

three.js

tutorial version 1.0

three.js

JavaScript 3D Library

The screenshot shows the official [three.js](https://threejs.org) website. The page features a sidebar on the left with various navigation links, and a main content area on the right displaying a grid of 3D projects.

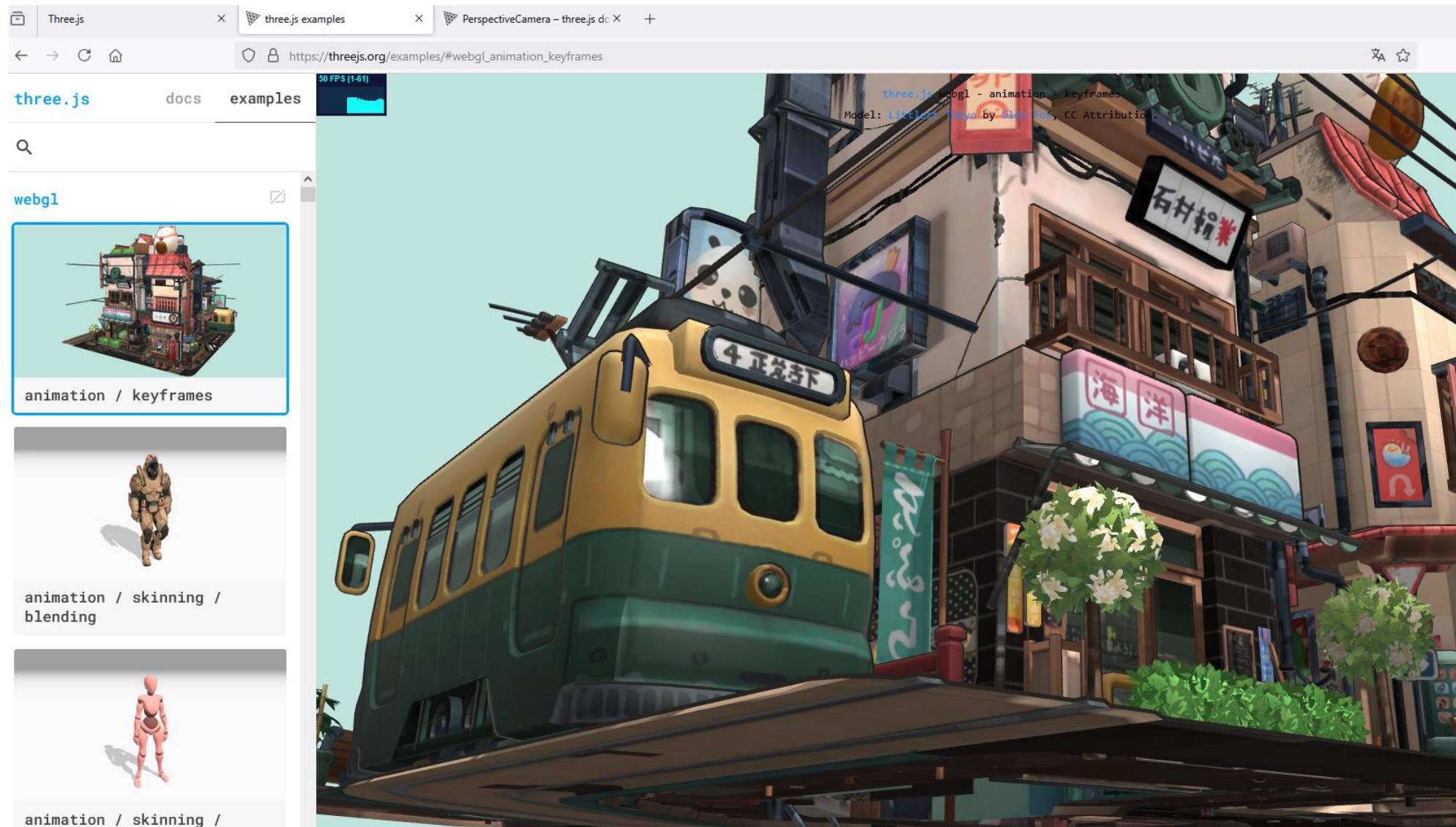
Navigation Links (Sidebar):

- Learn**
 - documentation (highlighted with a red box)
 - examples (highlighted with a red box)
 - editor
 - nodes
 - gpt
- Community**
 - questions
 - discord
 - forum
 - twitter
- Code**
 - github
 - download
- Resources**
 - Three.js Fundamentals (highlighted with a red box)
 - Three.js Journey
 - Learn Three.js
 - 初めてのThree.js
- Merch**
 - T-Shirts

Main Content Area: The right side of the page displays a 5x6 grid of 30 small images, each representing a different 3D project or application built with three.js. Some of the visible titles include "COASTAL WORLD", "UNIQUE JEWELRY", "THEATRE JS v0.5", "The Museum of Annoying Experiences", "nodeltoy", "GEMINI", "Discobrain", "MEDAL OF HONOR ABOVE AND BEYOND", and "ROGUE ENGINE".

<https://threejs.org/>

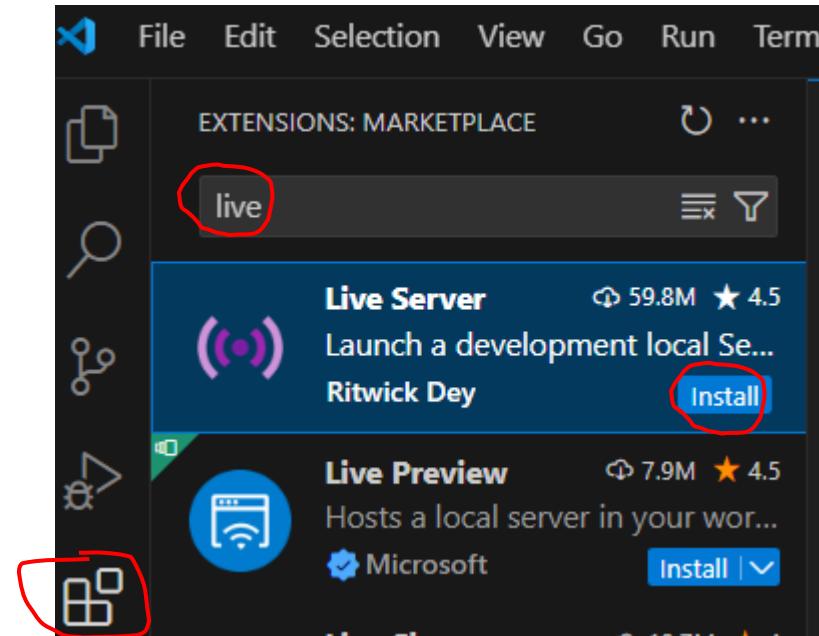
three.js examples



https://threejs.org/examples/#webgl_animation_keyframes

Visual Studio Code Live Server

instalacja Live Server w VSC



Cube

The screenshot shows the VS Code interface with the following details:

