

# Canvas First Step

2024 年 10 月 12 日

## 概要

WebGL を扱いやすくした three.js を利用して 3D アニメーションを作成してみよう。

## 目次

1	はじめに	2
1.1	読み間違えないでね . . . . .	2
1.2	注意 . . . . .	2
2	three.js 入門	3
2.1	05-11.html 立方体を配置してみよう . . . . .	3
2.2	05-12.html 立方体を回してみよう . . . . .	4
2.3	05-13.html インタラクティブ性を追加しよう . . . . .	5
2.4	05-21.html ライトを当ててみよう . . . . .	6
2.5	05-22.html 球に画像を貼ってみよう . . . . .	8
2.6	05-23.html マウスの X 軸によって見える方向を変えよう . . . . .	9
2.7	05-31.html 大量のオブジェクトを描画してみよう . . . . .	11
2.8	05-32.html 最適化してみよう . . . . .	13
3	できちゃった人	15

# 1 はじめに

## 1.1 読み間違えないでね

ソースコード 1 読み間違えないでね

---

```
1 数字: 0123456789
2 小文字:abcdefghijklmnopqrstuvwxyz
3 大文字:ABCDEFGHIJKLMNOPQRSTUVWXYZ
4
5 1:イチ
6 l:小文字のエル
7 i:小文字のアイ
8 !:ビックリマーク
9 |:バーティカルバー。Shift と ¥ を押したものの。
10
11 0:ゼロ
12 o:小文字のオー
13 O:大文字のオー
14
15 .:ピリオド
16 ,:コンマ
```

---

## 1.2 注意

- これから出てくるソースコードには、左に「行番号」と呼ばれる番号が出てくるけど、入力する必要はないからね。
- script タグの中で「//」で始まる文は、コメントで、プログラムは読み飛ばすよ。
- コピーできるところはコピーして効率よく入力して行こう
- 徐々に追加されていくから、量が多く見えるけど、平気だよ！

## 2 three.js 入門

参考 : <https://ics.media/tutorial-three/>

### 2.1 05-11.html 立方体を配置してみよう

ソースコード 2 05-11.html

---

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script>
7       // ページの読み込みを待つ
8       window.addEventListener('DOMContentLoaded', init);
9
10      function init() {
11        // サイズを指定
12        const width = 960;
13        const height = 540;
14
15        // レンダラーを作成
16        const renderer = new THREE.WebGLRenderer({
17          canvas: document.querySelector('#myCanvas'),
18        });
19        renderer.setPixelRatio(window.devicePixelRatio);
20        renderer.setSize(width, height);
21
22        // シーンを作成
23        const scene = new THREE.Scene();
24
25        // カメラを作成
26        const camera = new THREE.PerspectiveCamera(45, width / height);
27        camera.position.set(0, 0, +1000);
28
29        // 箱を作成
30        const geometry = new THREE.BoxGeometry(400, 400, 400);
31        const material = new THREE.MeshNormalMaterial();
32        const box = new THREE.Mesh(geometry, material);
33        scene.add(box);
34
35        // レンダリング
36        renderer.render(scene, camera);
37      }
38    </script>
39  </head>
```

