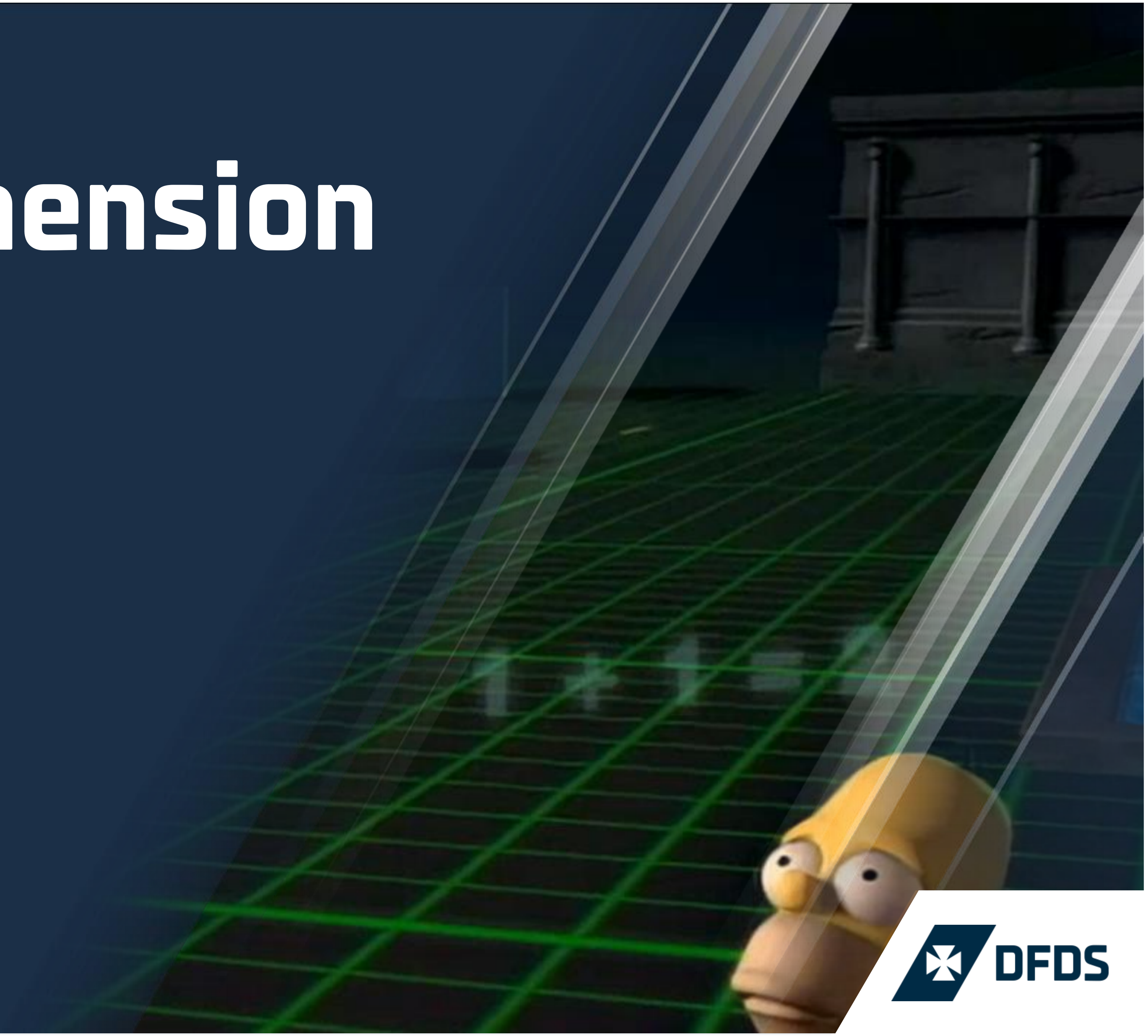
August 26h 2021

Adding a Dimension to React

React Three Fiber



Stop trying to do 3D effects with CSS

A case study on how not to do it

- My goal was to play create some glass morphism with some 3D effects.
  - With some cool background effect – Animated SVG
  - 3D scroll effect

- The result was ultra-low FPS and me losing interest in project :)
  - Blur effects are to CPU expensive
  - Rendering errors