- File Explorer (Left):** Shows a folder named "CUBE" containing "index.html" and "main.js". The "index.html" file is highlighted with a red oval.
- Code Editor (Right):** Displays the content of "index.html".

```
<!DOCTYPE html>
<html>
<head>
    <meta charset='utf-8'>
    <title>Three.js</title>
    <script type="importmap">
        {
            "imports": {
                "three": "https://cdn.jsdelivr.net/npm/three@0.173.0/build/three.module.js",
                "three/addons/": "https://cdn.jsdelivr.net/npm/three@0.173.0/examples/jsm/"
            }
        }
    </script>
</head>
<body>
    <script type="module" src="/main.js"></script>
</body>
</html>
```
- Annotations:**
 - A red oval highlights the "CUBE" folder in the File Explorer.
 - An orange arrow points from the text "index.html" at the bottom left to the "index.html" file in the code editor.
 - A red line highlights the version numbers "0.173.0" in the import URLs.
 - An orange line highlights the URL "https://threejs.org/docs/index.html#manual/en/introduction/Installation" at the bottom right.

projekt Cube składa się z dwóch plików: index.html oraz main.js

wersja biblioteki (najlepiej najnowsza)

index.html

Cube

File Edit Selection View Go Run Terminal Help ← → 🔍 cube

main.js x

main.js > ...

```
1 import * as THREE from 'three';
2
3 const scene = new THREE.Scene();
4 const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );
5
6 const renderer = new THREE.WebGLRenderer();
7 renderer.setSize( window.innerWidth, window.innerHeight );
8 renderer.setAnimationLoop( animate );
9 document.body.appendChild( renderer.domElement );
10
11 const geometry = new THREE.BoxGeometry( 1, 1, 1 );
12 const material = new THREE.MeshBasicMaterial( { color: 0xff0000 } );
13 const cube = new THREE.Mesh( geometry, material );
14 scene.add( cube );
15
16 camera.position.z = 2;
17
18 function animate() {
19
20     cube.rotation.x += 0.01;
21     cube.rotation.y += 0.01;
22
23     renderer.render( scene, camera );
24 }
25 }
```

fov aspect ratio near far

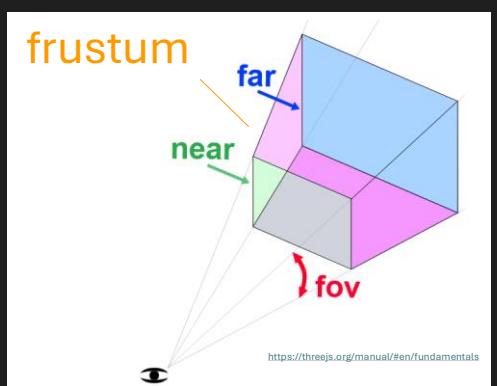
frustum

far

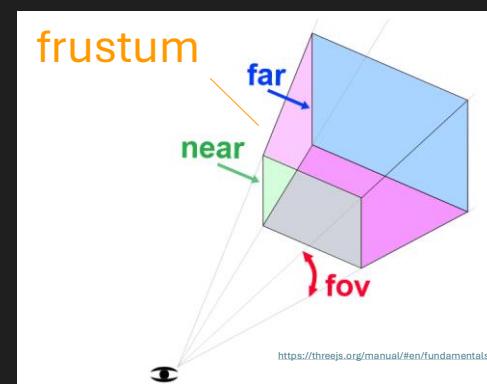
near

fov

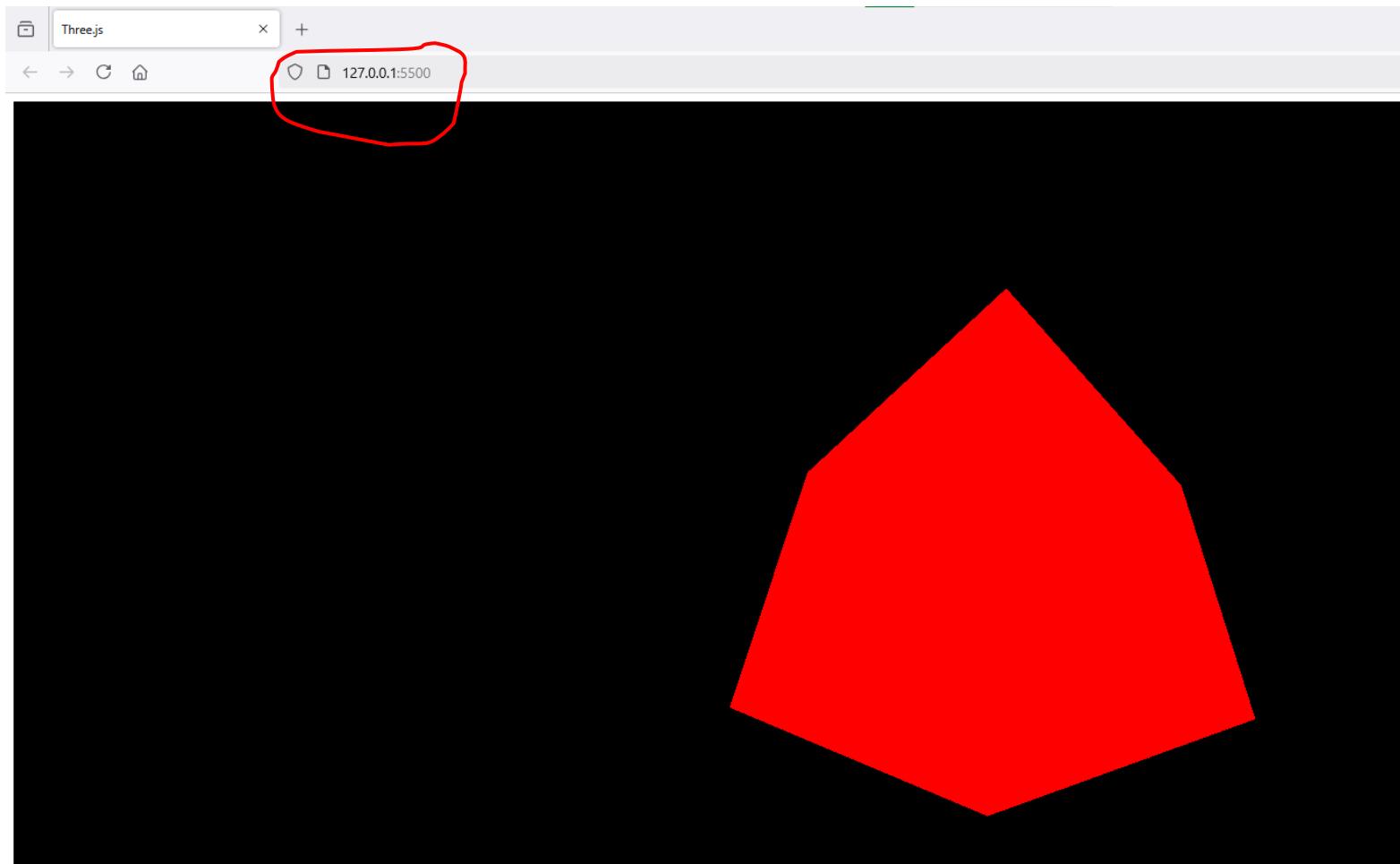
https://threejs.org/manual/#en/fundamentals



