3D-fying Your Website

Using React Three Fiber (R3F)

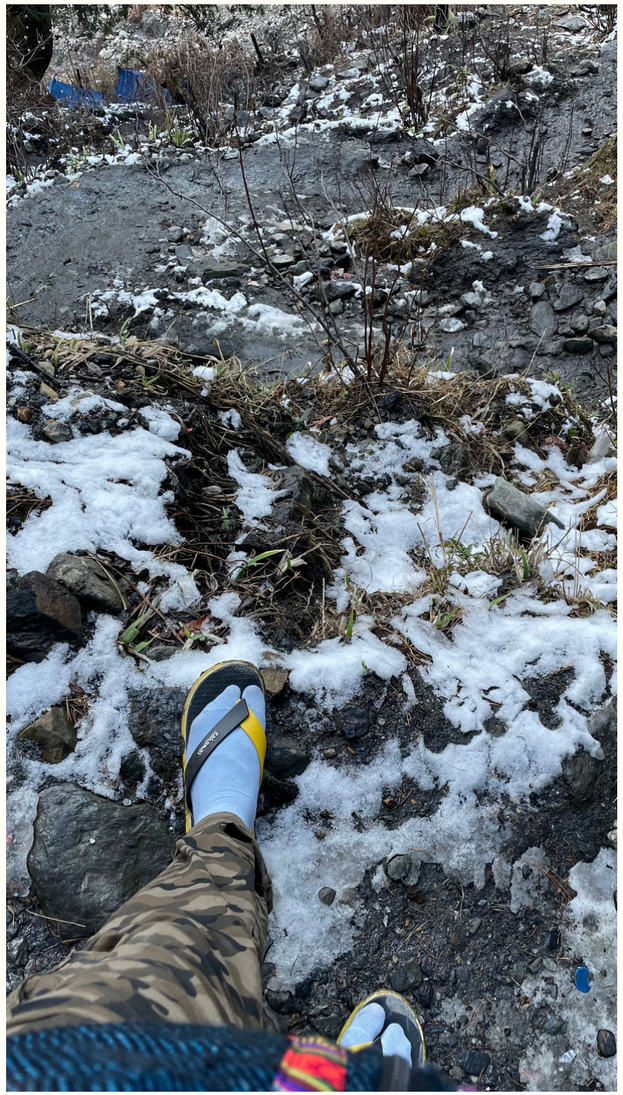
Alok Kumar

Hey, I'm **Alok**, your friendly neighborhood fullStack developer From UtkalLabs. I'm all about JavaScript - coding, blogging, Writing ebooks, and talking at events. When I'm not in the tech world, you can catch me watching anime, writing poems or jamming to music.

but I'm not just a tech geek. I love cooking up a storm in the kitchen and going on adventures.

Fun fact About me: I once trekked through snow & mud in just slippers to Grahan village situated at 7,700 ft.

I’m not lying...



...

