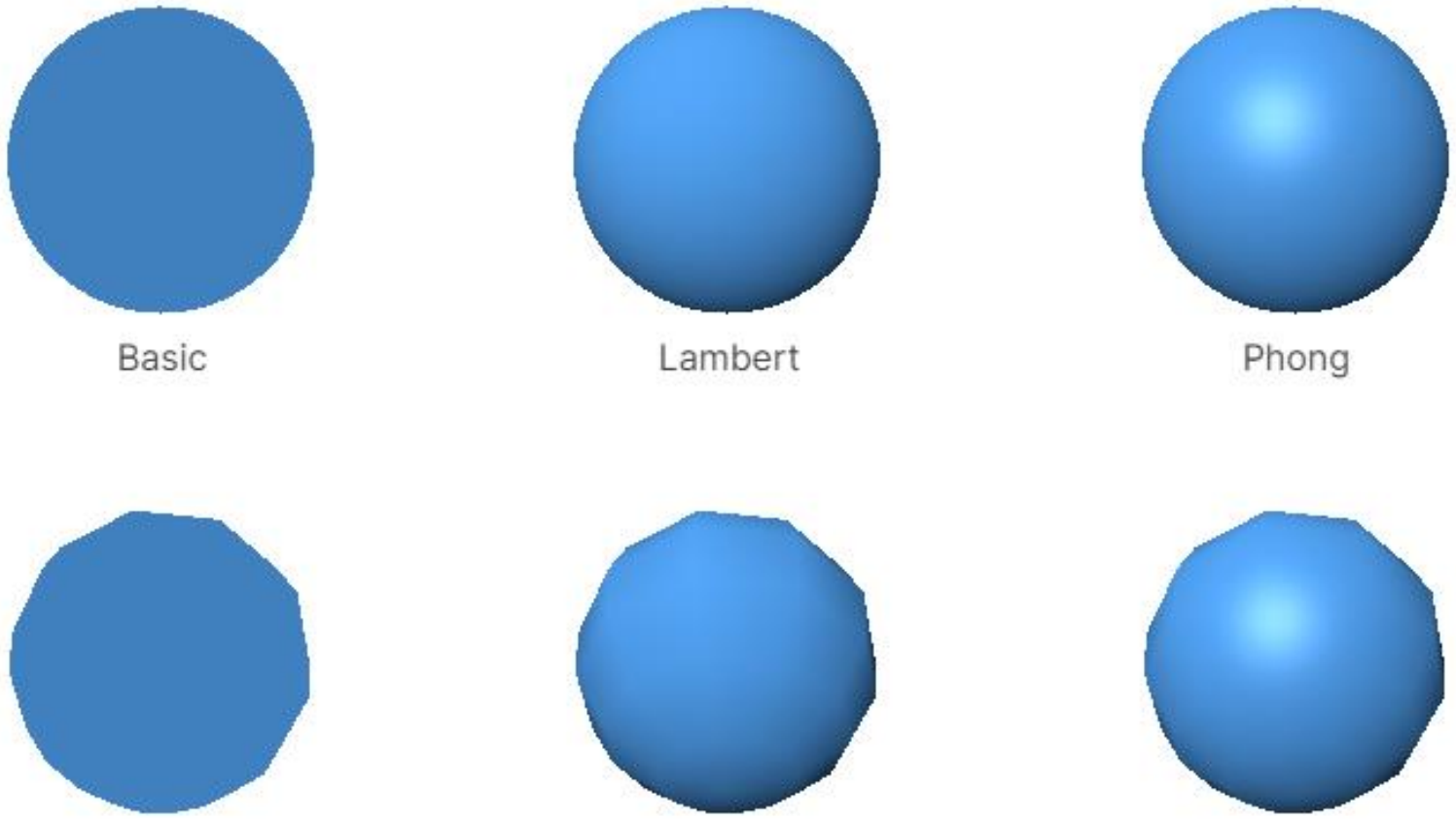


Three.js – 3D Geometries

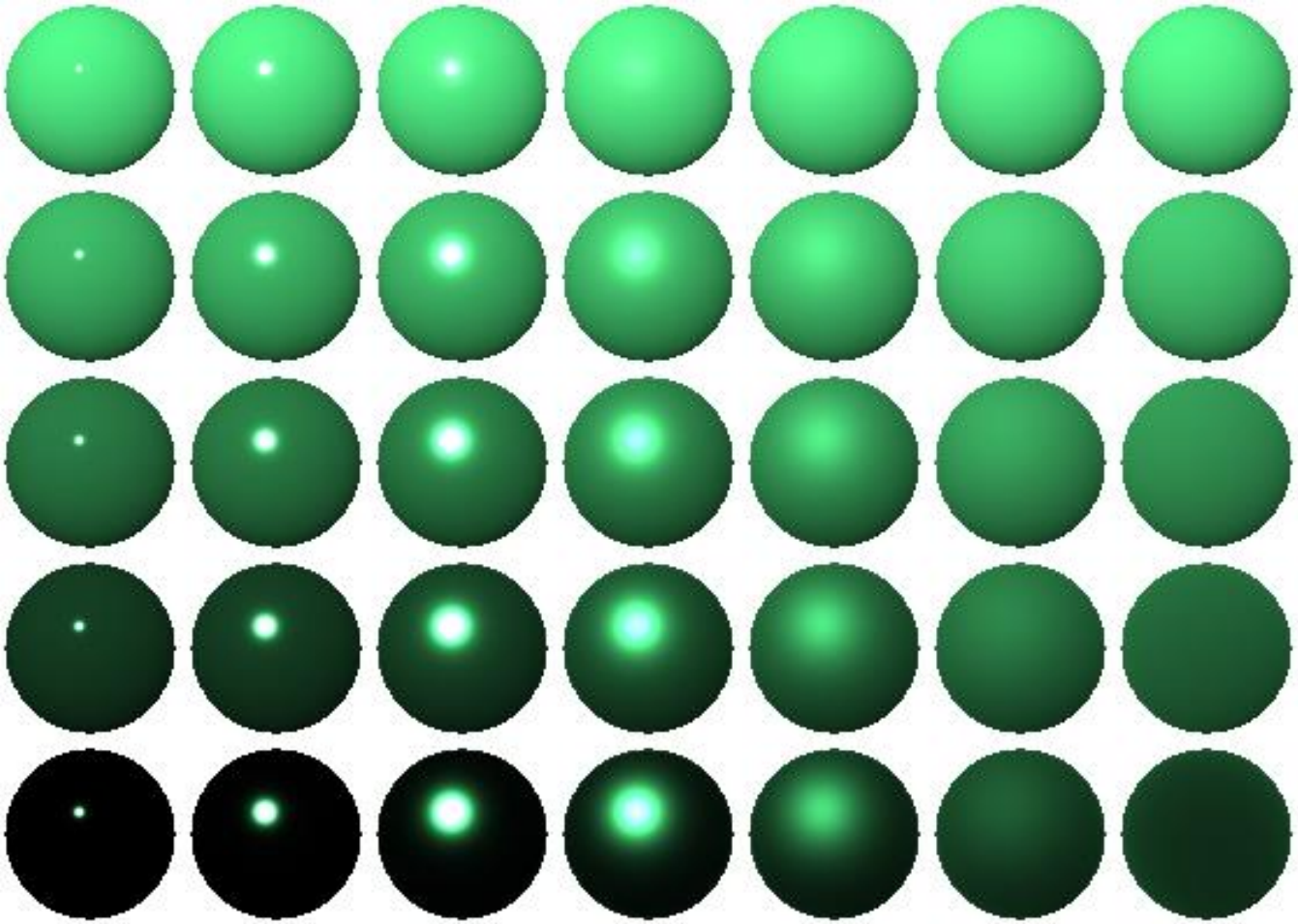


