# Back to basics – part 2 – Three.js

Notes: Using an API, less code, but bigger download – balance NIH and DRY against YGNI, possible limitations (e.g. Euler angles may cause gimbal lock and applied in particular making a particular rotation problematic or even impossible).

## 1 – Rotatable Cube

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Three.js coloured cube</title>
  <script type="text/javascript" src="three.min.js"></script>
  <script src="OrbitControls.js"></script>
</head>
<body>
  <script type="text/javascript">
    var cube;
    var scene = new THREE.Scene();
    var camera = new THREE.PerspectiveCamera(75, 500 / 500, 0.1, 1000);
    camera.position.z = 20;

    var renderer = new THREE.WebGLRenderer();
    renderer.setSize(500, 500);
    document.body.appendChild(renderer.domElement);

    var cubeMaterials = [];
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0xff3333 }));
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0xff8800 }));
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0xffff33 }));
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0x33ff33 }));
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0x3333ff }));
    cubeMaterials.push(new THREE.MeshBasicMaterial({ color: 0x8833ff }));

    var cubeMaterials = new THREE.MeshFaceMaterial(cubeMaterials);
    var cubeGeometry = new THREE.BoxGeometry(10, 10, 10);
    cube = new THREE.Mesh(cubeGeometry, cubeMaterials);
    scene.add(cube);

    var controls = new THREE.OrbitControls(camera, renderer.domElement);

    (function render() {
      requestAnimationFrame(render);
      renderer.render(scene, camera);
      controls.update();
    })();
  </script>
</body>
</html>
```

## 2 – Rotatable Cube → Dice (texture from base64)

Notes: Cross Origin Resource Sharing (CORS) can cause problems. One way around it is using encoding.

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Three.js coloured cube</title>
  <script type="text/javascript" src="three.min.js"></script>
  <script src="OrbitControls.js"></script>
</head>
<body>
  <script type="text/javascript">
    var cube;
    var scene = new THREE.Scene();
    var camera = new THREE.PerspectiveCamera(75, 500 / 500, 0.1, 1000);
    var textures = [];
    for (var i = 1; i < 7; i++) {
      textures.push(THREE.ImageUtils.loadTexture(imagesSource[i]));
    }

    camera.position.z = 20;

    var renderer = new THREE.WebGLRenderer();
    renderer.setSize(500, 500);
    document.body.appendChild(renderer.domElement);

    var cubeMaterials = [];
    for (var i = 0; i < 6; i++) {
      cubeMaterials.push(new THREE.MeshBasicMaterial({ map: textures[i] }));
    }

    var cubeMaterials = new THREE.MeshFaceMaterial(cubeMaterials);
    var cubeGeometry = new THREE.BoxGeometry(10, 10, 10);
    cube = new THREE.Mesh(cubeGeometry, cubeMaterials);
    scene.add(cube);

    var controls = new THREE.OrbitControls(camera, renderer.domElement);

    (function render() {
      requestAnimationFrame(render);
      renderer.render(scene, camera);
      controls.update();
    })();
  </script>
</body>
</html>
```

# 3 – Rotatable Cube → Dice (texture from file)

Notes: This will only work if you're serving the page on server, or using the appropriate browser settings.

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Three.js dice</title>
  <script type="text/javascript" src="three.min.js"></script>
  <script src="OrbitControls.js"></script>
</head>
<body>
  <script type="text/javascript">
    var cube;
    var scene = new THREE.Scene();
    var camera = new THREE.PerspectiveCamera(75, 500 / 500, 0.1, 1000);
    var textures = [];
    for (var i = 1; i < 7; i++) {
      textures.push(THREE.ImageUtils.loadTexture(i + ".png"));
    }

    camera.position.z = 20;

    var renderer = new THREE.WebGLRenderer();
    renderer.setSize(500, 500);
    document.body.appendChild(renderer.domElement);

    var cubeMaterials = [];
    for (var i = 0; i < 6; i++) {
      cubeMaterials.push(new THREE.MeshBasicMaterial({ map: textures[i] }));
    }

    var cubeMaterials = new THREE.MeshFaceMaterial(cubeMaterials);
    var cubeGeometry = new THREE.BoxGeometry(10, 10, 10);
    cube = new THREE.Mesh(cubeGeometry, cubeMaterials);
    scene.add(cube);

    var controls = new THREE.OrbitControls(camera, renderer.domElement);