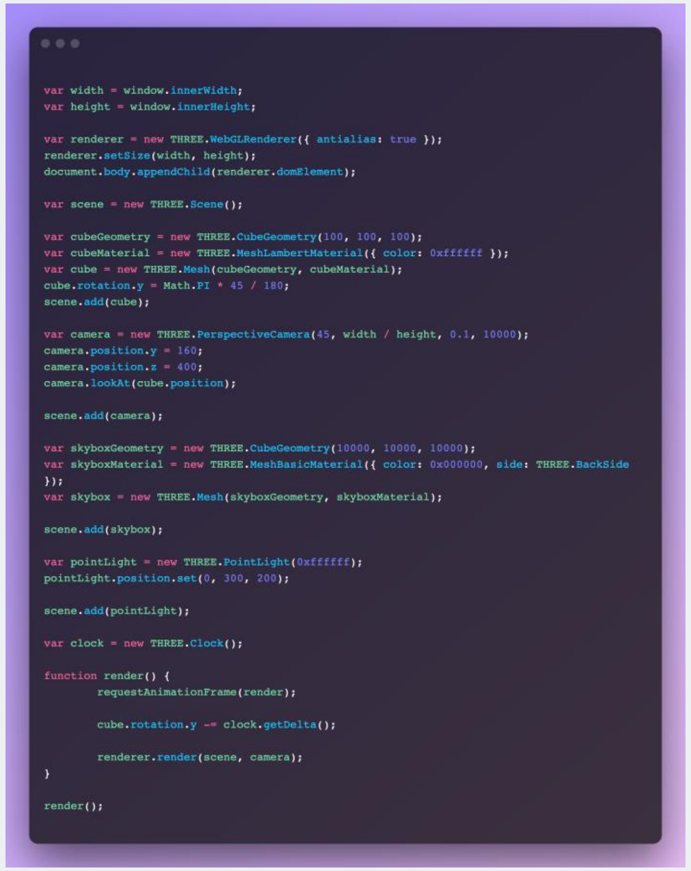
- There is an easier solutions out there to do cool effects

far away
lower
transparency
closer
higher
transparency
subtle
shadow
for extra
depth

DFDS

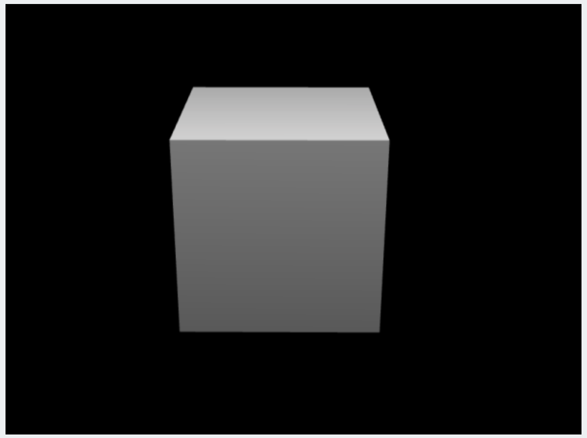
Three.js to the rescue

A WebGL library created by Ricardo Cabello in 2010 and have 1300 contributors on GitHub



```
var width = window.innerWidth;
var height = window.innerHeight;

var renderer = new THREE.WebGLRenderer({ antialias: true });
renderer.setSize(width, height);
document.body.appendChild(renderer.domElement);

var scene = new THREE.Scene();

var cubeGeometry = new THREE.CubeGeometry(100, 100, 100);
var cubeMaterial = new THREE.MeshLambertMaterial({ color: 0xffffff });
var cube = new THREE.Mesh(cubeGeometry, cubeMaterial);
cube.rotation.y = Math.PI * 45 / 180;
scene.add(cube);

var camera = new THREE.PerspectiveCamera(45, width / height, 0.1, 10000);
camera.position.y = 160;
camera.position.z = 400;
camera.lookAt(cube.position);

scene.add(camera);

var skyboxGeometry = new THREE.CubeGeometry(10000, 10000, 10000);
var skyboxMaterial = new THREE.MeshBasicMaterial({ color: 0x000000, side: THREE.BackSide
});
var skybox = new THREE.Mesh(skyboxGeometry, skyboxMaterial);

scene.add(skybox);

var pointLight = new THREE.PointLight(0xffffff);
pointLight.position.set(0, 300, 200);

scene.add(pointLight);

var clock = new THREE.Clock();

function render() {
        requestAnimationFrame(render);

        cube.rotation.y -= clock.getDelta();

        renderer.render(scene, camera);
}

render();
```

=



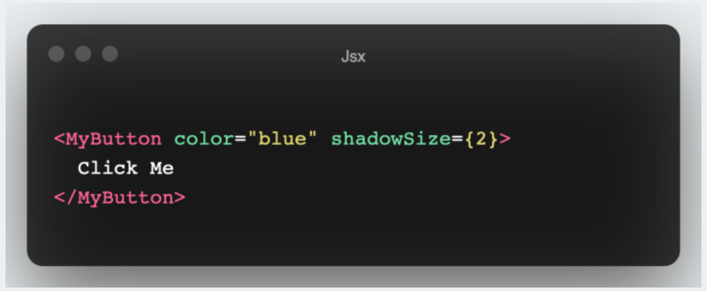
=

https://codepen.io/chadritchie/pen/ykciH?editors=0010

What does this have to do with React.js?