```
40 <body>
41     <canvas id="myCanvas"></canvas>
42 </body>
43 </html>
```

---

正方形が見えるだけで、立方体だってわからないね。

## 2.2 05-12.html 立方体を回してみよう

ソースコード 3 05-12.html

---

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8"/>
5   <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6   <script>
7     // ページの読み込みを待つ
8     window.addEventListener('DOMContentLoaded', init);
9
10    function init() {
11
12      // サイズを指定
13      const width = 960;
14      const height = 540;
15
16      // レンダラーを作成
17      const renderer = new THREE.WebGLRenderer({
18        canvas: document.querySelector('#myCanvas')
19      });
20      renderer.setPixelRatio(window.devicePixelRatio);
21      renderer.setSize(width, height);
22
23      // シーンを作成
24      const scene = new THREE.Scene();
25
26      // カメラを作成
27      const camera = new THREE.PerspectiveCamera(45, width / height);
28      camera.position.set(0, 0, +1000);
29
30      // 箱を作成
31      const geometry = new THREE.BoxGeometry(400, 400, 400);
32      const material = new THREE.MeshNormalMaterial();
33      const box = new THREE.Mesh(geometry, material);
34      scene.add(box);
35
36      tick();
37
38      // 毎フレーム時に実行されるループイベントです
```

```

39     function tick() {
40         box.rotation.y += 0.01;
41         renderer.render(scene, camera); // レンダリング
42
43         requestAnimationFrame(tick);
44     }
45 }
46 </script>
47 </head>
48 <body>
49     <canvas id="myCanvas"></canvas>
50 </body>
51 </html>

```

---

## 2.3 05-13.html インタラクティブ性を追加しよう

6 行目に注意

ソースコード 4 05-13.html

---

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8"/>
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script src="https://unpkg.com/three@0.137.4/examples/js/controls/OrbitControls.js
      "></script>
7     <script>
8         // ページの読み込みを待つ
9         window.addEventListener('DOMContentLoaded', init);
10
11     function init() {
12
13         // サイズを指定
14         const width = 960;
15         const height = 540;
16
17         // レンダラーを作成
18         const renderer = new THREE.WebGLRenderer({
19             canvas: document.querySelector('#myCanvas')
20         });
21         renderer.setPixelRatio(window.devicePixelRatio);
22         renderer.setSize(width, height);
23
24         // シーンを作成
25         const scene = new THREE.Scene();
26
27         // カメラを作成
28         const camera = new THREE.PerspectiveCamera(45, width / height);

```

```

29     camera.position.set(0, 0, +1000);
30
31     // カメラコントローラーを作成
32     const controls = new THREE.OrbitControls(camera, document.body);
33
34     // 箱を作成
35     const geometry = new THREE.BoxGeometry(400, 400, 400);
36     const material = new THREE.MeshNormalMaterial();
37     const box = new THREE.Mesh(geometry, material);
38     scene.add(box);
39
40     tick();
41
42     // 毎フレーム時に実行されるループイベントです
43     function tick() {
44         box.rotation.y += 0.01;
45         renderer.render(scene, camera); // レンダリング
46
47         requestAnimationFrame(tick);
48     }
49 }
50 </script>
51 </head>
52 <body>
53     <canvas id="myCanvas"></canvas>
54 </body>
55 </html>

```

---

## 2.4 05-21.html ライトを当ててみよう

05-12.html を改造したほうがいいかも

ソースコード 5 05-21.html

---

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script>
7       // ページの読み込みを待つ
8       window.addEventListener('DOMContentLoaded', init);
9
10      function init() {
11        // サイズを指定
12        const width = 960;
13        const height = 540;
14
15        // レンダラーを作成

```

```

16     const renderer = new THREE.WebGLRenderer({
17         canvas: document.querySelector('#myCanvas'),
18     });
19     renderer.setSize(width, height);
20
21     // シーンを作成
22     const scene = new THREE.Scene();
23
24     // カメラを作成
25     const camera = new THREE.PerspectiveCamera(45, width / height);
26     camera.position.set(0, 0, +1000);
27
28     // 球体を作成
29     const geometry = new THREE.SphereGeometry(300, 30, 30);
30     // マテリアルを作成
31     const material = new THREE.MeshStandardMaterial({ color: 0xff0000 });
32     // メッシュを作成
33     const mesh = new THREE.Mesh(geometry, material);
34     // 3D 空間にメッシュを追加
35     scene.add(mesh);
36
37     // 平行光源
38     const directionalLight = new THREE.DirectionalLight(0xffffff);
39     directionalLight.position.set(1, 1, 1);
40     // シーンに追加
41     scene.add(directionalLight);
42
43     tick();
44
45     // 毎フレーム時に実行されるループイベントです
46     function tick() {
47         // レンダリング
48         renderer.render(scene, camera);
49
50         requestAnimationFrame(tick);
51     }
52 }
53 </script>
54 </head>
55 <body>
56     <canvas id="myCanvas"></canvas>
57 </body>
58 </html>

