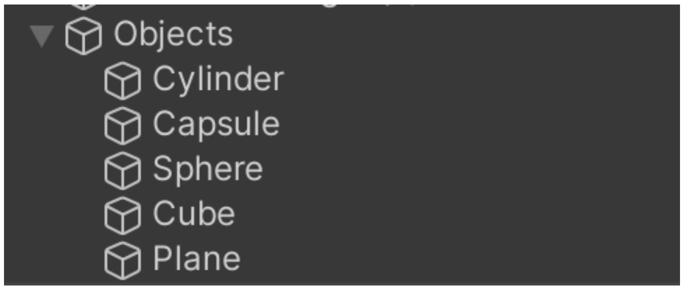
CSE462/562 – Augmented Reality (Fall 2022) Homework #4

Ahmet Tuğkan Ayhan
1901042692

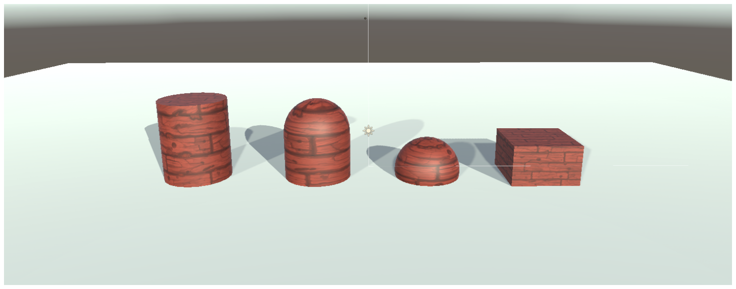
Programming Part

Objects

Except the plane object, every object is created with same material.

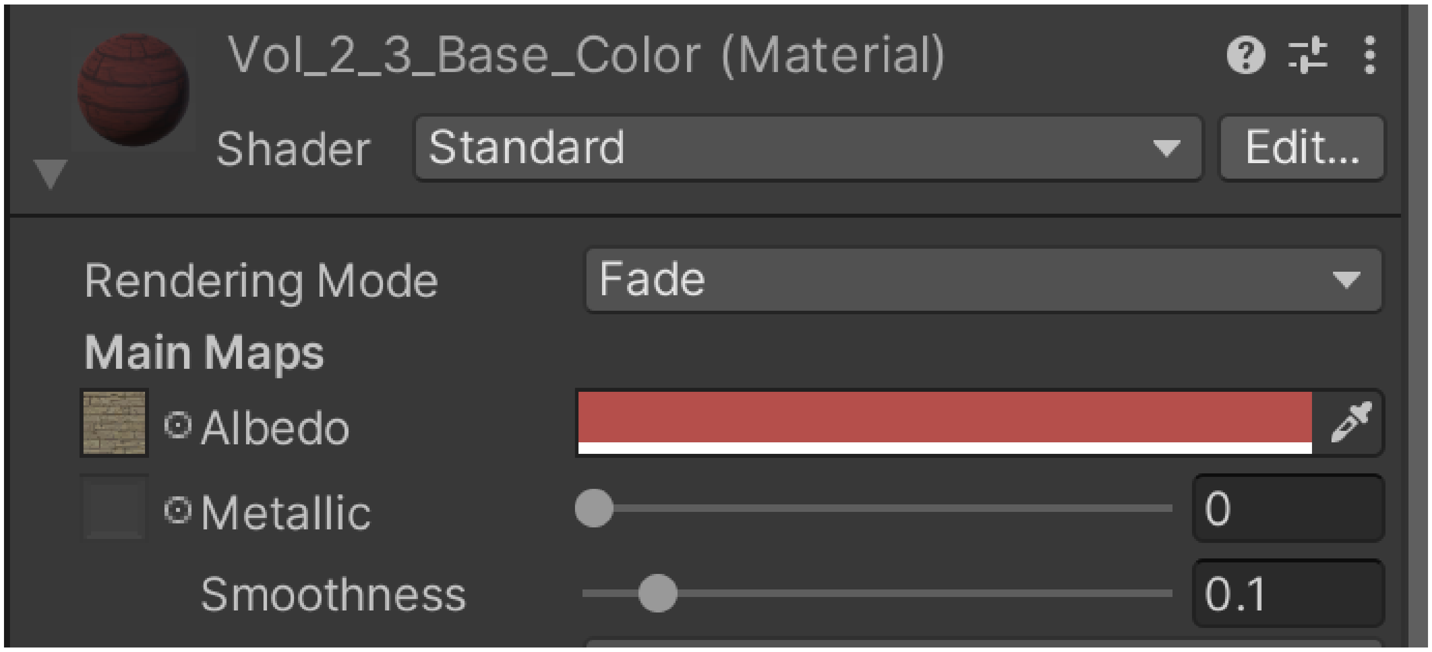


Objects
Cylinder
Capsule
Sphere
Cube
Plane



Material

For the material, I used LowlyPoly Stylized Wood Texture. Then I decreased metallic and smoothness values to achieve Limbertian Material look as much as possible.