main.js



Cube



Live Server



Cube + Capsule

three.js docs examples

en ▾

LineCurve
LineCurve3
QuadraticBezierCurve
QuadraticBezierCurve3
SplineCurve

Geometries

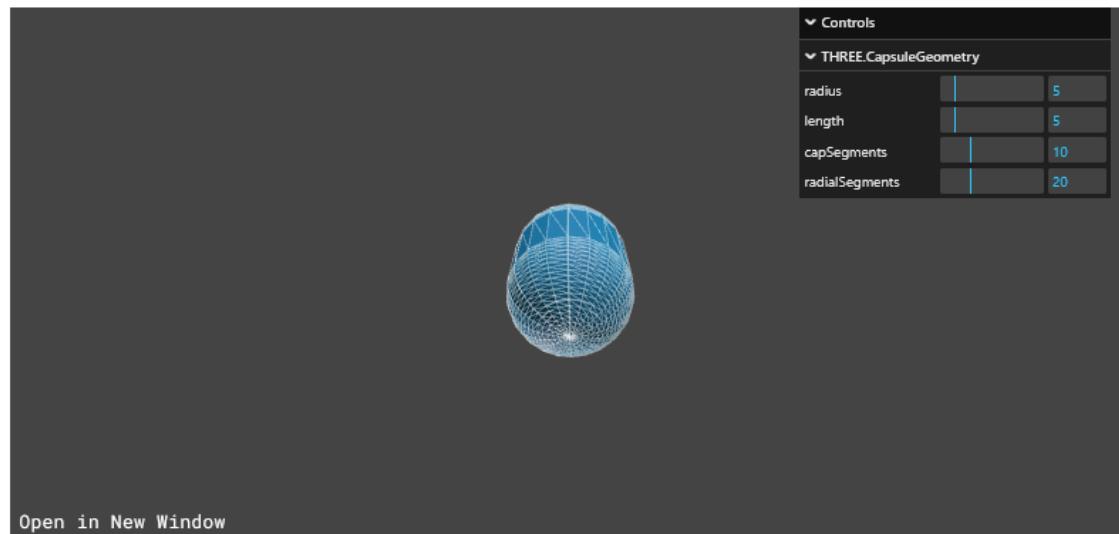
BoxGeometry
CapsuleGeometry

CircleGeometry
ConeGeometry
CylinderGeometry
DodecahedronGeometry
EdgesGeometry
ExtrudeGeometry
IcosahedronGeometry
LatheGeometry
OctahedronGeometry
PlaneGeometry
PolyhedronGeometry
RingGeometry
ShapeGeometry
SphereGeometry
TetrahedronGeometry
TorusGeometry
TorusKnotGeometry

BufferGeometry → LatheGeometry →

CapsuleGeometry

CapsuleGeometry is a geometry class for a capsule with given radii and height. It is constructed using a lathe.



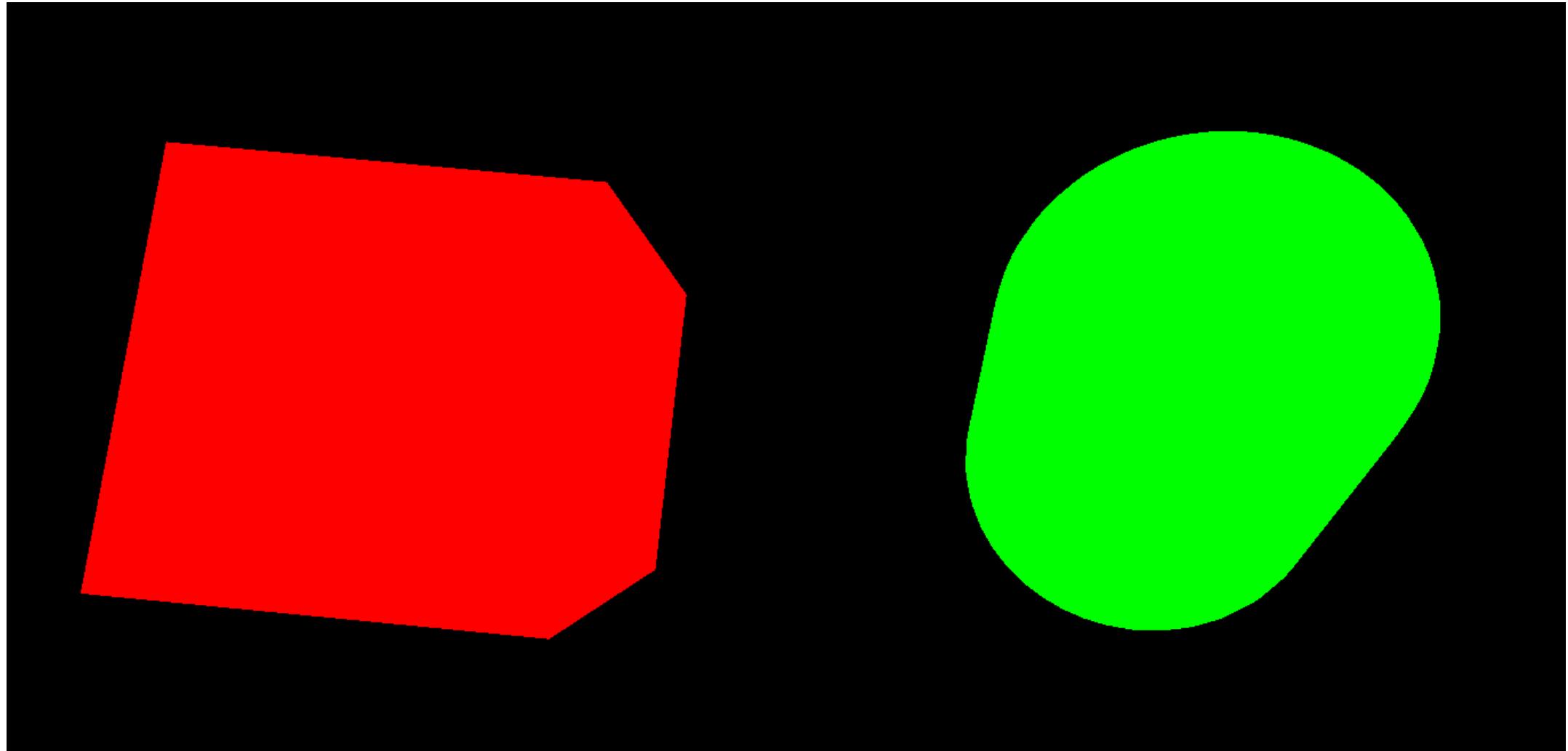
Code Example

```
const geometry = new THREE.CapsuleGeometry( 1, 1, 4, 8 );
const material = new THREE.MeshBasicMaterial( {color: 0x00ff00} );
const capsule = new THREE.Mesh( geometry, material ); scene.add( capsule );
```