Contents

01 Intro to Three.js & R3F

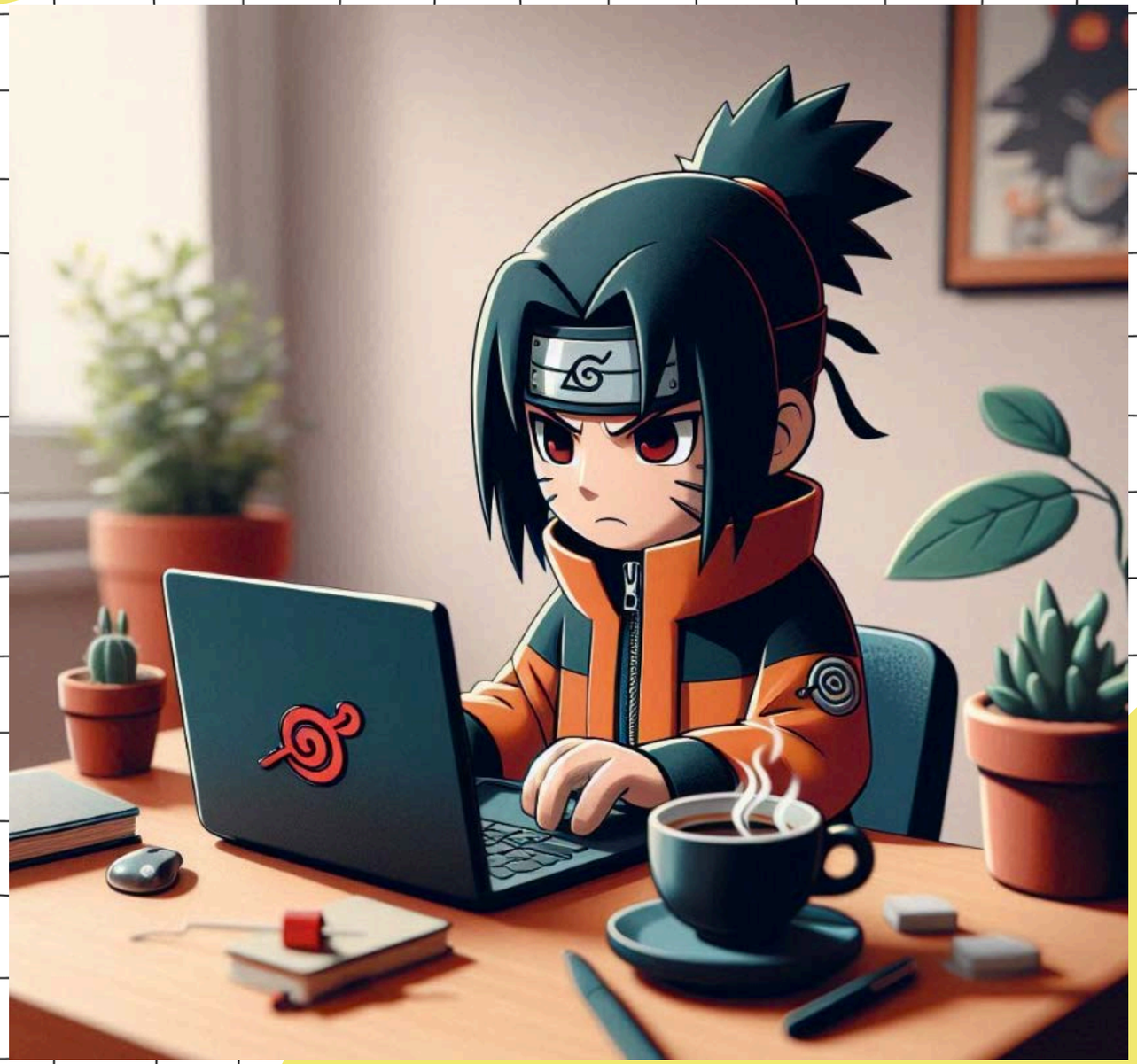
02 Getting Started & Setting Up

03 Creating 3D Scenes

04 Environment and Staging

05 Loading Models

06 Deployment



Introduction to Three.js and React Three Fiber (R3F)

webGL

- A JavaScript API for rendering 2D and 3D graphics within any compatible web browser
- Provides low-level access to the GPU, enabling high-performance rendering of 3D graphics
- Basis for Libraries like Three.js

threeJS

- Abstracts the complexities of WebGL with a more accessible API.
- Includes scene graph, built-in geometries, materials, textures, lighting, and animations.
- Works seamlessly in all modern web browsers.
- Extensive documentation, examples, and a large support network.

Examples

- https://eyes.nasa.gov/apps/solar-system/#/home
- https://neal.fun/design-the-next-iphone/
- https://aloks-portfolio.netlify.app/

R3F

- Combines React's declarative syntax with Three.js's 3D capabilities.
- Uses JSX and React hooks to manage 3D scenes.
- Leverages React components for reusable 3D objects and logic.
- Integrates smoothly with existing React projects.

DREI

- A growing collection of useful helpers and fully functional, ready-made abstractions for @react-three/fiber.

Getting Started and Setting Up

Setting Up Vite Project

- Installing Vite and initializing a new project
- Integrating React and R3F into the Vite setup

Basic Components

- A scene that will contain objects
- An object
- A camera
- A renderer

Basic Components

Scene:

- The container for all 3D objects.

Geometry:

- Defines the shape of 3D objects.

Material:

- Defines the appearance (color, texture) of 3D objects.

Mesh:

- Combines geometry and material to create a visible 3D object.

Example

...

Link

Creating 3D Scenes

Native threeJS

...

```
<body>
    <canvas class="webgl"></canvas>
</body>
```

Native threeJS

```
// Canvas
const canvas = document.querySelector("canvas.webgl");
// Scene
const scene = new THREE.Scene();
// Object
const geometry = new THREE.BoxGeometry(1, 1, 1);
const material = new THREE.MeshBasicMaterial({ color: "red" });
const mesh = new THREE.Mesh(geometry, material);
scene.add(mesh);
```

Native threeJS

```
// Camera
const camera = new THREE.PerspectiveCamera(75, sizes.width /
sizes.height);
scene.add(camera);

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

renderer.render(scene, camera);
```

R3F

```
<Canvas>
  <mesh >
    <torusKnotGeometry />
    <meshBasicMaterial />
  </mesh>
</Canvas>
```

Axes Helper

The primary purpose of AxesHelper is to provide a visual reference for the X, Y, and Z axes in your 3D scene.

OrbitControls

Purpose:

- Enables user interaction with the 3D scene.
- Enable users to rotate, zoom, and pan the camera around the 3D scene.

Transformations

Positioning:

- Changing an object's location in 3D space.

Rotating:

- Adjusting an object's orientation around its axes.

Scaling:

- Modifying an object's size along the X, Y, and Z axes.

Animations

Purpose:

- Executes code on every frame render.

Basic Syntax:

- Importing and using **useFrame**.

Animating Objects:

- Changing properties like position, rotation, and scale.

Environment and Staging

Lighting

Purpose:

- Adds realism and depth to 3D scenes.

Types of Lights:

- Ambient, Directional, Point, and Spotlights.

Properties:

- Color, intensity, and position.

Shadow Configuration:

- Enabling and configuring shadows for realism.

Types

Ambient Light:

- Provides a general illumination that affects all objects equally.

Directional Light:

- Simulates sunlight or other distant light sources.

Point Light:

- Emits light in all directions from a single point, similar to a light bulb.

Spotlight:

- Emits a cone of light in a specific direction, similar to a flashlight.

Shadows

- Shadows add depth and realism to 3D scenes by simulating the way light interacts with objects.
- They help viewers understand the position and scale of objects relative to each other and their environment.
- Balancing shadow quality and performance is crucial.

Background

Purpose:

- Sets the visual backdrop for the 3D scene.

Types of Backgrounds:

- Solid color, gradient, image, and environment maps.

Models

Models

- Incorporating complex 3D assets into your scene.
- Models can represent anything from characters and vehicles to entire environments.
- Supported Formats: common formats like GLTF/GLB, OBJ, FBX.

Deployment

Deployment

Build Process:

- Optimizing and building the project.

Deployment Platforms:

- Vercel, Netlify, GitHub Pages, etc.

Thank you

@thecoollearner