# LIGHTS

# Adding Lights

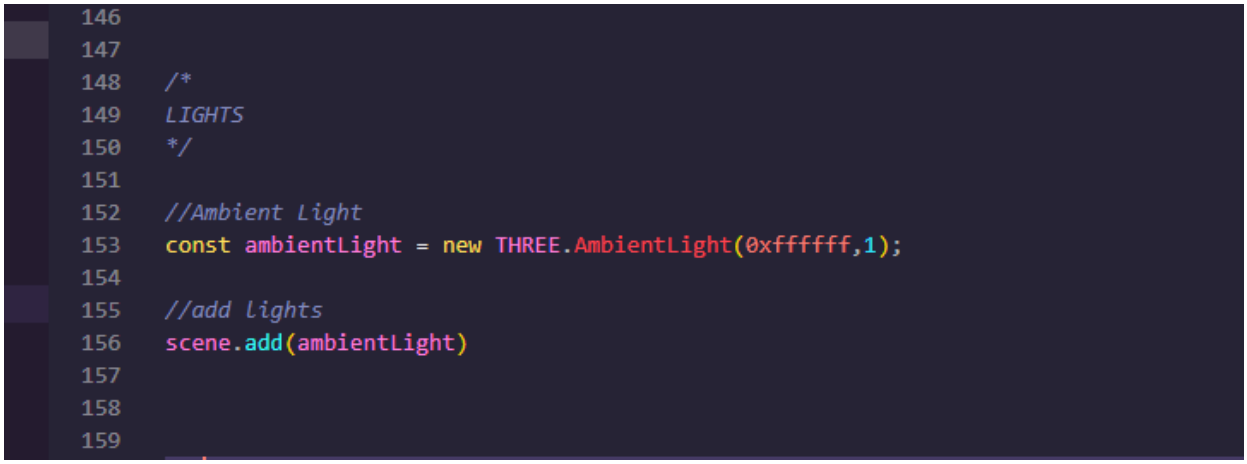
-set a variable to new THREE.LightType(parameters) then add to scene.

```
//Ambient Light
const ambientLight = new THREE.AmbientLight(0xffffff,1);
scene.add(ambientLight)
```

## Ambient Lights

# Const x = new THREE.AmbientLight(color,intensity,)

-this light comes from everywhere, and is uniform.



```
146
147
148   /*
149   LIGHTS
150   */
151
152   //Ambient Light
153   const ambientLight = new THREE.AmbientLight(0xffffff,1);
154
155   //add lights
156   scene.add(ambientLight)
157
158
159
```

-you can also instantiate with dot notation. example : **ambientLight.color = new THREE.Color(0xffffff)**