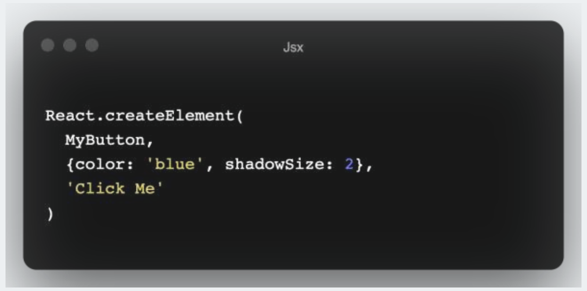
React is handling the component tree and we have different renders like ReactDom and React-Native

Now we have a new render called React-Three-Fiber and it is converts a React tree into underlying three.js calls. Just like we have in React Jsx



Jsx

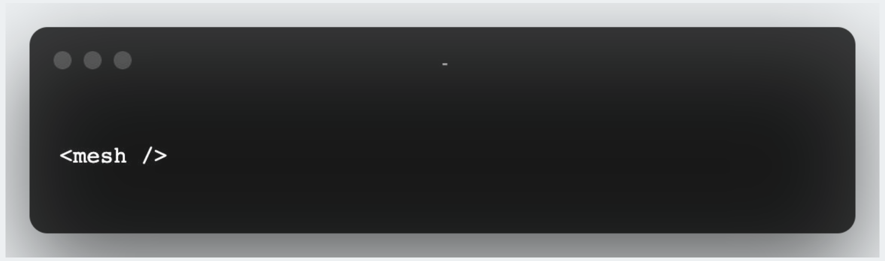
```
<MyButton color="blue" shadowSize={2}>
  Click Me
</MyButton>
```



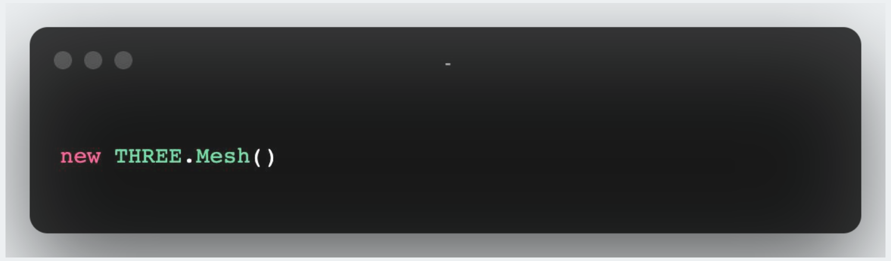
Jsx

```
React.createElement(
  MyButton,
  {color: 'blue', shadowSize: 2},
  'Click Me'
)
```