MATERIALS & ILLUMINATION

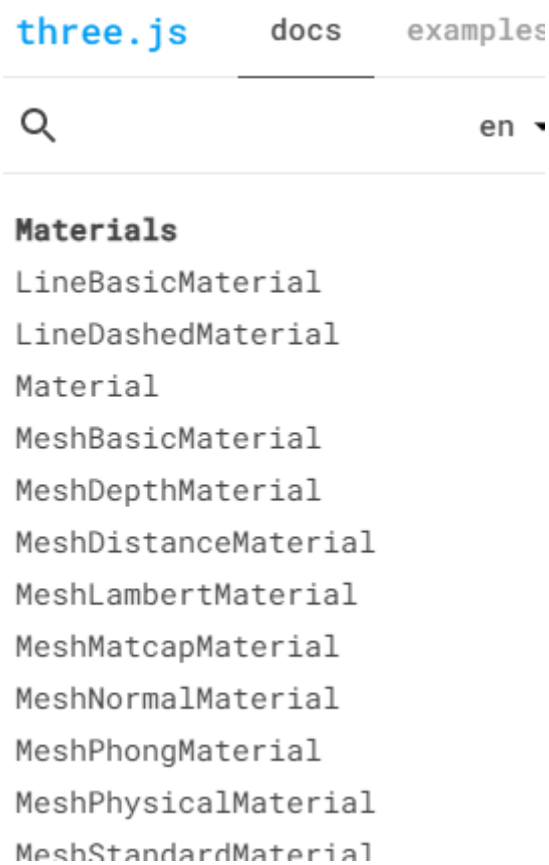
Three.js – Materials & Level-of-Detail



Three.js – Illumination effects