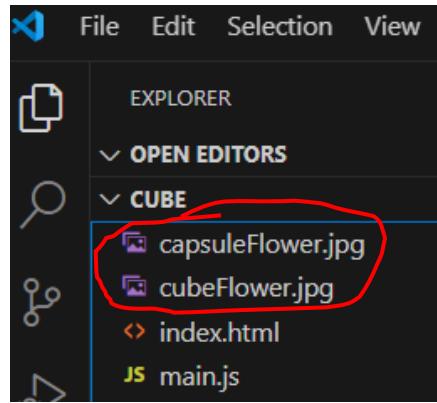
Cube + Capsule

```
JS main.js  X  
JS main.js > ...  
1 import * as THREE from 'three';  
2  
3 const scene = new THREE.Scene();  
4 const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );  
5  
6 const renderer = new THREE.WebGLRenderer();  
7 renderer.setSize( window.innerWidth, window.innerHeight );  
8 renderer.setAnimationLoop( animate );  
9 document.body.appendChild( renderer.domElement );  
10  
11 const geometry = new THREE.BoxGeometry( 1, 1, 1 );  
12 const material = new THREE.MeshBasicMaterial( { color: 0xff0000 } );  
13 const cube = new THREE.Mesh( geometry, material );  
14 scene.add( cube );  
15  
16 const capsuleGeometry = new THREE.CapsuleGeometry( 0.5, 0.5, 10, 40 );  
17 const capsuleMaterial = new THREE.MeshBasicMaterial( {color: 0x00ff00} );  
18 const capsule = new THREE.Mesh( capsuleGeometry, capsuleMaterial );  
19 scene.add( capsule );  
20  
21 camera.position.z = 2;  
22  
23 // Pozycjonowanie figur obok siebie  
24 cube.position.x = -1; // Przesuń sześciąn w lewo  
25 capsule.position.x = 1; // Przesuń kapsułę w prawo  
26  
27  
28 function animate() {  
29  
30     cube.rotation.x += 0.01;  
31     cube.rotation.y += 0.01;  
32  
33     capsule.rotation.x += 0.09;  
34     capsule.rotation.y += 0.03;  
35  
36     renderer.render( scene, camera );  
37  
38 }
```

Cube + Capsule



Cube + Capsule + texture



```
JS main2.js > ...
1 import * as THREE from 'three';
2
3 const scene = new THREE.Scene();
4 const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );
5
6 const renderer = new THREE.WebGLRenderer();
7 renderer.setSize( window.innerWidth, window.innerHeight );
8 renderer.setAnimationLoop( animate );
9 document.body.appendChild( renderer.domElement );
10
11 const geometry = new THREE.BoxGeometry( 1, 1, 1 );
12 const texture = new THREE.TextureLoader().load('cubeFlower.jpg');
13 const material = new THREE.MeshBasicMaterial( { map: texture } );
14 const cube = new THREE.Mesh( geometry, material );
15 scene.add( cube );
16
17 const capsuleGeometry = new THREE.CapsuleGeometry( 0.5, 0.5, 10, 40 );
18 const capsuleTexture = new THREE.TextureLoader().load('capsuleFlower.jpg');
19 const capsuleMaterial = new THREE.MeshBasicMaterial( { map: capsuleTexture } );
20 const capsule = new THREE.Mesh( capsuleGeometry, capsuleMaterial );
21 scene.add( capsule );
22
23 camera.position.z = 2;
24
25 // Pozycjonowanie figur obok siebie
26 cube.position.x = -1; // Przesuń sześcian w lewo
27 capsule.position.x = 1; // Przesuń kapsułę w prawo
28
29
30 function animate() {
31
32     cube.rotation.x += 0.01;
33     cube.rotation.y += 0.01;
34
35     capsule.rotation.x += 0.03;
36     capsule.rotation.y += 0.03;
37
38     renderer.render( scene, camera );
39
40 }
```