Vol_2_3_Base_Color (Material)

Shader: Standard

Rendering Mode: Fade

Main Maps

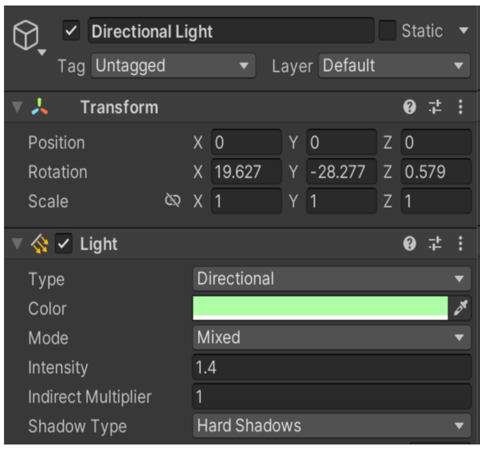
Albedo

Metallic: 0

Smoothness: 0.1

Lights

There are 3 directional light positioned at different angles to make shades of the objects more visible



✓ Directional Light — Static
Tag Untagged — Layer Default

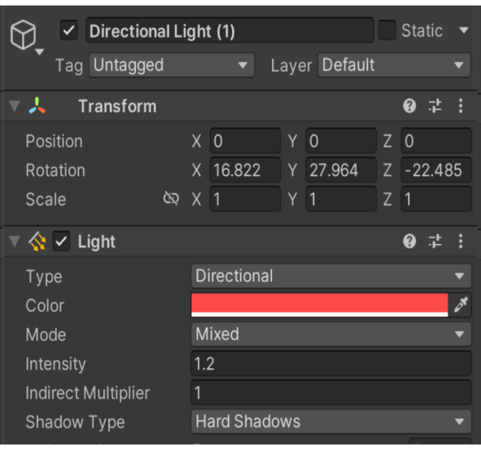
Transform

Position	X	0	Y	0	Z	0
Rotation	X	19.627	Y	-28.277	Z	0.579
Scale	X	1	Y	1	Z	1

✓ Light

Type	Directional
Color	🟩
Mode	Mixed
Intensity	1.4
Indirect Multiplier	1
Shadow Type	Hard Shadows

(Green Light)



✓ Directional Light (1) — Static
Tag Untagged — Layer Default

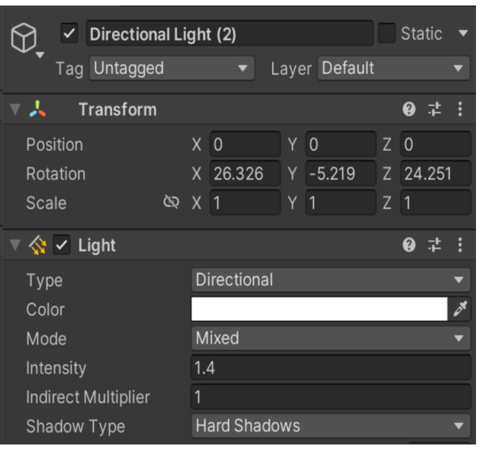
Transform

Position	X	0	Y	0	Z	0
Rotation	X	16.822	Y	27.964	Z	-22.485
Scale	X	1	Y	1	Z	1

✓ Light

Type	Directional
Color	🟥
Mode	Mixed
Intensity	1.2
Indirect Multiplier	1
Shadow Type	Hard Shadows

(Red Light)