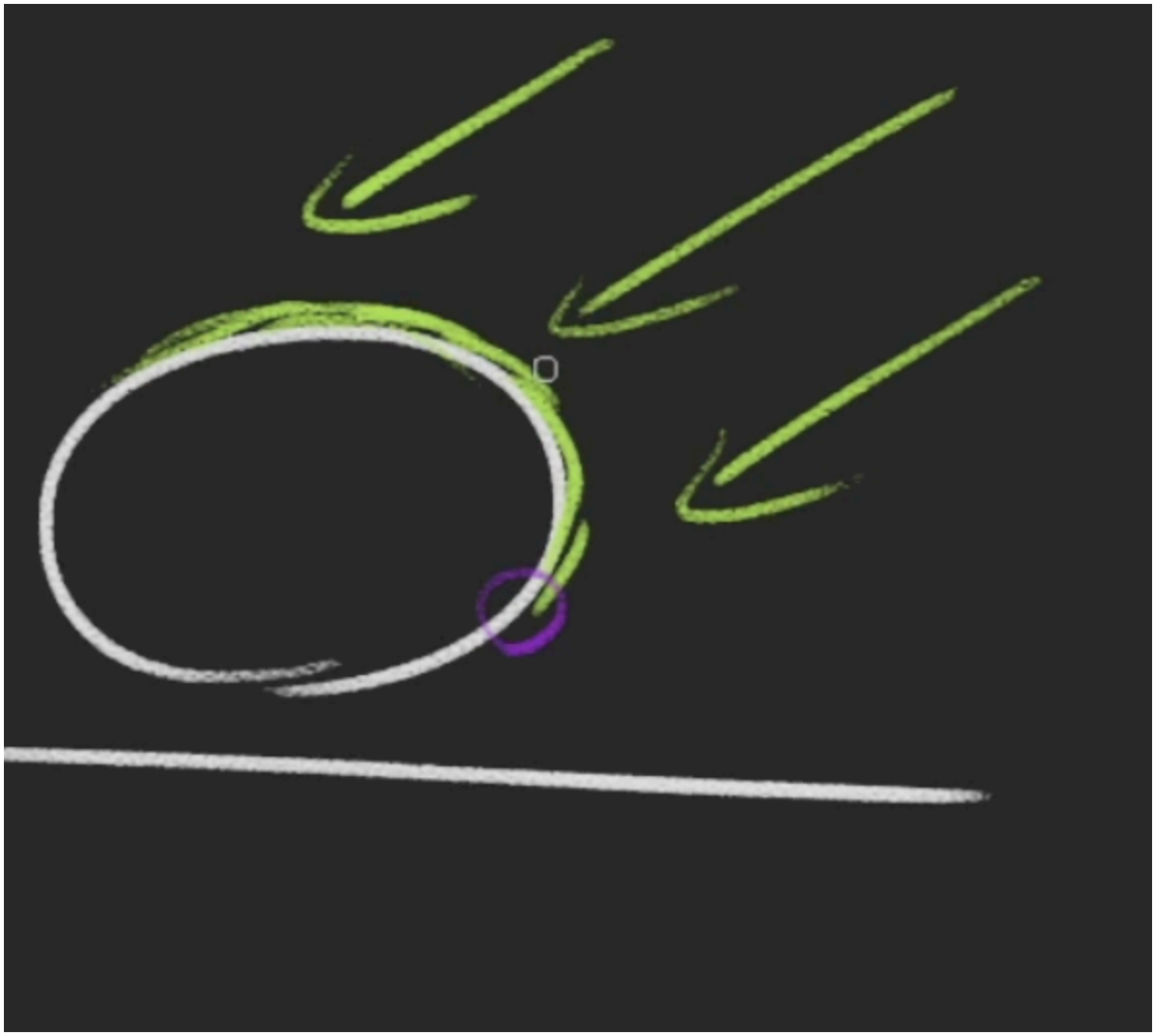
-lights every spot the same, even weird shapes.

-becasue of this **it can be used as dim lighting to hit behind things, to simulate light bouncing.**