```

---

## 2.5 05-22.html 球に画像を貼ってみよう

<http://planetpixelemporium.com/earth.html> から、color map をダウンロードして、img/earthmap1k.jpg に配置しておこう

ソースコード 6 一定時間ごとにランダムな長方形を描こう

---

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8"/>
5   <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6   <script>
7     // ページの読み込みを待つ
8     window.addEventListener('DOMContentLoaded', init);
9
10    // サイズを指定
11    const width = 960;
12    const height = 540;
13
14    function init() {
15      // レンダラーを作成
16      const renderer = new THREE.WebGLRenderer({
17        canvas: document.querySelector('#myCanvas')
18      });
19      renderer.setSize(width, height);
20
21      // シーンを作成
22      const scene = new THREE.Scene();
23
24      // カメラを作成
25      const camera = new THREE.PerspectiveCamera(45, width / height, 1,
26        10000);
27      camera.position.set(0, 0, +1000);
28
29      // 球体を作成
30      const geometry = new THREE.SphereGeometry(300, 30, 30);
31      // 画像を読み込む
32      const loader = new THREE.TextureLoader();
33      const texture = loader.load('img/earthmap1k.jpg');
34      // マテリアルにテクスチャーを設定
35      const material = new THREE.MeshStandardMaterial({
36        map: texture
37      });
38      // メッシュを作成
39      const mesh = new THREE.Mesh(geometry, material);
40      // 3D 空間にメッシュを追加
41      scene.add(mesh);
```



```

41
42     // 平行光源
43     const directionalLight = new THREE.DirectionalLight(0xFFFFFF);
44     directionalLight.position.set(1, 1, 1);
45     // シーンに追加
46     scene.add(directionalLight);
47
48     tick();
49
50     // 毎フレーム時に実行されるループイベントです
51     function tick() {
52         // メッシュを回転させる
53         mesh.rotation.y += 0.01;
54         // レンダリング
55         renderer.render(scene, camera);
56
57         requestAnimationFrame(tick);
58     }
59 }
60 </script>
61 </head>
62 <body>
63     <canvas id="myCanvas"></canvas>
64 </body>
65 </html>

```

---

## 2.6 05-23.html マウスのX軸によって見える方向を変えよう

ソースコード 7 05-23.html

---

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8"/>
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script>
7         // ページの読み込みを待つ
8         window.addEventListener('DOMContentLoaded', init);
9
10        // サイズを指定
11        const width = 960;
12        const height = 540;
13
14        function init() {
15            let rot = 0; // 角度
16            let mouseX = 0; // マウス座標
17
18            // マウス座標はマウスが動いた時のみ取得できる

```

```

19         document.addEventListener("mousemove", (event) => {
20             mouseX = event.pageX;
21         });
22
23         // レンダラーを作成
24         const renderer = new THREE.WebGLRenderer({
25             canvas: document.querySelector('#myCanvas')
26         });
27         renderer.setSize(width, height);
28
29         // シーンを作成
30         const scene = new THREE.Scene();
31
32         // カメラを作成
33         const camera = new THREE.PerspectiveCamera(45, width / height, 1,
34             10000);
35         camera.position.set(0, 0, +1000);
36
37         // 球体を作成
38         const geometry = new THREE.SphereGeometry(300, 30, 30);
39         // 画像を読み込む
40         const loader = new THREE.TextureLoader();
41         const texture = loader.load('img/earthmap1k.jpg');
42         // マテリアルにテクスチャーを設定
43         const material = new THREE.MeshStandardMaterial({
44             map: texture
45         });
46         // メッシュを作成
47         const mesh = new THREE.Mesh(geometry, material);
48         // 3D 空間にメッシュを追加
49         scene.add(mesh);
50
51         // 平行光源
52         const directionalLight = new THREE.DirectionalLight(0xFFFFFF);
53         directionalLight.position.set(1, 1, 1);
54         // シーンに追加
55         scene.add(directionalLight);
56
57         tick();
58
59         // 毎フレーム時に実行されるループイベントです
60         function tick() {
61             // メッシュを回転させる
62             mesh.rotation.y += 0.01;
63
64             // マウスの位置に応じて角度を設定
65             // マウスのX座標がステージの幅の何%の位置にあるか調べてそれを360度で乗算
66             // する