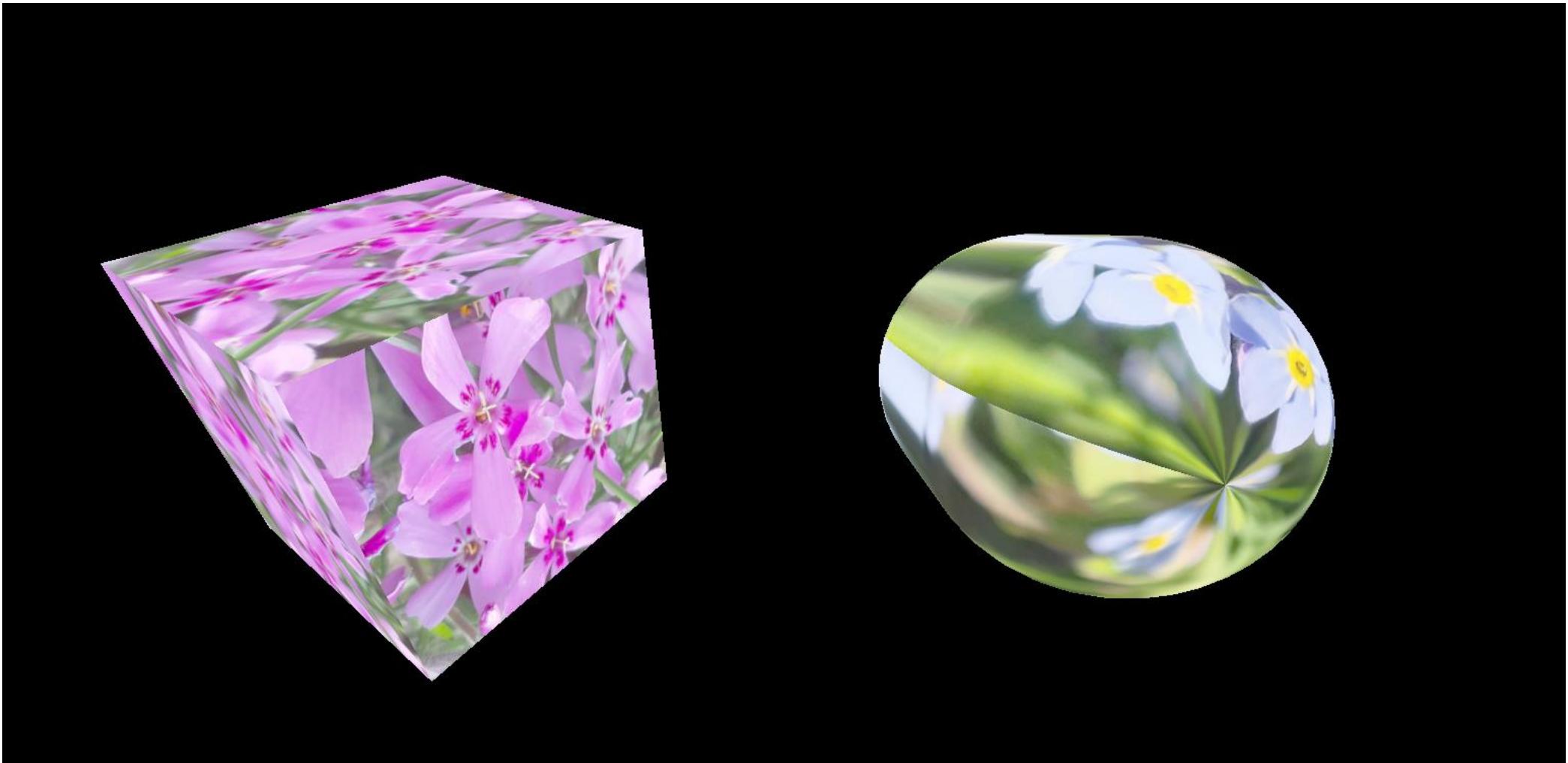


cubeFlower.jpg

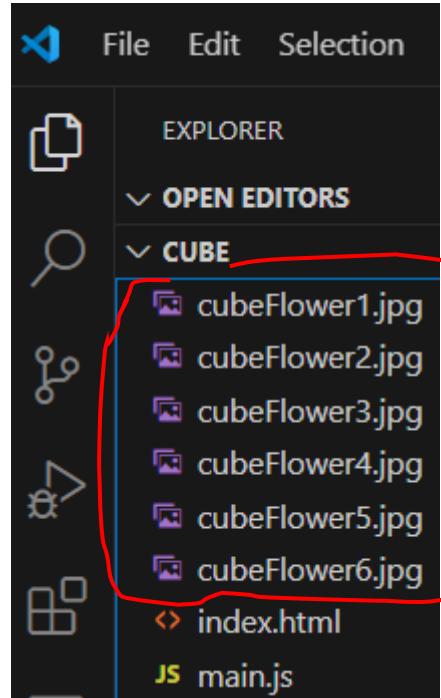


capsuleFlower.jpg

Cube + Capsule + texture

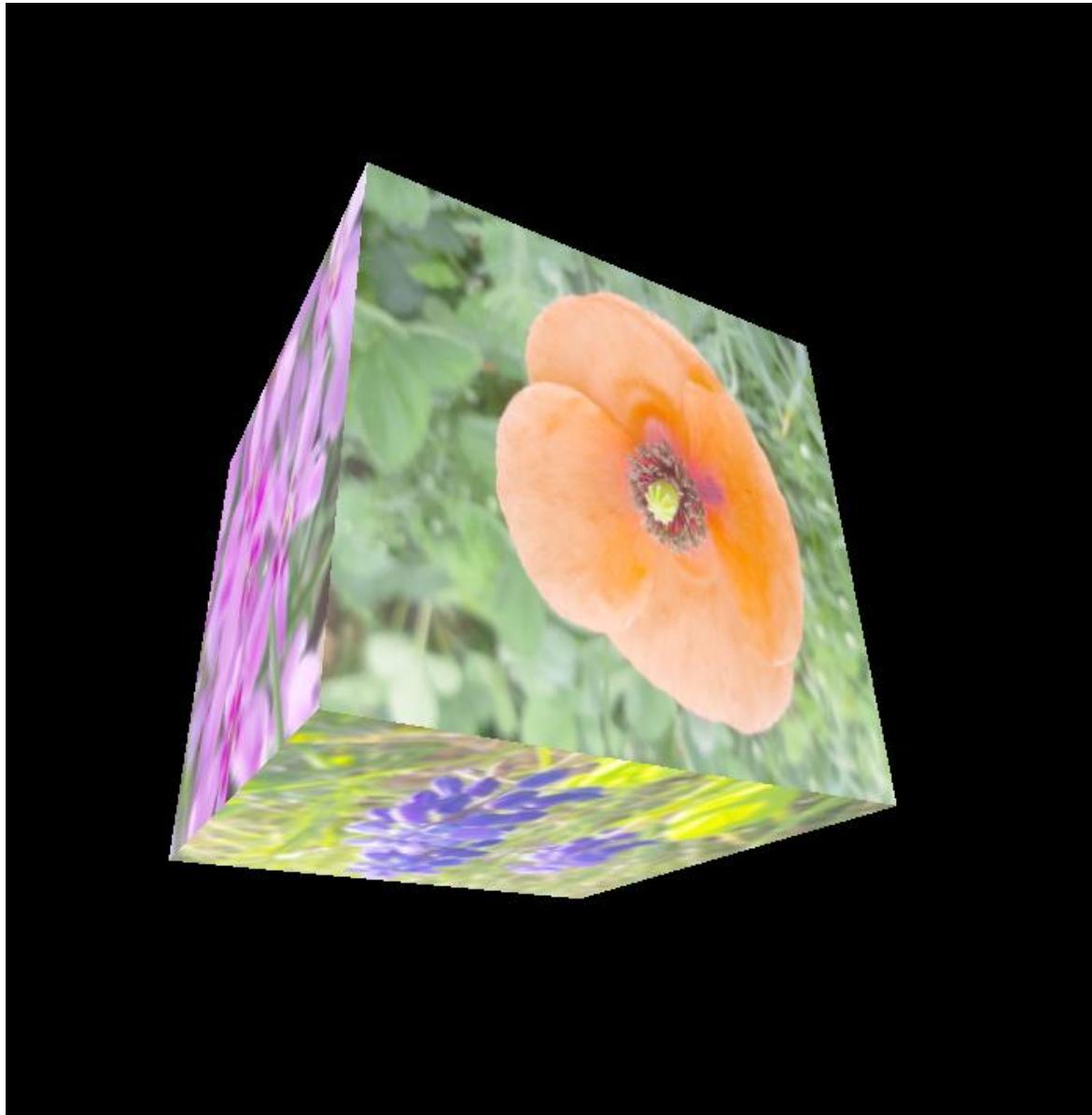


Cube + textures



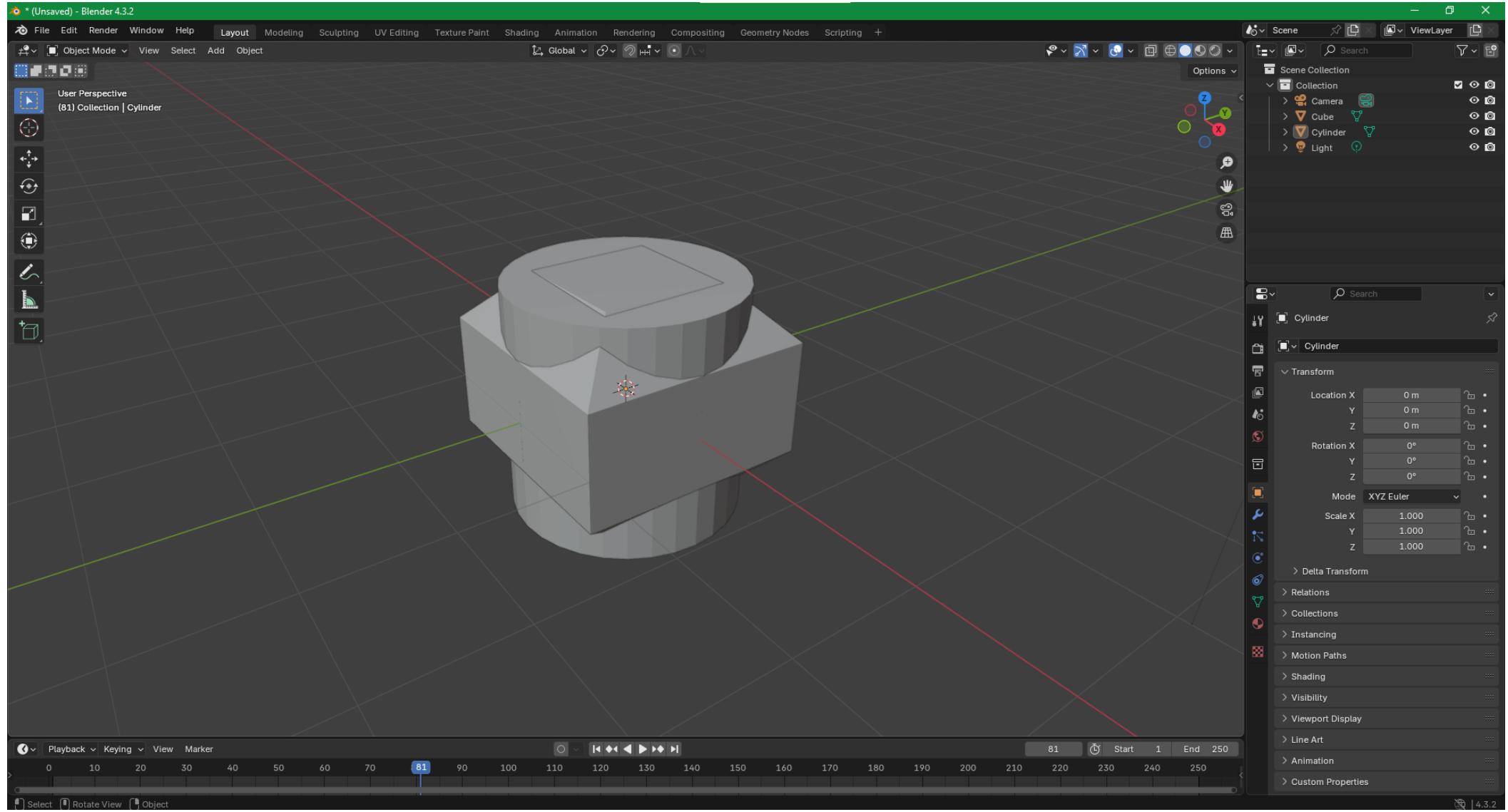
```
JS main3.js > ...
1 import * as THREE from 'three';
2
3 const scene = new THREE.Scene();
4 const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );
5
6 const renderer = new THREE.WebGLRenderer();
7 renderer.setSize( window.innerWidth, window.innerHeight );
8 renderer.setAnimationLoop( animate );
9 document.body.appendChild( renderer.domElement );
10
11 // tekstury dla każdej ściany w cube
12 const texturePaths = [
13     './cubeFlower1.jpg', // Przód
14     './cubeFlower2.jpg', // Tył
15     './cubeFlower3.jpg', // Góra
16     './cubeFlower4.jpg', // Dół
17     './cubeFlower5.jpg', // Lewo
18     './cubeFlower6.jpg', // Prawo
19 ];
20
21 // tworzy tablicę tekstur
22 const textures = texturePaths.map(path => new THREE.TextureLoader().load(path));
23 // tworzy tablicę materiałów
24 const materials = textures.map(texture => new THREE.MeshBasicMaterial({ map: texture }));
25
26 const geometry = new THREE.BoxGeometry( 1, 1, 1 );
27 const cube = new THREE.Mesh( geometry, materials );
28 scene.add( cube );
29
30 camera.position.z = 2;
31
32 function animate() {
33
34     cube.rotation.x += 0.01;
35     cube.rotation.y += 0.01;
36
37     renderer.render( scene, camera );
38
39 }
```