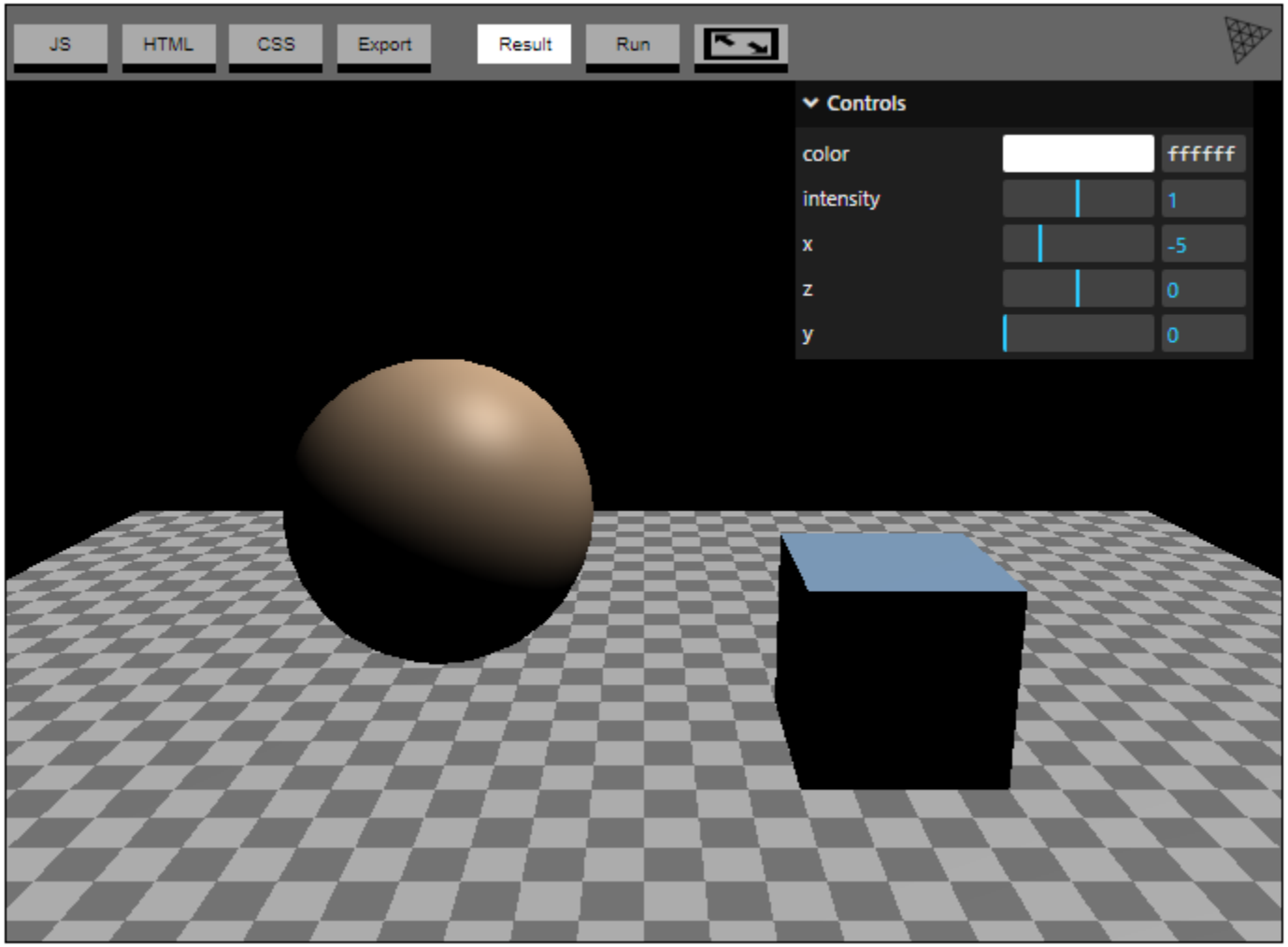


Three.js – Materials

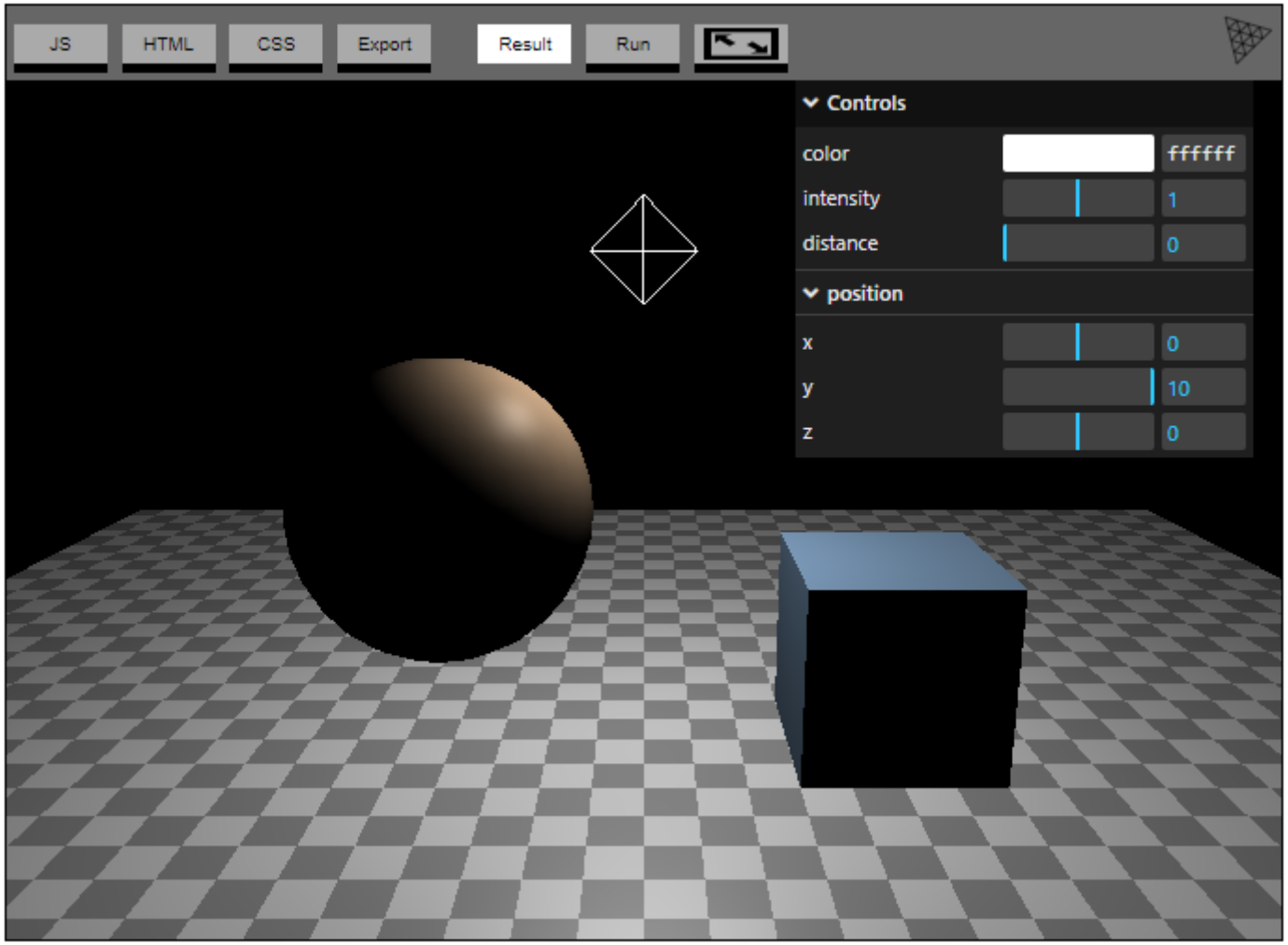