✓ Directional Light (2) — Static
Tag Untagged — Layer Default

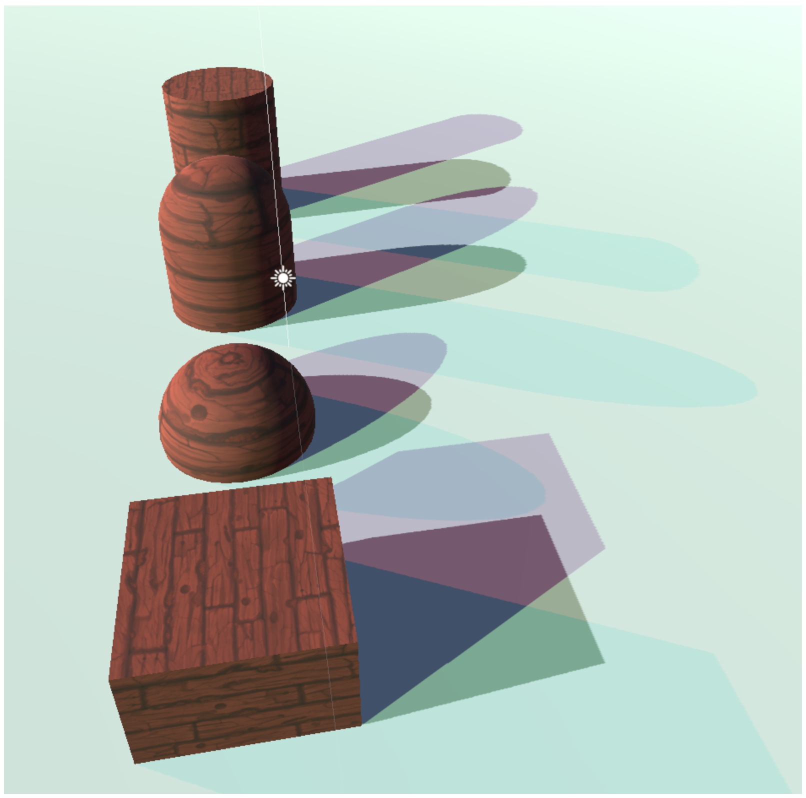
Transform

Position	X	0	Y	0	Z	0
Rotation	X	26.326	Y	-5.219	Z	24.251
Scale	X	1	Y	1	Z	1

✓ Light

Type	Directional
Color	⬜
Mode	Mixed
Intensity	1.4
Indirect Multiplier	1
Shadow Type	Hard Shadows

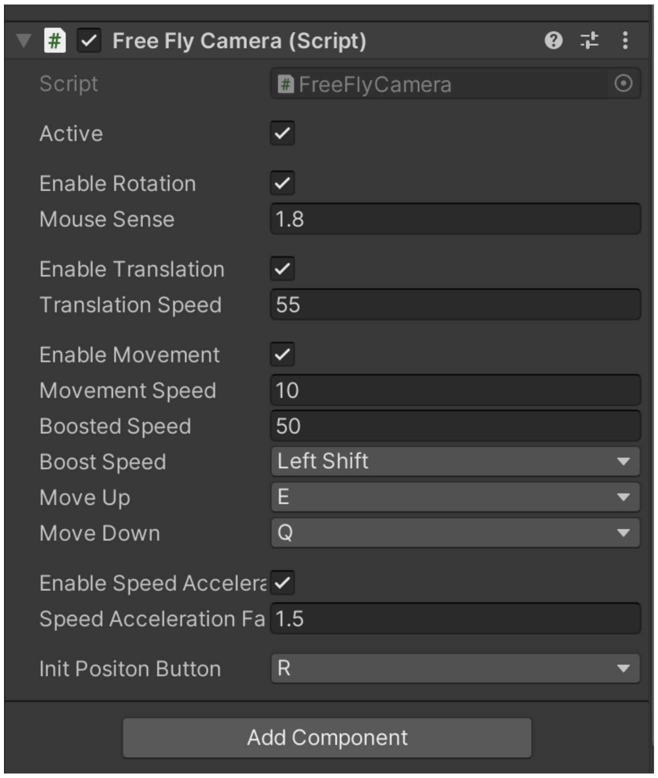
(Normal Light)



(Objects and their shadows)