Cube + textures

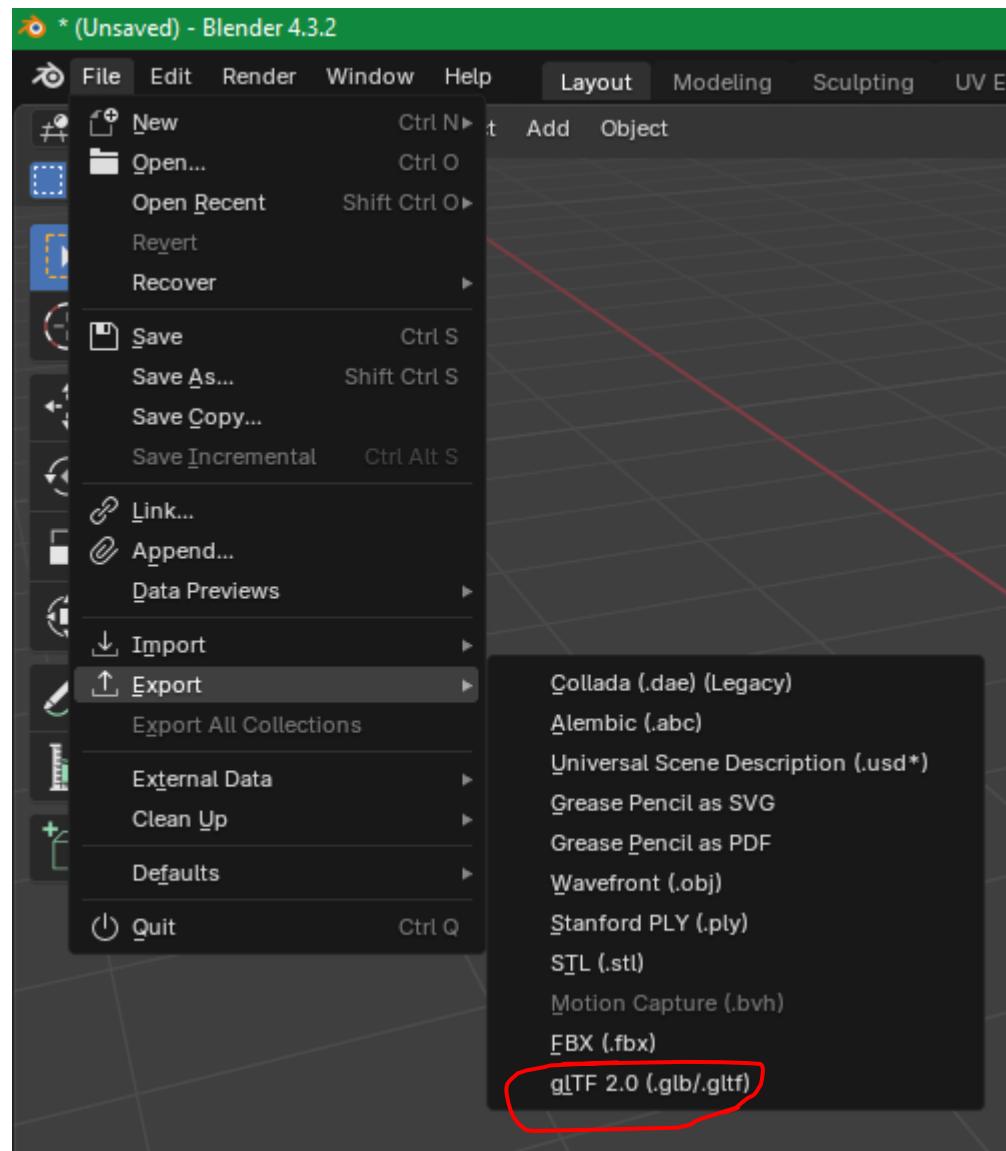


glb models

tworzymy model glb w aplikacji *Blender*



glb models



stworzony model eksportujemy
jako plik glb

darmowe modele glb

free glb models

Free asset packs [View all >](#)

Scifi Gun Pack
Quaternius 7 models

Homely House Set
Isa Lousberg 27 models

100 Avatars R2
Polygonal Mind 94 models

Free 3D models

Jet
Poly by Google

Mazda RX-7
IvOfficial

Car
Poly by Google

<https://rigmodels.com/index.php?searchkeyword=glb-model>

<https://poly.pizza/>

node.js instalacja

Run JavaScript Everywhere

Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.

[Download Node.js \(LTS\) ↗](#)

Downloads Node.js **v22.14.0¹** with long-term support.
Node.js can also be installed via [version managers](#).

Want new features sooner? Get **Node.js v23.8.0^{↗1}** instead.

[Create an HTTP Server](#) [Write Tests](#) [Read and Hash a File](#) [Streams Pipeline](#) [Work with Threads](#)

