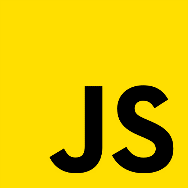
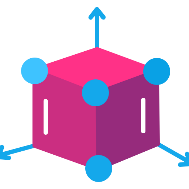
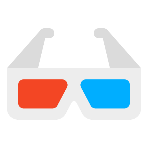
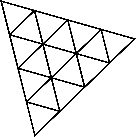
[](https://www.w3schools.com/js/)

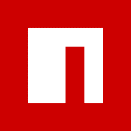
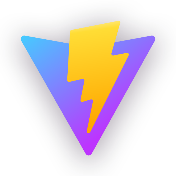


three.js



[](https://threejs.org/)

Three.js is a cross-browser [**JavaScript**](https://developer.mozilla.org/JavaScript) library and application programming interface used to create and display animated 3D computer graphics in a web browser using [**WebGL**](https://www.khronos.org/webgl/)

[](https://vitejs.dev/)

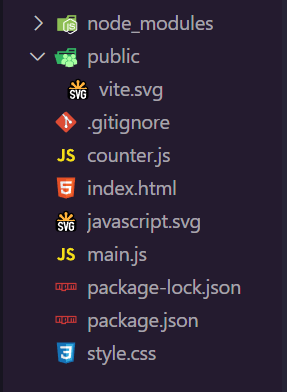
or

**npm init**

**npm i three**

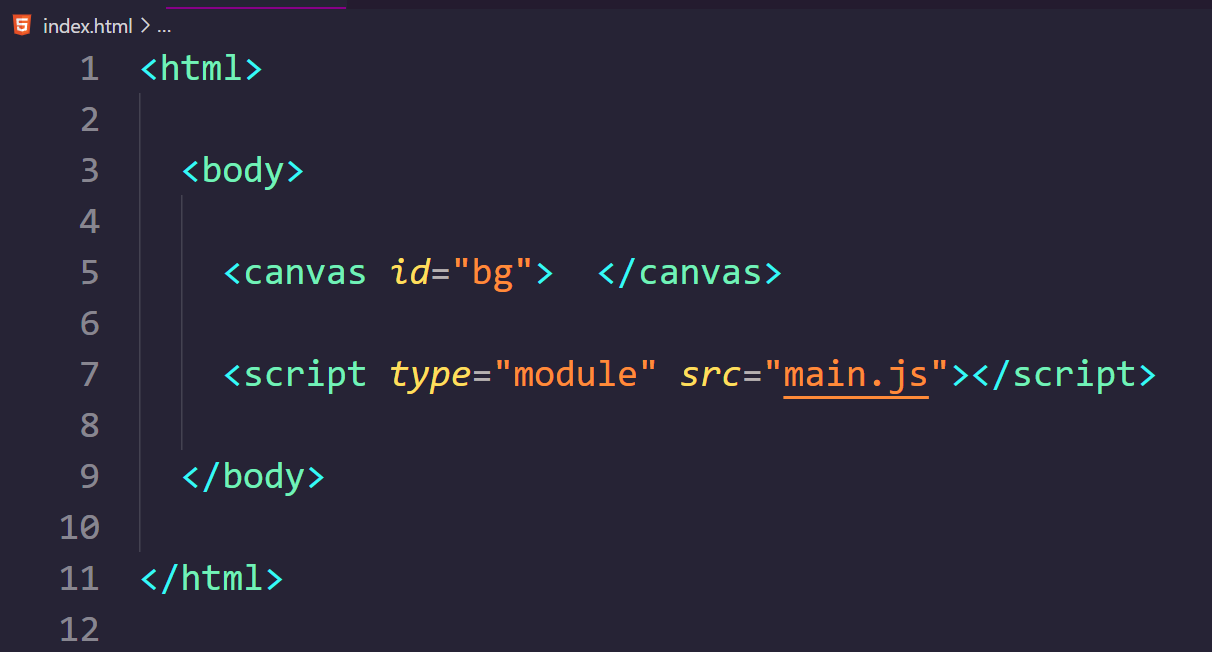
**npm init vite or npm create vite@latest**

