|  |
| --- |
| React Three Fiber  training project  By Wawa Sensei mesh portal material [Watch on youtube](https://www.youtube.com/watch?v=2W_VR92Pqgs&list=PLpepLKamtPjiUF6PvVUbIFhx9HaS0qJs_) |

A screenshot of a black screen

Description automatically generated

# Overview

## AFTER WATCHING THE VIDEO

A screenshot of a computer screen

Description automatically generatedThe result will be:

Sensei use two open source for background and model, which are:

* [Blockade Labs Skybox - AI-Generated 3D Worlds](https://skybox.blockadelabs.com/) for the Background
* [Quaternius • Ultimate Monsters](https://quaternius.com/packs/ultimatemonsters.html) for the Monsters model

Conclude of 10 parts of progressing

* 1: BlockadeLabs worlds
* 2: Pocket Monsters
* 3: MeshPortalMaterial
* 4: Refacto
* 5: Text
* 6: Enter into worlds
* 7: Camera Controls
* 8: Hover monster animation
* 9: useCursor
* 10: Final result

This is an application using useTextures function and the BackFace, GLTFJSX, useAnimation, useCursor,…

Now let’s start

# DETAIL THE WORK

## BLOCKADELABS WORLDS

Access the link of 3D Texture Background Generator and download like 3 of them for 3 kind of themes such as fire, water, nature world.

Just access to [Blockade Labs Skybox - AI-Generated 3D Worlds](https://skybox.blockadelabs.com/) and

* Fill in the prompt the description of background which you want to see.
* Select style.
* Press ‘Generate’.

For the time-saving, I will download all the assets that needs for this project (Textures, Font)

A screenshot of a computer program

Description automatically generated

Initialize project with some Basic R3F components:

* OrbitControl
* ambientLight
* <mesh> and any object you can feel comfortable inside with material.

src/App.js

A computer screen with text

Description automatically generated

src/index.js

A screen shot of a computer program

Description automatically generated

Starter pack is done:

A colorful circle with a white background

Description automatically generated

Now you want to test the texture apply on that sphere using meshBasicMaterial map with useTexture function variables.

Then you will want to add Environment preset=”sunset” to get nice light theme.

A screen shot of a computer program

Description automatically generated

A beach with rocks and water

Description automatically generated with medium confidence

The face will be always convex to our view. So we just need Backface.

Also Increasing segment width and height makes texture more smooth.

Note: Backface means when you drag the texture to the left, the texture move to the right.

A screen shot of a computer program

Description automatically generated

## POCKET MONSTERS

Go to [Quaternius • Ultimate Monsters](https://quaternius.com/packs/ultimatemonsters.html) to download some free 3D models that we gonna use in this example.

My choice:

A screenshot of a computer

Description automatically generated

Use command of GLTFJSX to convert .gltf file to jsx file.

* npx gltfjsx public/models/<GLTF file name> -o “public/models/<JSX output file name>” -r “<root folder path which GLTF files will be served>”

A screenshot of a computer program

Description automatically generated

⚠️The different with and without -r: It will correct the path that link to GTLF file.

* With -r “public”

A screen shot of a computer code

Description automatically generated

* Without -r “public”

A screen shot of a computer screen

Description automatically generated

Then we rename “Model” to any names you like and set animation for it before adding to the scene.

⚠️Can use useEffect once to console.log the actions to pick action names

A screen shot of a computer code

Description automatically generated

Final add the mesh to the scene, also adjust the position of it.

A screen shot of a computer program

Description automatically generated

Cartoon cat on a beach

Description automatically generated

## MESHPORTALMATERIAL

You can find document of MeshPortalMaterial at the offical Drei Github.

The only rule for this mesh to work properly is:

* There is a mesh acting as children of MeshPortalMaterial

Example:

A screenshot of a computer code

Description automatically generated

The above mesh, we will call it Cat’s mesh

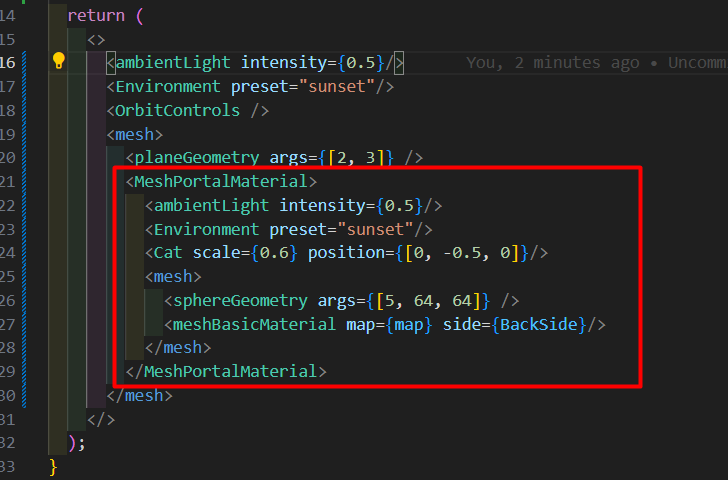
In this example, we want to make the portal one as plane. So we add <planeGeometry/> and <MeshPortalMaterial> into a mesh.