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with `node server.mjs`
```

JavaScript

[Copy to clipboard](#)

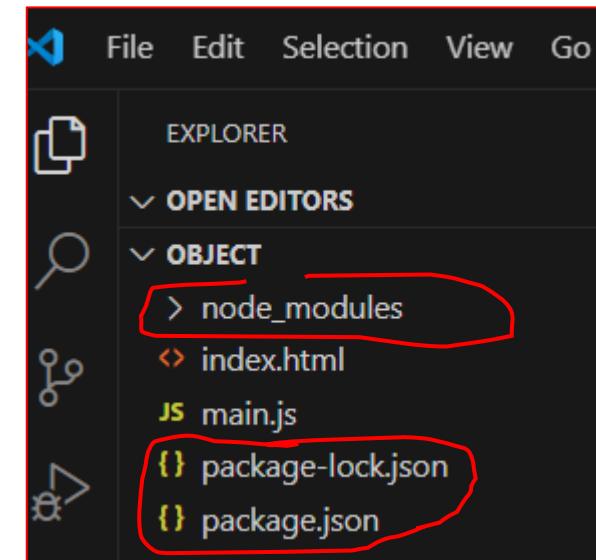
Learn more what Node.js is able to offer with our [Learning materials](#).

<https://nodejs.org/>

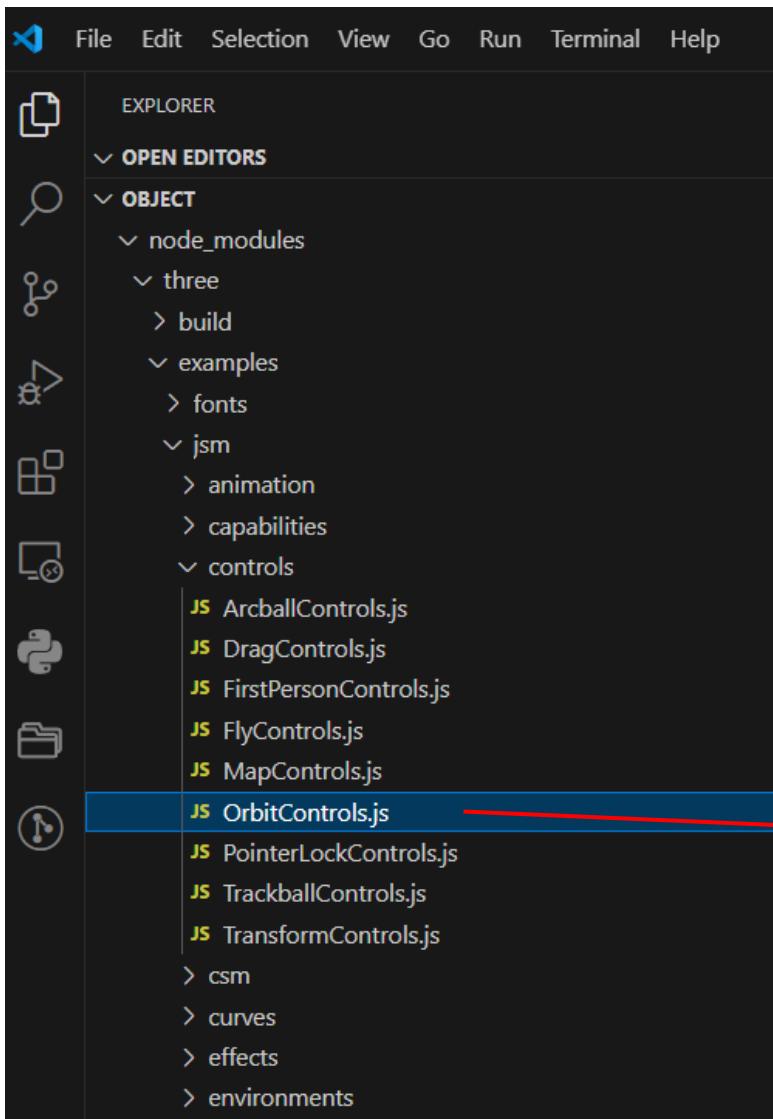
three.js instalacja

tworzymy katalog Object i instalujemy w nim three (w terminalu)

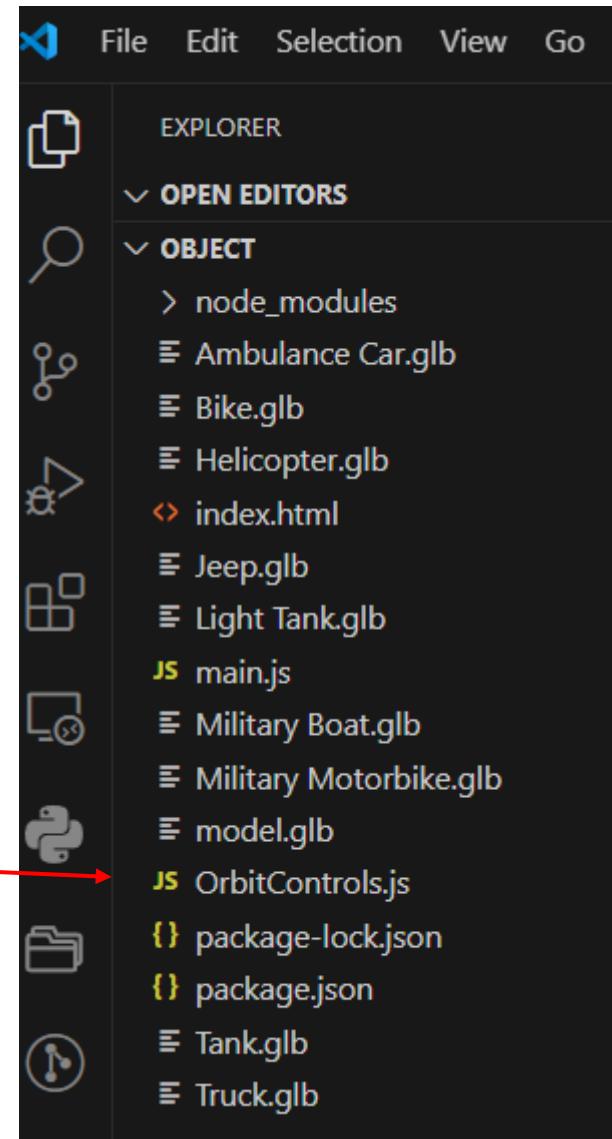
```
npm install --save three
```



glb models



OrbitControls.js kopujemy do katalogu projektu



glb models

index.html

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists various files and folders. The 'OPEN EDITORS' section is expanded, showing the contents of the 'index.html' file. This file contains the following code:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset='utf-8'>
    <title>Three.js</title>
    <script type="importmap">
        {
            "imports": {
                "three": "https://cdn.jsdelivr.net/npm/three@0.173.0/build/three.module.js",
                "three/addons/": "https://cdn.jsdelivr.net/npm/three@0.173.0/examples/jsm/"
            }
        }
    </script>
</head>
<body>
    <script type="module" src="/main.js"></script>
</body>
</html>
```

A red oval highlights the 'index.html' entry in the Explorer sidebar. The code editor's title bar also displays 'index.html'.

glb models

main.js 1/2

```
js main.js  x |  
js main.js > ...  
1 import * as THREE from 'three';  
2 import { GLTFLoader } from 'three/addons/loaders/GLTFLoader.js';  
3 import { OrbitControls } from './OrbitControls.js';  
4  
5 const scene = new THREE.Scene();  
6 const camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.1, 1000 );  
7  
8 const renderer = new THREE.WebGLRenderer();  
9 renderer.setSize( window.innerWidth, window.innerHeight );  
10 renderer.setPixelRatio( window.devicePixelRatio );  
11 document.body.appendChild( renderer.domElement );  
12  
13 // Dodaj światło  
14 const ambientLight = new THREE.AmbientLight( 0xffffff, 1.5 );  
15 scene.add( ambientLight );  
16  
17 const directionalLight = new THREE.DirectionalLight( 0xffffff, 1.5 );  
18 directionalLight.position.set( 1, 1, 1 ).normalize();  
19 scene.add( directionalLight );  
20  
21 const pointLight = new THREE.PointLight( 0xffffff, 1, 100 );  
22 pointLight.position.set( 0, 2, 0 );  
23 scene.add( pointLight );  
24  
25 camera.position.z = 5;  
26  
27 // Utwórz OrbitControls  
28 const controls = new OrbitControls( camera, renderer.domElement );  
29  
30 // Opcjonalnie: Ustaw punkt docelowy kontroli  
31 controls.target.set( 0, 0, 0 );  
32 controls.update();
```

glb models

main.js 2/2

```
33
34 renderer.setClearColor(0xffffffff, 1); // Biały kolor tła
35
36 const loader = new GLTFLoader();
37
38 // załóż głb model
39 loader.load('model.glb', function ( gltf ) {
40
41     scene.add( gltf.scene );
42
43 }, undefined, function ( error ) {
44
45     console.error('Błąd ładowania modelu:', error );
46
47 } );
48
49 const animate = function () {
50     requestAnimationFrame( animate );
51
52     controls.update(); // Ważne: Uaktualnij kontrolę w każdej klatce
53     renderer.render( scene, camera );
54 };
55
56 animate();
```

nazwa modelu

glb models

model stworzony w *Blenderze* na
stronie internetowej