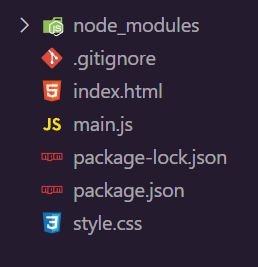
**npm i three**

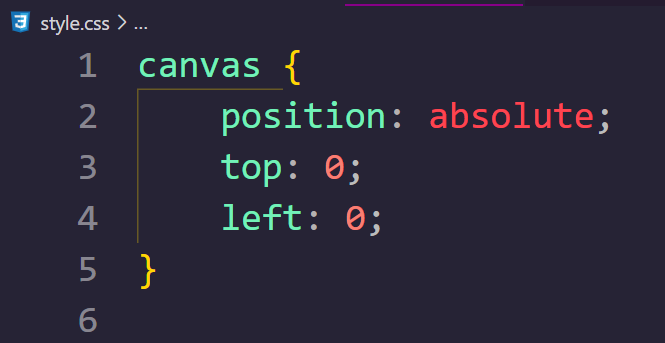


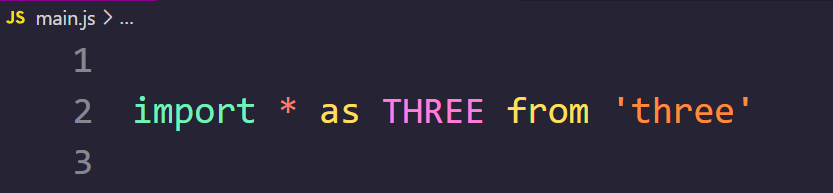
**npm run dev**

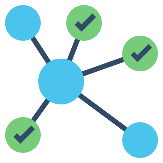
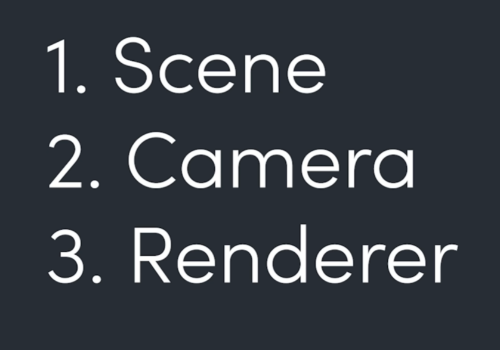
more information….









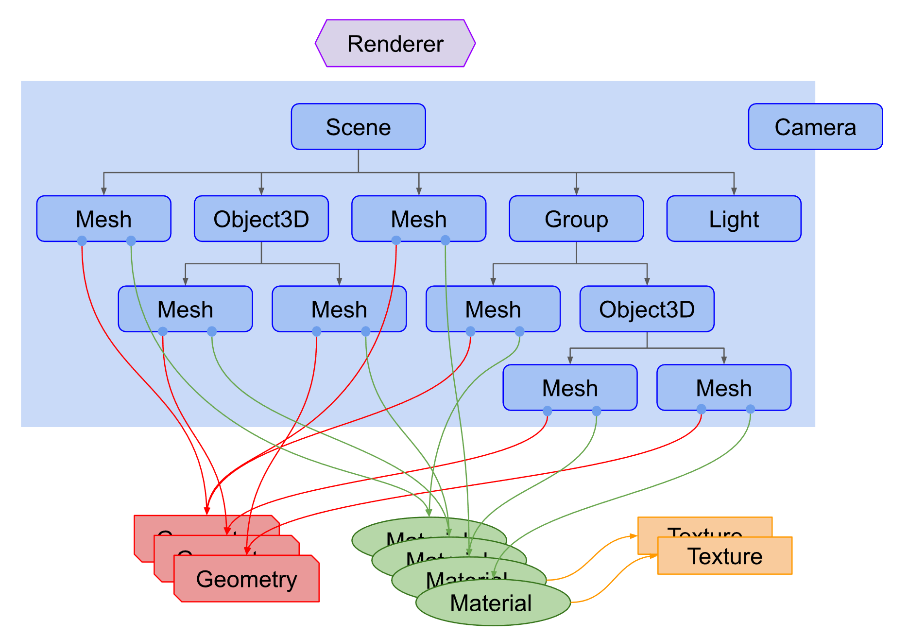
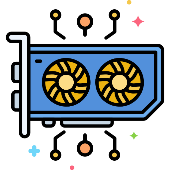