# Directional Lights

## Const x = new THREE.DirectionalLight(color, intensity)

**Have a sun-like effect as if the sun rays were traveling in parallel.**



-Starts off on top first, but you can change direction of the light

-**directionalLightObjectName.posistion.set(x, y, z)**

# Hemisphere Light

**Const x = new THREE.HemisphereLight(color(skycolor), groundColor, intensity)**

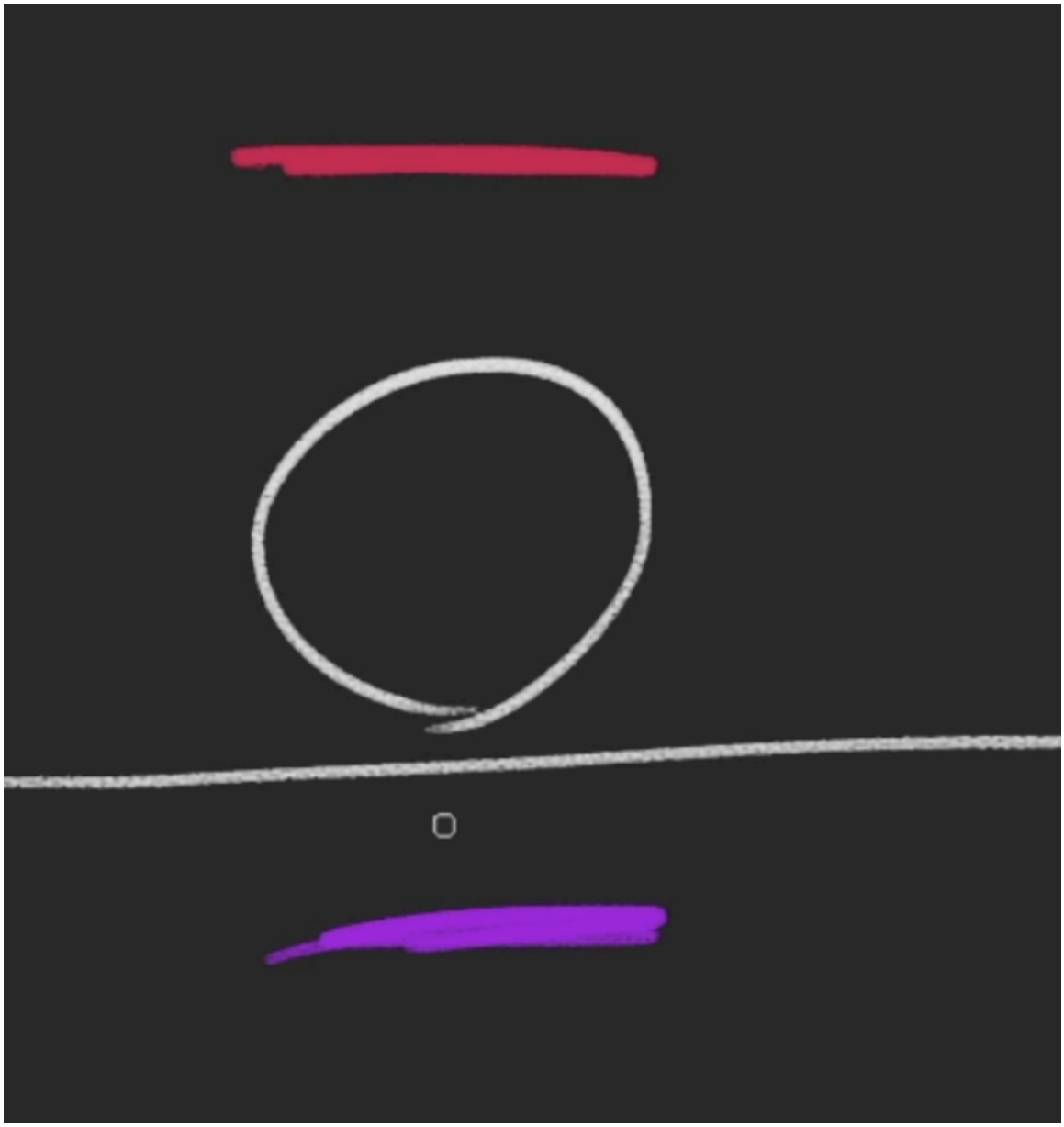
**-almost like a gradient**

```
//hemisphere light
const hemisphereLight = new THREE.HemisphereLight();
scene.add(hemisphereLight);
hemisphereLight.intensity = 1;
hemisphereLight.color = new THREE.Color(0xff0000);
hemisphereLight.groundColor = new THREE.Color(0x0000ff);

/**
```

**Or**

```
//hemisphere light
const hemisphereLight = new THREE.HemisphereLight(0xff0000, 0x0000ff, 1);
```



# Point Lights

**-new THREE.PointLight(color, intensity, distance, decay)**

-kind of like a lighter

-a pointer light. Starts at dead center of camera. You can offset it like this:

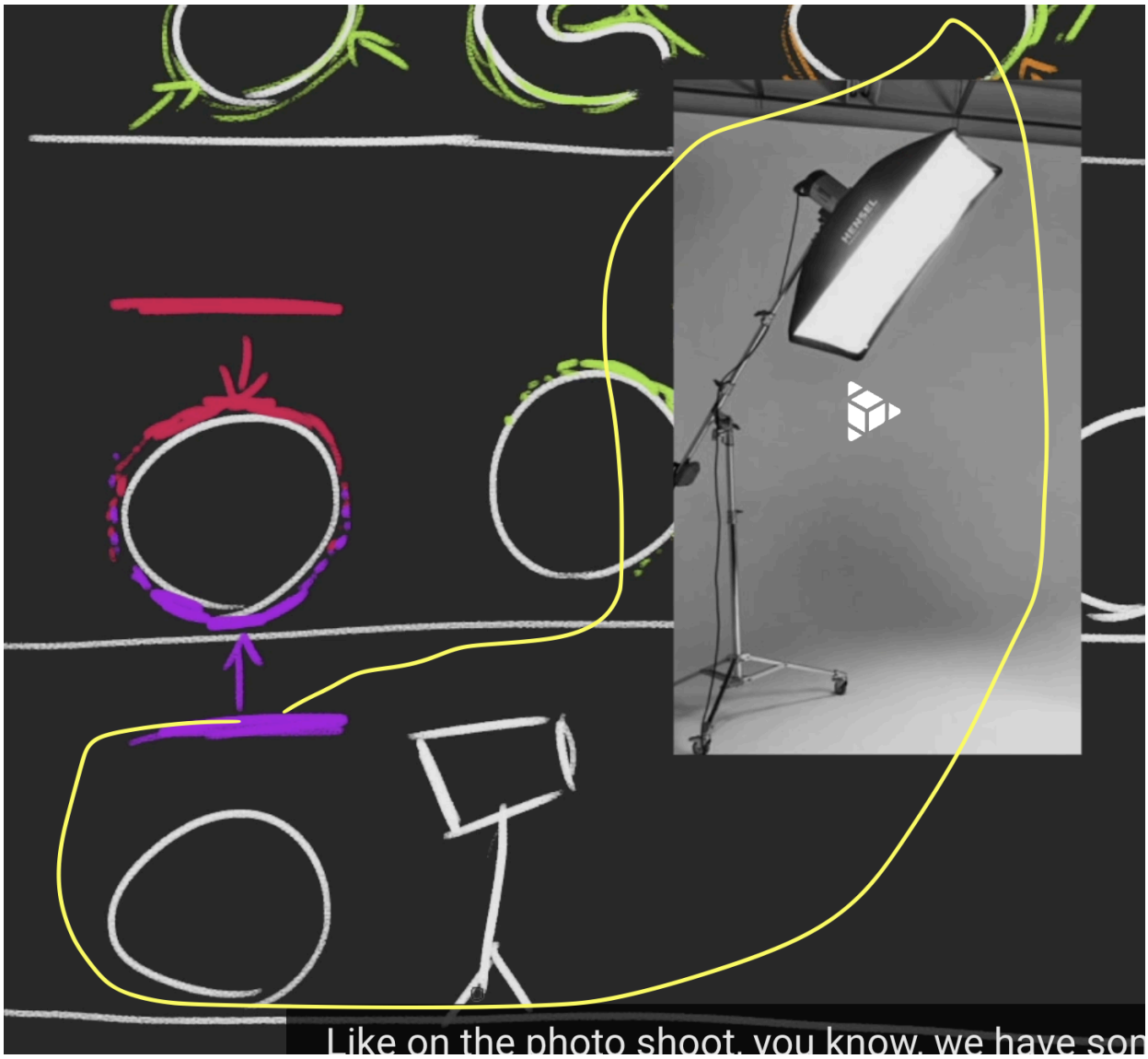
```
161   //Ambient Light
162   const ambientLight = new THREE.AmbientLight(0xffffff,1);
163
164   //point light
165   const pointLight = new THREE.PointLight(0xffffff, 30)
166
167   pointLight.position.x = 2
168   pointLight.position.y = 3
169   pointLight.position.z = 4
170
171   //add lights
172   scene.add(ambientLight, pointLight)
173
174
```

- Other properties **.distance and .decay**
- **.distance** is the distance the light goes
- **.decay** is the fade of the light. Kind of like a feather

# Rect Area Light

Const x = new THREE.RectAreaLight(color, intensity, width, height)

**A mix between directional and diffuse light**




**Only world with MeshStandardMaterial and MeshPhsicalMaterial**

**.LookAt() method** is great for this light. You can do it after you position it, and it will have the light look at what you put in the ().

-For example we can put a vetor3 in the () for **.LookAt(new THREE.Vector3())** and since we didnt say where the vector is, it will default to 0, 0, 0 which is the center of the screen.