Order of the mesh will be

<mesh>

* <planeGeometry />
* <MeshPortalMaterial>
  + Cat’s mesh

⚠️ Put light in the nested <mesh> too or we won’t see anything.



And the result

A cartoon cat on a beach

Description automatically generated

⚠️ The limit of the <mesh> is that you can’t turn the plane arround to see their ‘but’ 😁

So we set the MeshPortalMaterial a side={THREE.DoubleSide}

A screen shot of a computer program

Description automatically generated

This THREE is from

Import \* as THREE from ‘three.js’

A screen shot of a computer code

Description automatically generated

The result is:

A cartoon of a fish

Description automatically generated

⚠️If we look from the side, the plane is plat so we can’t see a thing when plane align with our view 90-degree angle.

So solution to fix that is replace the plane with the Rounded Box where you can set the depth to see from the side angle

A screen shot of a computer program

Description automatically generated

|  |  |
| --- | --- |
| Before (~90 degrees)  A letter i with a blue sky  Description automatically generated | After |

## REFACTO

Because we gonna make 2 more small mesh like what we are doing.

So we need a Component to generate more easily, called MonsterStage

MonsterStage will return the <group {…props}> of all Cat’s mesh, the Cat and the texture will all be variables.

A screen shot of a computer

Description automatically generated

Generate two more with 2nd , 3rd mesh with different model and texture with some small adjust about position and rotation

A computer screen shot of text

Description automatically generated

Remember to set idle animation for 2nd and 3rd model

## TEXT

Add the Text at the bottom of the card to show more description about the scene

First, we add the <Text> to MonsterStage function

Then:

* We set font, fontSize, position
* We set anchorY to align the bottom of box.

A screen shot of a computer program

Description automatically generated

But Why z-position is 0.051 and anchorY is bottom ?

The z position of RoundedBox is 0.1 and if we set 0.05 z for the <Text>, it will conflict with the front and back face. So +0.001 to make sure, it’s on front face.

The anchorY “bottom” will move slightly the Text toward the Y position to 0 point.

The result:

A screenshot of a video game

Description automatically generated

Now, to make the text to stand out, we need to set color and name, but as variable like this

A computer code on a black background

Description automatically generated

A screen shot of a computer program

Description automatically generated

The result:

A group of cartoon characters

Description automatically generated

## ENTER INTO WORLDS

We need a flag to show that which section is active right now.

Declare

* Const [active, setActive] = useState(‘’);

And we put active and setActive into properties of MonsterStage

A screen shot of a computer code

Description automatically generated

On the RoundedBox, we set event onDoubleClick to setActive scene name.

When the active is correct with the name, MeshPortalMaterial will blend with the global background by set blend = 1

And when the name is reactive, that mean the Mesh will close.

A screen shot of a computer program

Description automatically generated

To make the blend become smoother, we useRef and useFrame on the <MeshPortalMaterial>

* Install maath to calculate transition to blend the background.
* useFrame with \_state, delta to help the calculate function more accurate
* Combine with easing.damp(<object>, <prop>, <target>, <delta>)

A computer screen with text and red arrow

Description automatically generated

⚠️Research this maath library

Result: smoother animation

A cactus in a field

Description automatically generated

## CAMERA CONTROLS

Now we set the camera to center the scene, we useRef on <CameraControls>.

* Declare a Ref on Camera
* useEffect whenever active is changed.
* Set the position of the target to the camera to look at.
* useThree to select the component inside a scene
* scene.getObjectByName(active).getWorldPosition(targetPosition); to pass the active roundedBox position to targetPosition
* Set the camera control: the position of target camera looks at, the position of camera.

A screen shot of a computer program

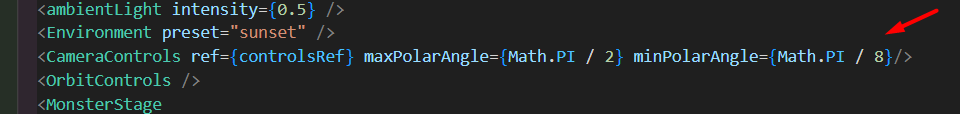
Description automatically generated

A screen shot of a computer code

Description automatically generated

⚠️Remember to remove Orbitcontrols for the CameraControls to rotate.

To fix the user experience when they can keep rotating to max out by set maxPolarAngle and minPolarAngle



## HOVER MONSTER ANIMATION

A computer code with white text

Description automatically generated with medium confidence

A screen shot of a computer code

Description automatically generated

Set hover to animate the model

A computer screen shot of a program code

Description automatically generated

## USECURSOR

To max out the UX

## 10. FINAL RESULT