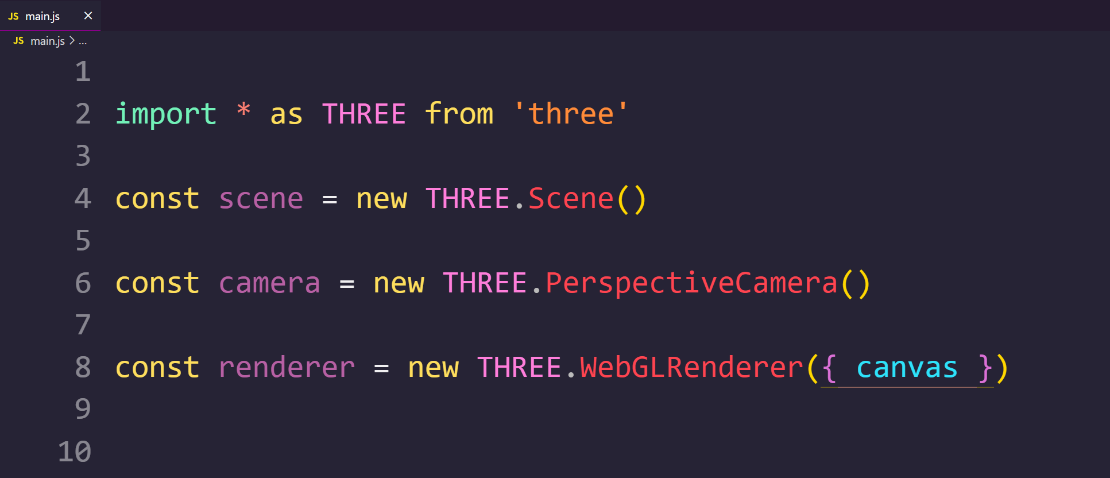
const scene = new THREE.Scene()

const camera = new THREE.PerspectiveCamera()

const renderer = new THREE.WebGLRenderer()

**3 Fundamental Objects**

[](https://threejs.org/manual/#en/fundamentals)



**Fundamental Steps**





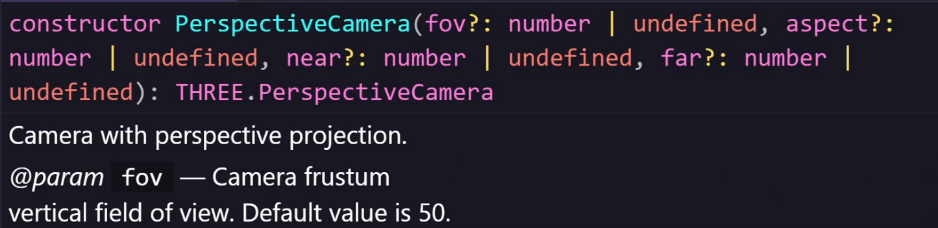
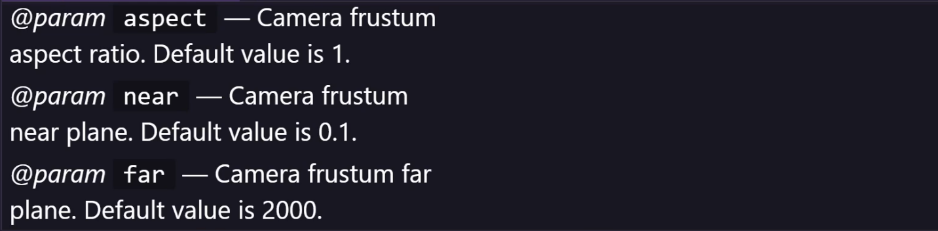
camera