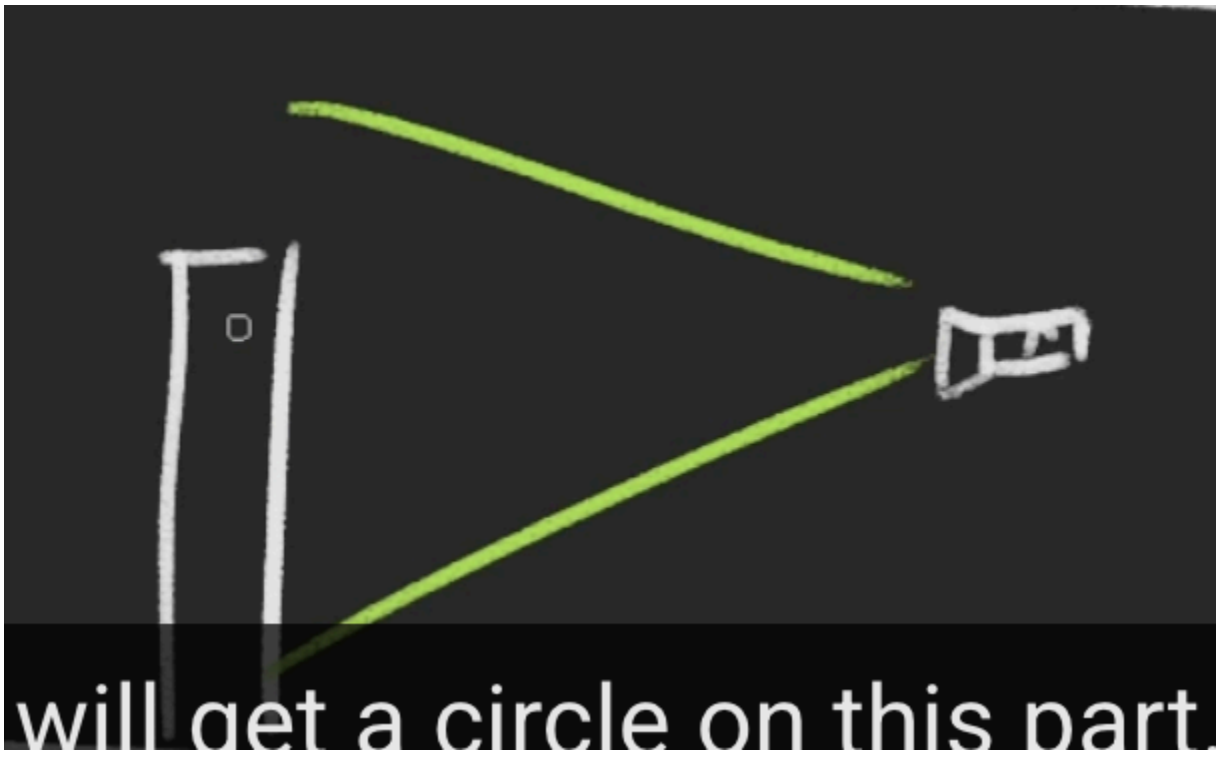
# Spotlight

**Const x = new THREE.SpotLight(color, intensity,distance, angle, penumbra, decay)**

**Example:**
```
const spotLight = new THREE.SpotLight(0x78ff00, 4.5, 10, Math.PI * 0.1, 0.25, 1)
```

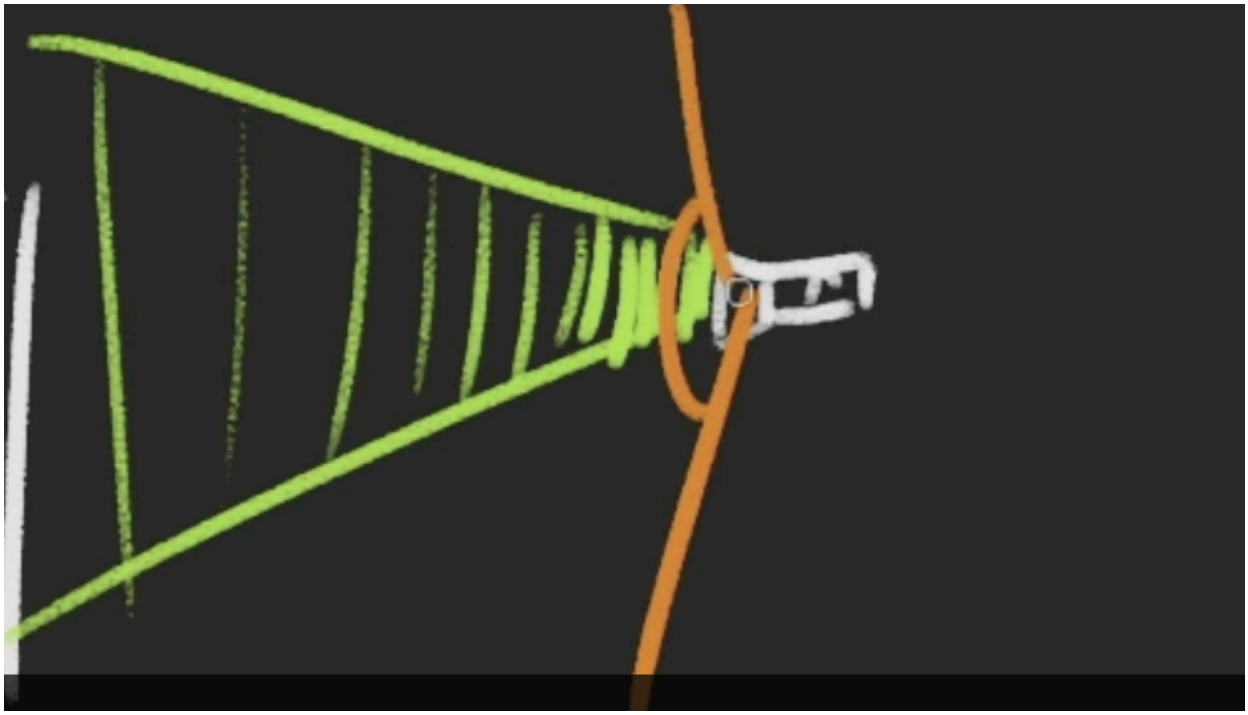
-perameters: color, intensity, distance, angle, penumbra, decay

-a cone of light



-**.distance** kind of like a fade. Further the distance the less powerful

-**.angle -** the angle that the light starts off ass.



-**.penumbra** smooths out the light, so the circle isnt as shar. Blurs the circles edge

**HAVE TO ROTATE DIFFERENTLY**

**Spotlight.target.posistion.x = 2**

**scene.add(spotlight.target)**

**YOU HAVE TO ADD THE SPOTLIGHT.TARGET TO THE SCENE, THEN USE THAT TO MOVE SPOTLIGHT**

# Performances

-Lights cost a lot in terms of performance

-50 is the light limit, and it would be too many

-have as few lights as possible

## Minimal Cost Lights

-Ambient Lights

-Hemisphere lights

## Moderate Cost

-Directional Light

-Point Light

## High Cost

-spot light

-react light

# Baking

-Baking the light into the texture, so you do not have to use the actual light

-downside is we cannot move the light anymore, since there is no light, and we have to load bigger textures

-better to do this in a 3D texture

# Helpers

-used to assist us with positioning the lights.

-shows where the light is

**HemisphereLightHelper**

-new THREE.HemiSphereLightHelper(lightHere,size)

**DirectionalLightHelper**

```
const directionalLightHelper = new THREE.DirectionalLightHelper(
  directionalLight,
  0.2
);
scene.add(directionalLightHelper);
```

**PointLightHelper**

-new THREE.PointLightHelper(variable, size)

**RectAreaLightHelper**

-this one you have to Import

```
import { RectAreaLightHelper } from "three/examples/jsm/Addons.js";
```

```
const rectLightHelper = new RectAreaLightHelper(reactLight);
scene.add(rectLightHelper);
```

**SpotLightHelper**

-new THREE.SpotLightHelper(variable)

# Toggle Off and On

-use **.visible** and set to false to turn off.

-you can then add to gui and have it toggle off and on:

```
const directionalLightHelper = new THREE.DirectionalLightHelper(
  directionalLight,
  0.2
);
scene.add(directionalLightHelper);
directionalLightHelper.visible = false;
directionalTweaks.add(directionalLightHelper, "visible");
/**
```