React-Three-Fiber does the same but generates three.js primitives instead



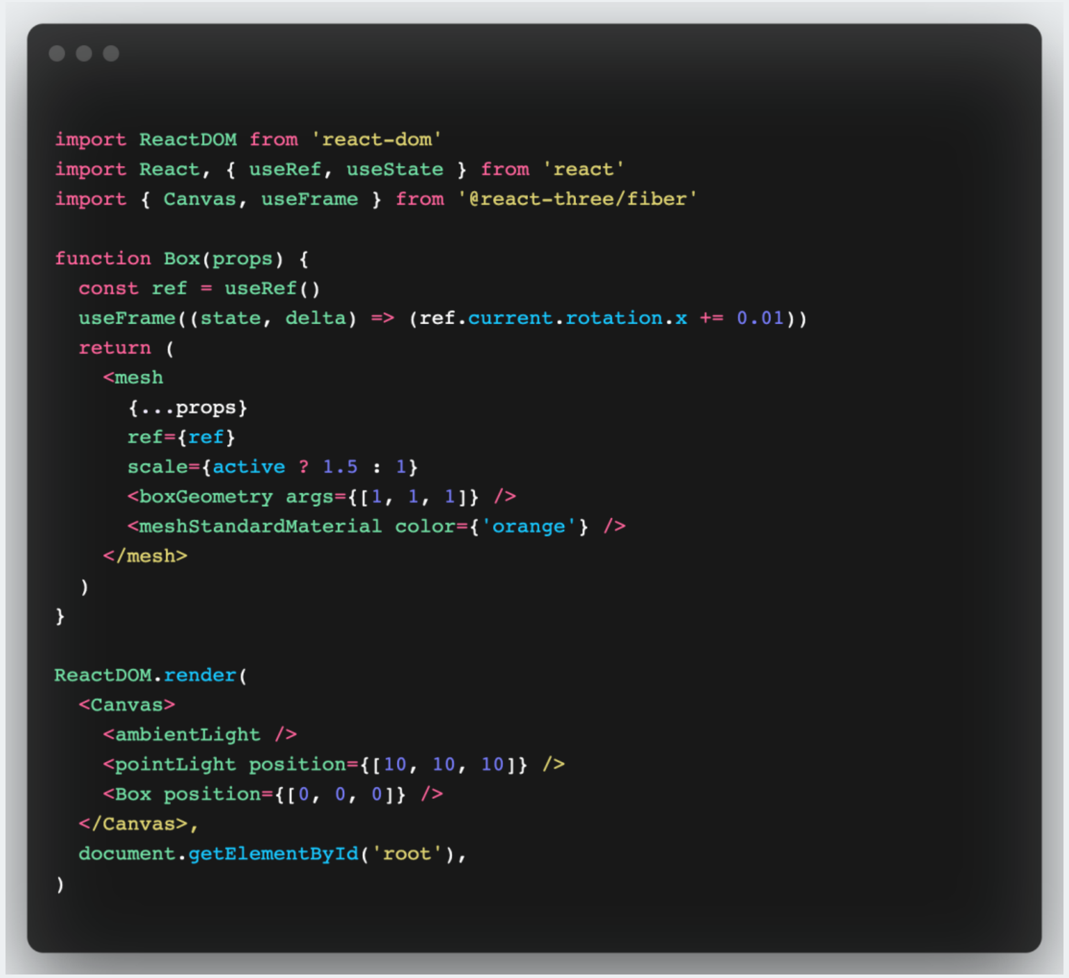
```
<mesh />
```



```
new THREE.Mesh()
```

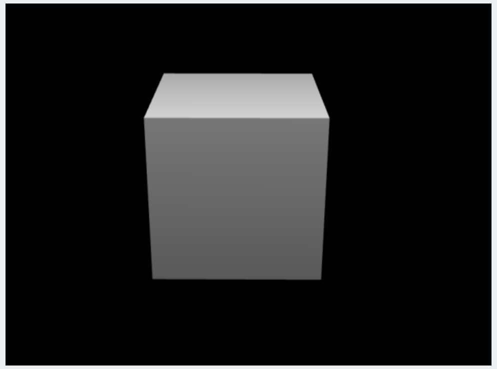
React Three Fiber

Now let's us create the same box writen in React.Js with React-three-fiber



```
import ReactDOM from 'react-dom'
import React, { useRef, useState } from 'react'
import { Canvas, useFrame } from '@react-three/fiber'

function Box(props) {
  const ref = useRef()
  useFrame((state, delta) => (ref.current.rotation.x += 0.01))
  return (
    <mesh
      {...props}
      ref={ref}
      scale={active ? 1.5 : 1}
      <boxGeometry args={[1, 1, 1]} />
      <meshStandardMaterial color={'orange'} />
    </mesh>
  )
}

ReactDOM.render(
  <Canvas>
    <ambientLight />
    <pointLight position={[10, 10, 10]} />
    <Box position={[0, 0, 0]} />
  </Canvas>,
  document.getElementById('root'),
)
```

=



=

DEMO

More information on React-Three-Fiber

- Package to checkout:
  - @react-three/gltfjsx – turns GLTFs into JSX components
  - @react-three/drei – useful helpers for react-three-fiber
  - @react-three/postprocessing – post-processing effects
  - @react-three/flex – flexbox for react-three-fiber
  - @react-three/xr – VR/AR controllers and events
  - @react-three/cannon – physics based hooks
  - zustand – state management
  - react-spring – a spring-physics-based animation library
  - react-use-gesture – mouse/touch gestures
  - leva – create GUI controls in seconds

- Demo repo: https://github.com/vLX42/react-three-fiber-demo
- Demo link: https://react-three-fiber-demo.vercel.app/