LIGHT SOURCES & SHADOWS

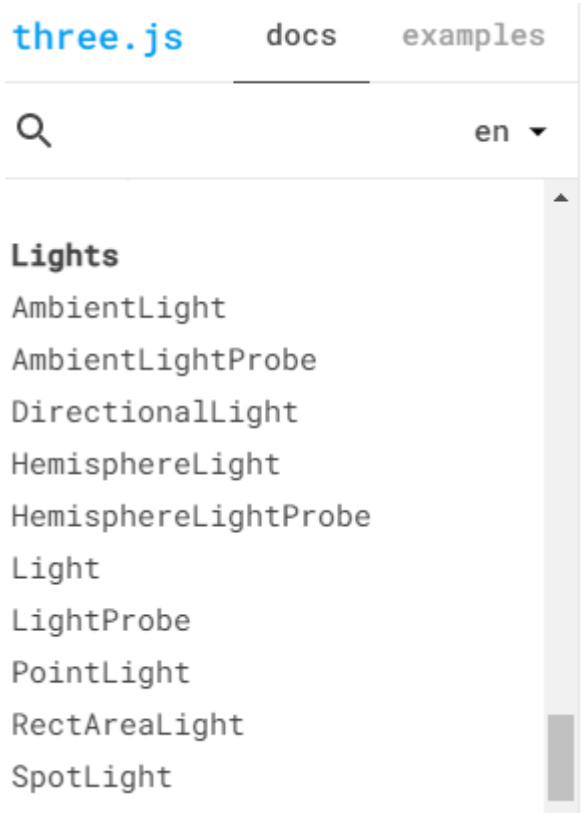
Three.js – Directional light source



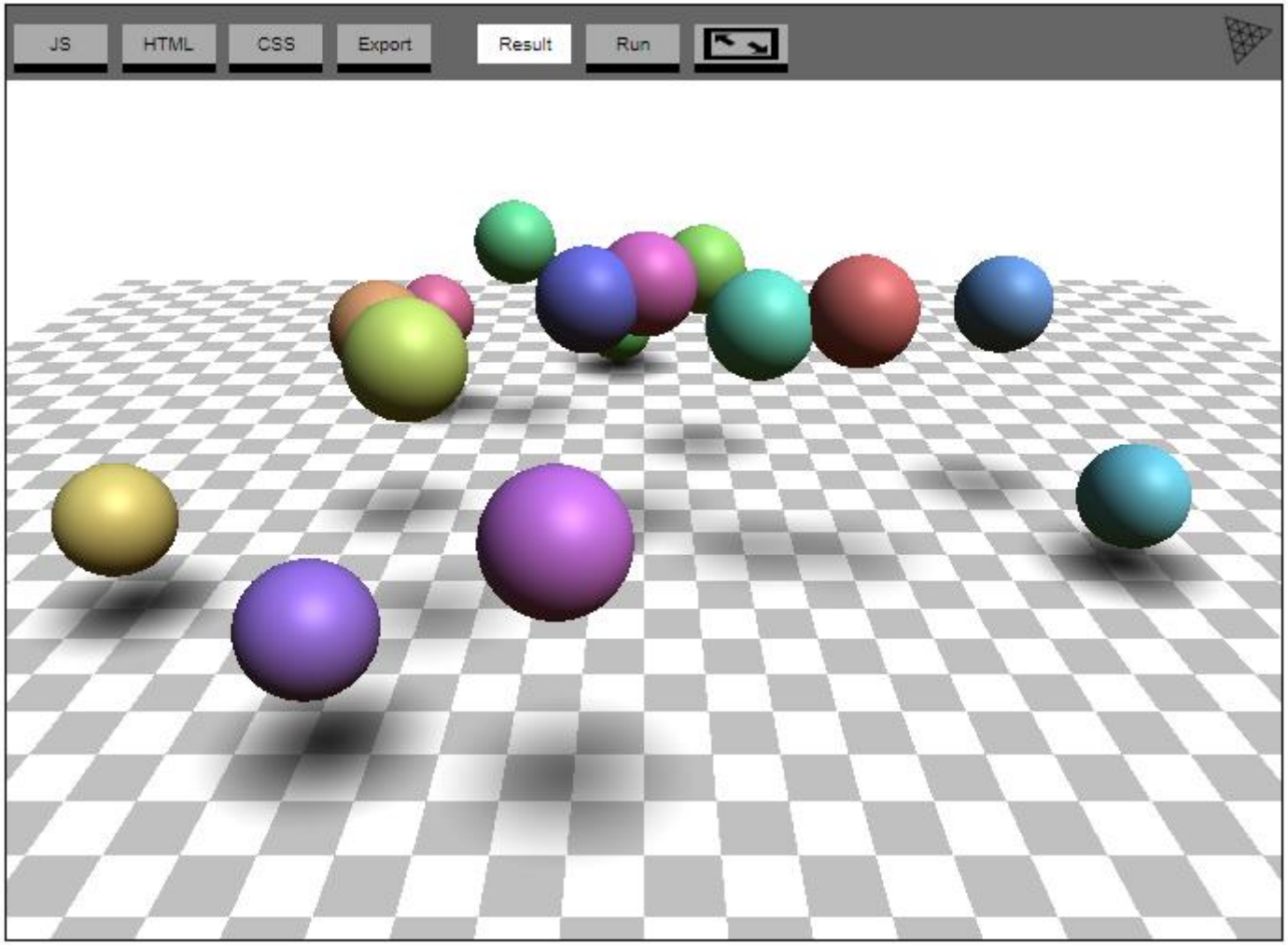
Three.js – Point light source



Three.js – Light sources



Three.js - Shadows



Acknowledgments

- These slides are based on the first chapter of J. Dirksen's book: Learning Three.js
- And on the Three.js manual
- Thanks!