const camera = new THREE.PerspectiveCamera( 50, window.innerWidth / window.innerHeight, 0.1, 2000 )



camera

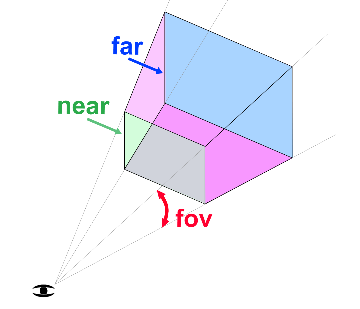
parameters



Field of view

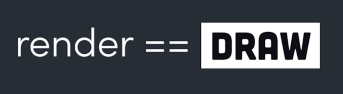
Aspect ratio

Viewing distance



renderer

const renderer = new THREE.WebGLRenderer({ canvas })



or

Need to draw on something

if the **canvas** has **id** , then

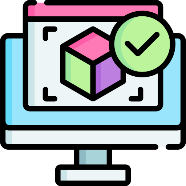
<canvas *id*="bg"> </canvas>



const renderer = new THREE.WebGLRenderer({

    canvas: document.querySelector('#bg')

})



Set **size** of rendered **output**

**output** is the image of the scene that captured by the camera

After creating **renderer**

**Rendering the output in browser**

const renderer = new THREE.WebGLRenderer({ canvas })

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

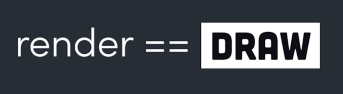
renderer.setPixelRatio( window.devicePixelRatio ) ( 5 )

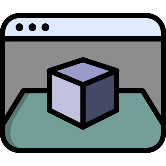
renderer.render ( scene, camera )

or

Set the **resolution** of the **output**

render method of renderer will draw the output in browser canvas





**Calling the render method at the end by using Recursive method is the ideal way**

*// Recursive Game Loop*

create new **3D object**

 scene.add( **3D object** )

const loop = () => **{**

  something.update**()**

  renderer.render**(** scene, camera **)**

  requestAnimationFrame**(** loop **)**

**}**

loop**()**



*// Boring Non Recursive*

create new **3D object**

 scene.add( **3D object** )

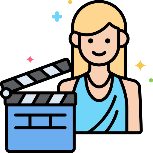
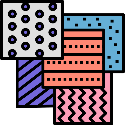
 renderer.render**(** scene, camera **)**

or



**Pseudo Code !**

more about animation loop…



more information….

const geometry = new THREE**.**SphereGeometry**(** 2, 10, 10 **)**

const material = new THREE**.**MeshStandardMaterial**(**{ color: '#00ff83' }**)**