```

```

65         const targetRot = (mouseX / window.innerWidth) * 360;
66         // イージングの公式を用いて滑らかにする
67         // 値 += (目標値 - 現在の値) * 減速値
68         rot += (targetRot - rot) * 0.02;
69
70         // ラジアンに変換する
71         const radian = rot * Math.PI / 180;
72         // 角度に応じてカメラの位置を設定
73         camera.position.x = 1000 * Math.sin(radian);
74         camera.position.z = 1000 * Math.cos(radian);
75         // 原点方向を見つめる
76         camera.lookAt(new THREE.Vector3(0, 0, 0));
77
78
79         // レンダリング
80         renderer.render(scene, camera);
81
82         requestAnimationFrame(tick);
83     }
84 }
85 </script>
86 </head>
87 <body>
88     <canvas id="myCanvas"></canvas>
89 </body>
90 </html>

```

---

## 2.7 05-31.html 大量のオブジェクトを描画してみよう

ソースコード 8 05-31.html

---

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script src="https://unpkg.com/stats.js@0.17.0/build/stats.min.js"></script>
7     <script>
8       // ページの読み込みを待つ
9       window.addEventListener('DOMContentLoaded', init);
10
11       function init() {
12         // サイズを指定
13         const width = 960;
14         const height = 540;
15         // 1辺あたりに配置するオブジェクトの個数
16         const CELL_NUM = 20;
17

```

```

18 // レンダラーを作成
19 const renderer = new THREE.WebGLRenderer({
20     canvas: document.querySelector('#myCanvas'),
21 });
22 renderer.setPixelRatio(window.devicePixelRatio);
23 renderer.setSize(width, height);
24
25 // シーンを作成
26 const scene = new THREE.Scene();
27
28 // カメラを作成
29 const camera = new THREE.PerspectiveCamera(45, width / height);
30 camera.position.set(0, 0, 400);
31
32 const container = new THREE.Group();
33 scene.add(container);
34
35 // 共通マテリアル
36 const material = new THREE.MeshNormalMaterial();
37 // Box
38 for (let i = 0; i < CELL_NUM; i++) {
39     for (let j = 0; j < CELL_NUM; j++) {
40         for (let k = 0; k < CELL_NUM; k++) {
41             // 立方体個別の要素を作成
42             const mesh = new THREE.Mesh(new THREE.BoxGeometry(5, 5, 5), material);
43
44             // XYZ 座標を設定
45             mesh.position.set(10 * (i - CELL_NUM / 2), 10 * (j - CELL_NUM / 2),
46                 10 * (k - CELL_NUM / 2));
47
48             // メッシュを 3D 空間に追加
49             container.add(mesh);
50         }
51     }
52 }
53
54 // フレームレートの数値を画面に表示
55 const stats = new Stats();
56 stats.domElement.style.position = 'absolute';
57 stats.domElement.style.top = '0px';
58 document.body.appendChild(stats.domElement);
59
60 tick();
61
62 // 毎フレーム時に実行されるループイベントです
63 function tick() {
64     container.rotation.x += Math.PI / 180;
65     container.rotation.y += Math.PI / 180;

```