    (function render() {
      requestAnimationFrame(render);
      renderer.render(scene, camera);
      controls.update();
    })();
  </script>
</body>
</html>
```

# Back to basics – part 2 – Babylon.js

## 4 – Rotatable Cube

Notes: Babylon.js takes more setting up than Three.js but seems more powerful when handling multi-textures.

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Babylon.js coloured cube</title>
</head>
<body>
  <div id="rootDiv">
    <canvas id="renderCanvas"></canvas>
  </div>
  <script src="babylon.min.js"></script>
  <script>
    var canvas = document.getElementById("renderCanvas");
    canvas.width = 500;
    canvas.height = 500;

    var engine = new BABYLON.Engine(canvas, true);
    var scene = new BABYLON.Scene(engine);

    // Create a camera looking at the origin (0,0,0)
    var camera = new BABYLON.ArcRotateCamera("Camera", 1, 0.8, 10, new BABYLON.Vector3(0, 0, 0), scene);

    // Create a light
    var light0 = new BABYLON.HemisphericLight("Hemi0", new BABYLON.Vector3(0, 1, 0), scene);
    light0.diffuse = new BABYLON.Color3(1, 1, 1);
    light0.specular = new BABYLON.Color3(1, 1, 1);
    light0.groundColor = new BABYLON.Color3(0, 0, 0);

    var materialBlue = new BABYLON.StandardMaterial("blue", scene);
    materialBlue.diffuseColor = new BABYLON.Color3(0.0,0.0, 1.0);
    var materialGreen = new BABYLON.StandardMaterial("green", scene);
    materialGreen.diffuseColor = new BABYLON.Color3(0.0, 1.0, 0.0);
    var materialRed = new BABYLON.StandardMaterial("red", scene);
    materialRed.diffuseColor = new BABYLON.Color3(1.0, 0.0, 0.0);
    var materialCyan = new BABYLON.StandardMaterial("cyan", scene);
    materialCyan.diffuseColor = new BABYLON.Color3(0.0, 1.0, 1.0);
    var materialMagenta = new BABYLON.StandardMaterial("magenta", scene);
    materialMagenta.diffuseColor = new BABYLON.Color3(1.0, 1.0, 0.0);
    var materialYellow = new BABYLON.StandardMaterial("yellow", scene);
    materialYellow.diffuseColor = new BABYLON.Color3(1.0, 0.0, 1.0);

    var multimat = new BABYLON.MultiMaterial("multi", scene);
    multimat.subMaterials.push(materialBlue);
    multimat.subMaterials.push(materialGreen);
    multimat.subMaterials.push(materialRed);
    multimat.subMaterials.push(materialCyan);
    multimat.subMaterials.push(materialMagenta);
    multimat.subMaterials.push(materialYellow);

    var box = BABYLON.Mesh.CreateBox("box", 4.0, scene);
    box.material = multimat;
    box.subMeshes = [];
    box.subMeshes.push(new BABYLON.SubMesh(0, 0, 4, 0, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(1, 4, 4, 6, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(2, 8, 4, 12, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(3, 12, 4, 18, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(4, 16, 4, 24, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(5, 20, 4, 30, 6, box));

    scene.activeCamera.attachControl(canvas);

    engine.runRenderLoop(function () {
      scene.render();
    });
  </script>
</body>
</html>
```

## 5 – Rotatable Cube → Dice (texture from base64)

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Babylon.js dice local encoded texture</title>
</head>

<body>
  <div id="rootDiv">
    <canvas id="renderCanvas"></canvas>
  </div>
  <script src="babylon.min.js"></script>
  <script src="dice_encoded.js"></script>
  <script>
    var canvas = document.getElementById("renderCanvas");
    canvas.width = 500;
    canvas.height = 500;

    var engine = new BABYLON.Engine(canvas, true);
    var scene = new BABYLON.Scene(engine);