const mesh = new THREE**.**Mesh**(** geometry, material **)**

scene**.**add**(** mesh **)**

**Add a mesh to the scene**

**Change the mesh color**

THREE.PointLight **(**

color: hexadecimal,

intensity: number,

distance: number,

decay: number**)**

**Add a light source to the scene**

**Set the camera to a better position**

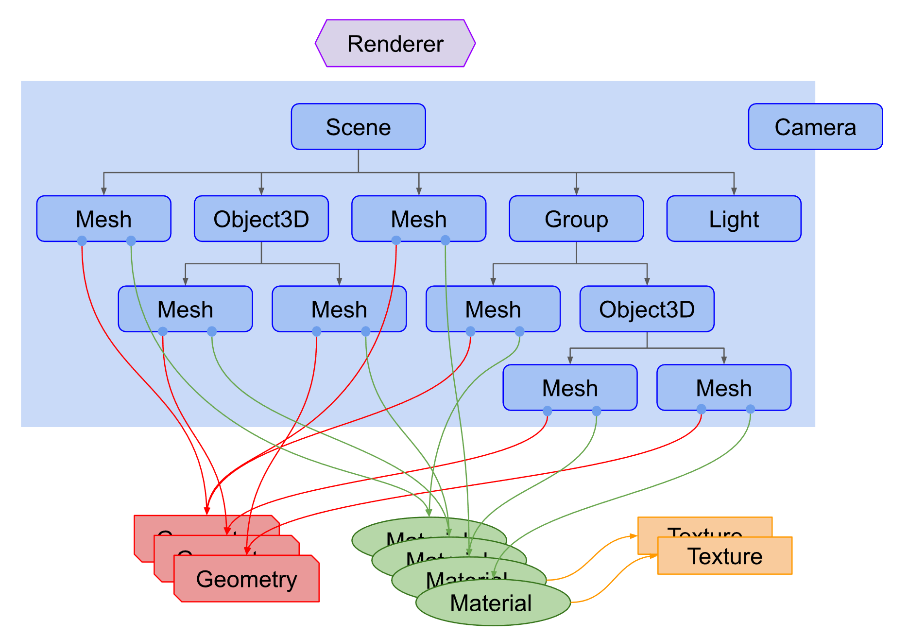
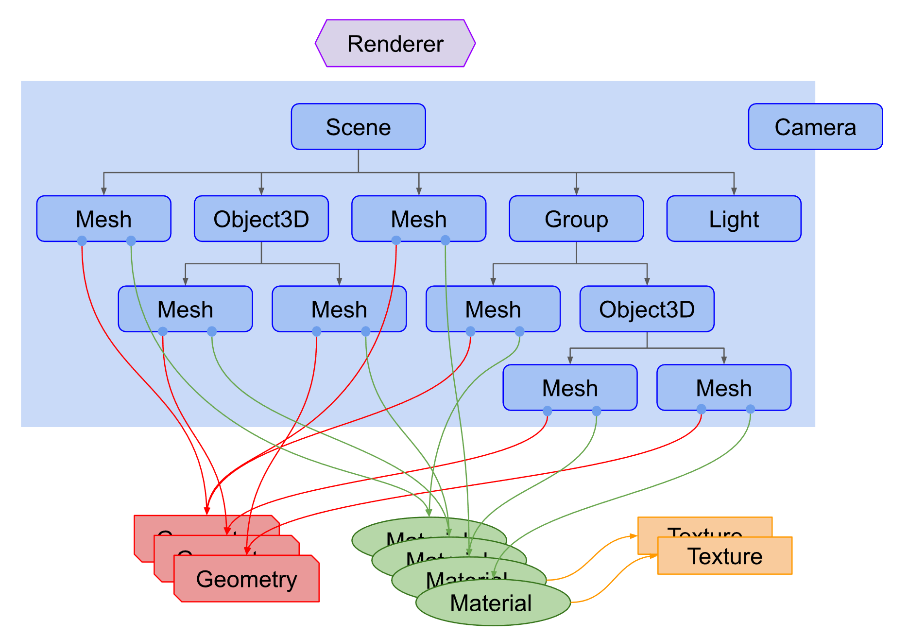
or

camera.position.z = 20

camera.position.setZ ( 20 )

after creating an object, always add it to the scene

 scene.add ( **object-name** )



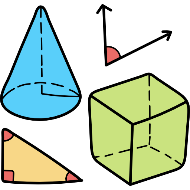
**Major steps are done, basic setup completed, now it’s time to add** **Lights** **&** **3D** **objects to the** **scene**