```

65
66     // レンダリング
67     renderer.render(scene, camera);
68
69     // レンダリング情報を画面に表示
70     document.getElementById('info').innerHTML = JSON.stringify(renderer.info.
        render, '', ' ');
71
72     // フレームレートを表示
73     stats.update();
74
75     requestAnimationFrame(tick);
76 }
77 }
78 </script>
79 </head>
80 <body>
81     <canvas id="myCanvas"></canvas>
82     <pre id="info"></pre>
83 </body>
84 </html>

```

---

## 2.8 05-32.html 最適化してみよう

ソースコード 9 05-32.html

---

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <script src="https://unpkg.com/three@0.137.4/build/three.min.js"></script>
6     <script src="https://unpkg.com/three@0.137.4/examples/js/utils/
        BufferGeometryUtils.js"></script>
7     <script src="https://unpkg.com/stats.js@0.17.0/build/stats.min.js"></script>
8
9     <script>
10      // ページの読み込みを待つ
11      window.addEventListener('DOMContentLoaded', init);
12
13      function init() {
14        // サイズを指定
15        const width = 960;
16        const height = 540;
17
18        // レンダラーを作成
19        const renderer = new THREE.WebGLRenderer({
20          canvas: document.querySelector('#myCanvas'),
21        });

```

```

22     renderer.setPixelRatio(window.devicePixelRatio);
23     renderer.setSize(width, height);
24
25     // シーンを作成
26     const scene = new THREE.Scene();
27
28     // カメラを作成
29     const camera = new THREE.PerspectiveCamera(45, width / height);
30     camera.position.set(0, 0, 400);
31
32     // 1辺あたりに配置するオブジェクトの個数
33     const CELL_NUM = 25;
34     // 結合用のジオメトリを格納する配列
35     const boxes = [];
36     for (let i = 0; i < CELL_NUM; i++) {
37         for (let j = 0; j < CELL_NUM; j++) {
38             for (let k = 0; k < CELL_NUM; k++) {
39                 // 立方体個別の要素を作成
40                 const geometryBox = new THREE.BoxGeometry(5, 5, 5);
41
42                 // 座標調整
43                 const geometryTranslated = geometryBox.translate(
44                     10 * (i - CELL_NUM / 2),
45                     10 * (j - CELL_NUM / 2),
46                     10 * (k - CELL_NUM / 2)
47                 );
48
49                 // ジオメトリを保存
50                 boxes.push(geometryTranslated);
51             }
52         }
53     }
54     // ジオメトリを生成
55     const geometry = THREE.BufferGeometryUtils.mergeBufferGeometries(boxes);
56
57     // マテリアルを作成
58     const material = new THREE.MeshNormalMaterial();
59     // メッシュを作成
60     const mesh = new THREE.Mesh(geometry, material);
61     scene.add(mesh);
62
63     // フレームレートの数値を画面に表示
64     const stats = new Stats();
65     stats.domElement.style.position = 'absolute';
66     stats.domElement.style.top = '10px';
67     document.body.appendChild(stats.domElement);
68
69     tick();

```

```

70
71     // 毎フレーム時に実行されるループイベントです
72     function tick() {
73         mesh.rotation.x += Math.PI / 180;
74         mesh.rotation.y += Math.PI / 180;
75
76         // レンダリング
77         renderer.render(scene, camera);
78
79         // レンダリング情報を画面に表示
80         document.getElementById('info').innerHTML = JSON.stringify(renderer.info.
            render, '', ' ');
81
82         // フレームレートを表示
83         stats.update();
84
85         requestAnimationFrame(tick);
86     }
87 }
88 </script>
89 </head>
90 <body>
91     <canvas id="myCanvas"></canvas>
92     <pre id="info"></pre>
93 </body>
94 </html>

```

---

### 3 できちゃった人

オブジェクト指向を使った

<https://ics.media/tutorial-three/class/>

[https://ics.media/tutorial-three/class\\_method/](https://ics.media/tutorial-three/class_method/)

をトライしてみよう。

以上