    // Create a camera looking at the origin (0,0,0)
    var camera = new BABYLON.ArcRotateCamera("Camera", 1, 0.8, 10, new BABYLON.Vector3(0, 0, 0), scene);

    // Create a light
    var light0 = new BABYLON.HemisphericLight("Hemi0", new BABYLON.Vector3(0, 1, 0), scene);
    light0.diffuse = new BABYLON.Color3(1, 1, 1);
    light0.specular = new BABYLON.Color3(1, 1, 1);
    light0.groundColor = new BABYLON.Color3(0, 0, 0);

    var textures = [];
    for (var i = 0; i < 6; i++) {
      textures.push(new BABYLON.Texture(imagesSource[i], scene));
    }

    var materials = [];
    for (var i = 0; i < 6; i++) {
      materials.push(new BABYLON.StandardMaterial("texture" + i, scene));
      materials[i].diffuseTexture = textures[i];
    }

    var multimat = new BABYLON.MultiMaterial("multi", scene);
    for (var i = 0; i < 6; i++) {
      multimat.subMaterials.push(materials[i]);
    }

    // Create a cube of size 4
    var box = BABYLON.Mesh.CreateBox("box", 4.0, scene);
    box.material = multimat;
    box.subMeshes = [];
    box.subMeshes.push(new BABYLON.SubMesh(0, 0, 4, 0, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(1, 4, 4, 6, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(2, 8, 4, 12, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(3, 12, 4, 18, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(4, 16, 4, 24, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(5, 20, 4, 30, 6, box));

    scene.activeCamera.attachControl(canvas);

    // Once the scene is loaded, just register a render loop to render it
    engine.runRenderLoop(function () {
      scene.render();
    });
  </script>
</body>
</html>
```

## 6 – Rotatable Cube → Dice (texture from file)

```
<!DOCTYPE html>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Babylon.js dice</title>
</head>

<body>
  <div id="rootDiv">
    <canvas id="renderCanvas"></canvas>
  </div>
  <script src="babylon.min.js"></script>
  <script>
    var canvas = document.getElementById("renderCanvas");
    canvas.width = 500;
    canvas.height = 500;

    var engine = new BABYLON.Engine(canvas, true);
    var scene = new BABYLON.Scene(engine);

    // Create a camera looking at the origin (0,0,0)
    var camera = new BABYLON.ArcRotateCamera("Camera", 1, 0.8, 10, new BABYLON.Vector3(0, 0, 0), scene);

    // Create a light
    var light0 = new BABYLON.HemisphericLight("Hemi0", new BABYLON.Vector3(0, 1, 0), scene);
    light0.diffuse = new BABYLON.Color3(1, 1, 1);
    light0.specular = new BABYLON.Color3(1, 1, 1);
    light0.groundColor = new BABYLON.Color3(0, 0, 0);

    var textures = [];
    for (var i = 1; i < 7; i++) {
      textures.push(new BABYLON.Texture(i + ".png", scene));
    }

    var materials = [];
    for (var i = 0; i < 6; i++) {
      materials.push(new BABYLON.StandardMaterial("texture" + i, scene));
      materials[i].diffuseTexture = textures[i];
    }

    var multimat = new BABYLON.MultiMaterial("multi", scene);
    for (var i = 0; i < 6; i++) {
      multimat.subMaterials.push(materials[i]);
    }

    // Create a cube of size 4
    var box = BABYLON.Mesh.CreateBox("box", 4.0, scene);
    box.material = multimat;
    box.subMeshes = [];
    box.subMeshes.push(new BABYLON.SubMesh(0, 0, 4, 0, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(1, 4, 4, 6, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(2, 8, 4, 12, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(3, 12, 4, 18, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(4, 16, 4, 24, 6, box));
    box.subMeshes.push(new BABYLON.SubMesh(5, 20, 4, 30, 6, box));

    scene.activeCamera.attachControl(canvas);

    // Once the scene is loaded, just register a render loop to render it
    engine.runRenderLoop(function () {
      scene.render();
    });
  </script>
</body>
</html>
```