**Lights Camera Action**

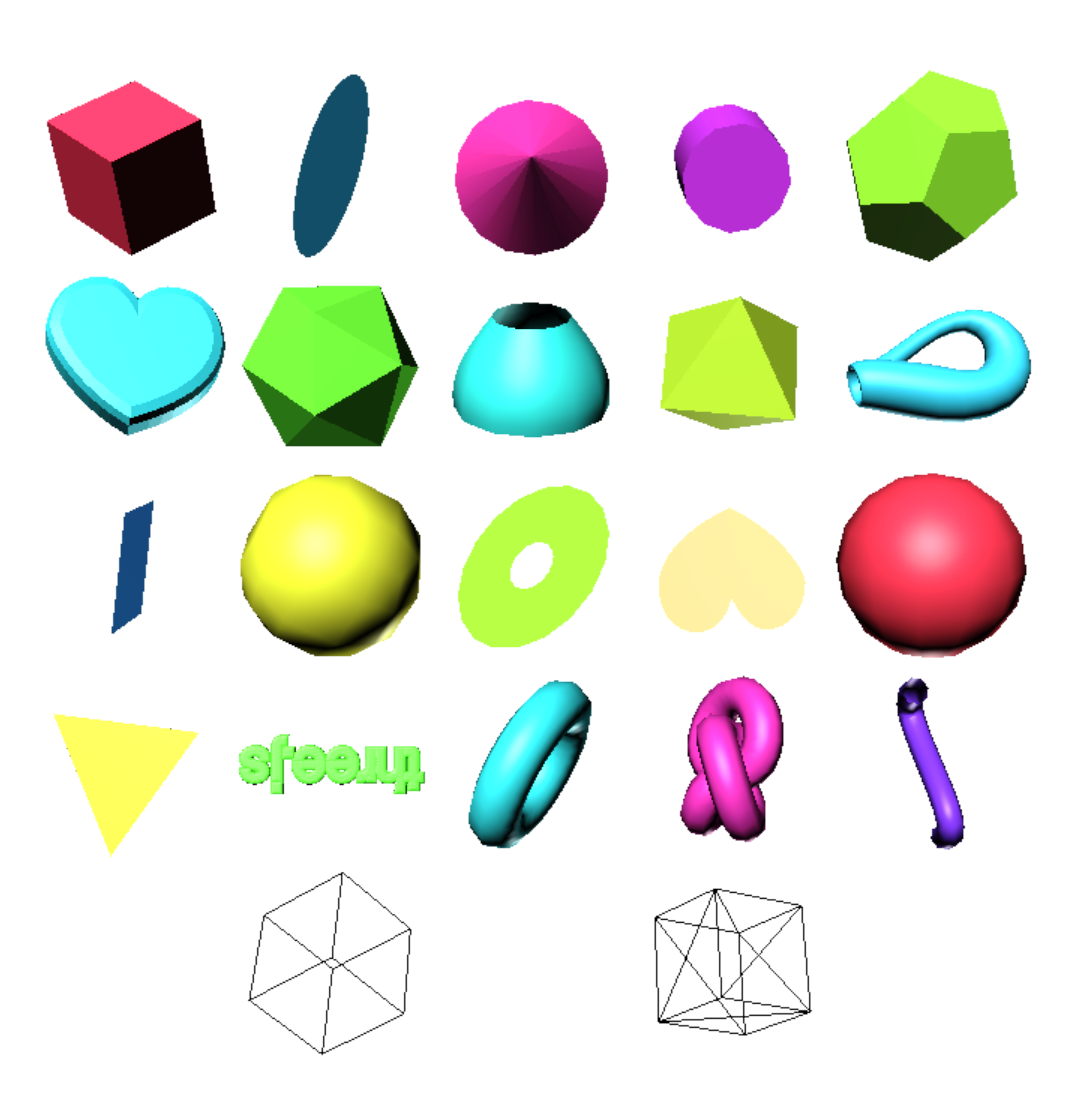
const light = new THREE**.**PointLight**(** 0xffffff **)**

light.position**.**set( 10,10,10 )

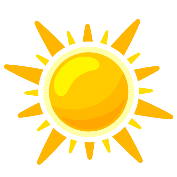
scene**.**add**(** light **)**



**Primitive Geometries**

[](https://threejs.org/manual/#en/primitives)

* **Geometries**
* [BoxGeometry](https://threejs.org/docs/api/en/geometries/BoxGeometry.html)
* [CapsuleGeometry](https://threejs.org/docs/api/en/geometries/CapsuleGeometry.html)
* [CircleGeometry](https://threejs.org/docs/api/en/geometries/CircleGeometry.html)
* [ConeGeometry](https://threejs.org/docs/api/en/geometries/ConeGeometry.html)
* [CylinderGeometry](https://threejs.org/docs/api/en/geometries/CylinderGeometry.html)
* [DodecahedronGeometry](https://threejs.org/docs/api/en/geometries/DodecahedronGeometry.html)
* [EdgesGeometry](https://threejs.org/docs/api/en/geometries/EdgesGeometry.html)
* [ExtrudeGeometry](https://threejs.org/docs/api/en/geometries/ExtrudeGeometry.html)
* [IcosahedronGeometry](https://threejs.org/docs/api/en/geometries/IcosahedronGeometry.html)
* [LatheGeometry](https://threejs.org/docs/api/en/geometries/LatheGeometry.html)
* [OctahedronGeometry](https://threejs.org/docs/api/en/geometries/OctahedronGeometry.html)
* [PlaneGeometry](https://threejs.org/docs/api/en/geometries/PlaneGeometry.html)
* [PolyhedronGeometry](https://threejs.org/docs/api/en/geometries/PolyhedronGeometry.html)
* [RingGeometry](https://threejs.org/docs/api/en/geometries/RingGeometry.html)
* [ShapeGeometry](https://threejs.org/docs/api/en/geometries/ShapeGeometry.html)
* [SphereGeometry](https://threejs.org/docs/api/en/geometries/SphereGeometry.html)
* [TetrahedronGeometry](https://threejs.org/docs/api/en/geometries/TetrahedronGeometry.html)
* [TorusGeometry](https://threejs.org/docs/api/en/geometries/TorusGeometry.html)
* [TorusKnotGeometry](https://threejs.org/docs/api/en/geometries/TorusKnotGeometry.html)
* [TubeGeometry](https://threejs.org/docs/api/en/geometries/TubeGeometry.html)
* [WireframeGeometry](https://threejs.org/docs/api/en/geometries/WireframeGeometry.html)



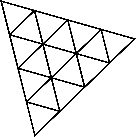
**Lights**

* **Lights**
* [AmbientLight](https://threejs.org/docs/api/en/lights/AmbientLight.html)
* [AmbientLightProbe](https://threejs.org/docs/api/en/lights/AmbientLightProbe.html)
* [DirectionalLight](https://threejs.org/docs/api/en/lights/DirectionalLight.html)
* [HemisphereLight](https://threejs.org/docs/api/en/lights/HemisphereLight.html)
* [HemisphereLightProbe](https://threejs.org/docs/api/en/lights/HemisphereLightProbe.html)
* [Light](https://threejs.org/docs/api/en/lights/Light.html)
* [LightProbe](https://threejs.org/docs/api/en/lights/LightProbe.html)
* [PointLight](https://threejs.org/docs/api/en/lights/PointLight.html)
* [RectAreaLight](https://threejs.org/docs/api/en/lights/RectAreaLight.html)
* [SpotLight](https://threejs.org/docs/api/en/lights/SpotLight.html)



**Cameras**

* **Cameras**
* [ArrayCamera](https://threejs.org/docs/api/en/cameras/ArrayCamera.html)
* [Camera](https://threejs.org/docs/api/en/cameras/Camera.html)
* [CubeCamera](https://threejs.org/docs/api/en/cameras/CubeCamera.html)
* [OrthographicCamera](https://threejs.org/docs/api/en/cameras/OrthographicCamera.html)
* [PerspectiveCamera](https://threejs.org/docs/api/en/cameras/PerspectiveCamera.html)
* [StereoCamera](https://threejs.org/docs/api/en/cameras/StereoCamera.html)

[](https://threejs.org/)