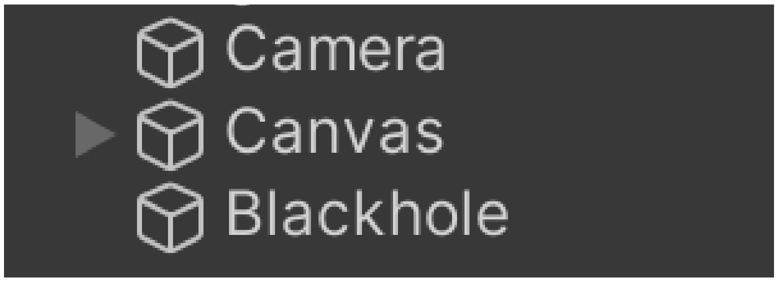
Camera

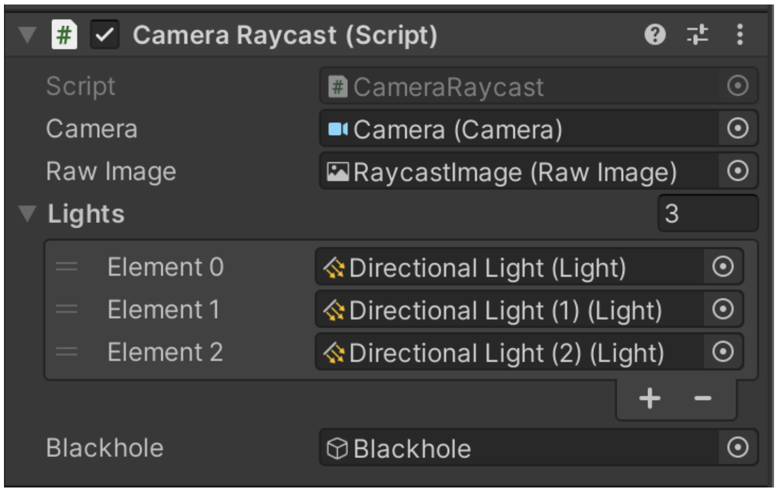
I added all of my scripts to camera object. To move the camera, I used an asset called Free Fly Camera. Then I added the script inside of asset to the Camera object.



Free Fly Camera (Script)

Script	FreeFlyCamera
Active	✓
Enable Rotation	✓
Mouse Sense	1.8
Enable Translation	✓
Translation Speed	55
Enable Movement	✓
Movement Speed	10
Boosted Speed	50
Boost Speed	Left Shift
Move Up	E
Move Down	Q
Enable Speed Accelera	✓
Speed Acceleration Fa	1.5
Init Positon Button	R





Camera Raycast (Script)

Script	CameraRaycast
Camera	Camera (Camera)
Raw Image	RaycastImage (Raw Image)
Lights	3
Element 0	Directional Light (Light)
Element 1	Directional Light (1) (Light)
Element 2	Directional Light (2) (Light)
Blackhole	Blackhole

Ray Casting (without blackhole)

The camera shoots 640x480 rays when the game starts. If the player presses the left mouse the rays refreshes and game shoots 640x480 rays from the camera again. But there are different mechanics between normal raycasting and raycasting with blackhole.

In normal raycasting, the game sends infinite length rays for each pixel. If the ray hits something, then the script identifies the color of the object which the ray hit and changes the color of the corresponding RawImage pixel.

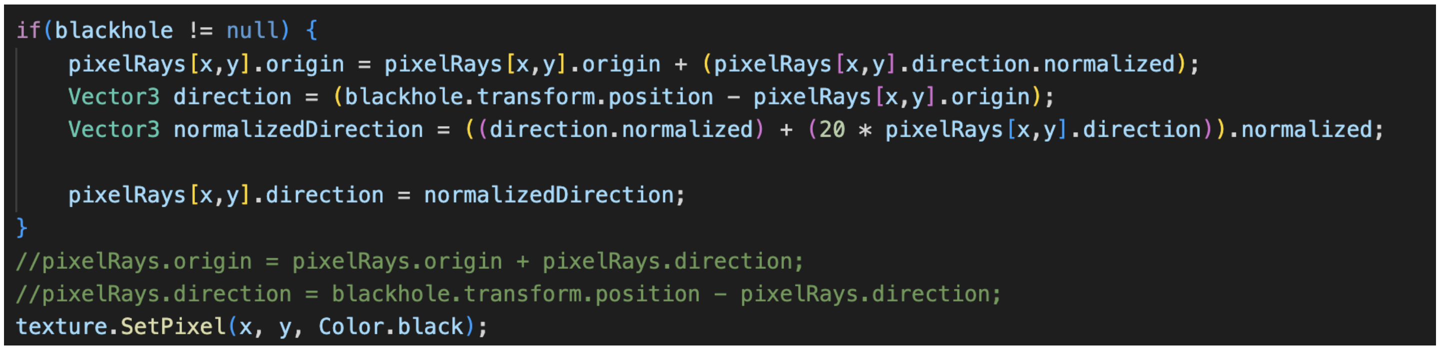
For the shadows, this time the game shoots rays from the location where previous rays hit, to the light source. If the rays hit something, then that means there is an object between the light source and the place where previous ray hit.

With every hit, the intensity of the shadow increases. To represent shadow, I used the code I pasted below:

```
pixelColor = Color.Lerp(pixelColor, Color.black, .3f);
```

Ray Casting (with blackhole)

When raycasting while there is a blackhole in the scene, instead of sending infinite length rays, I sent 1 unit rays and changed the rotation of the ray to the blackhole until it hits something.



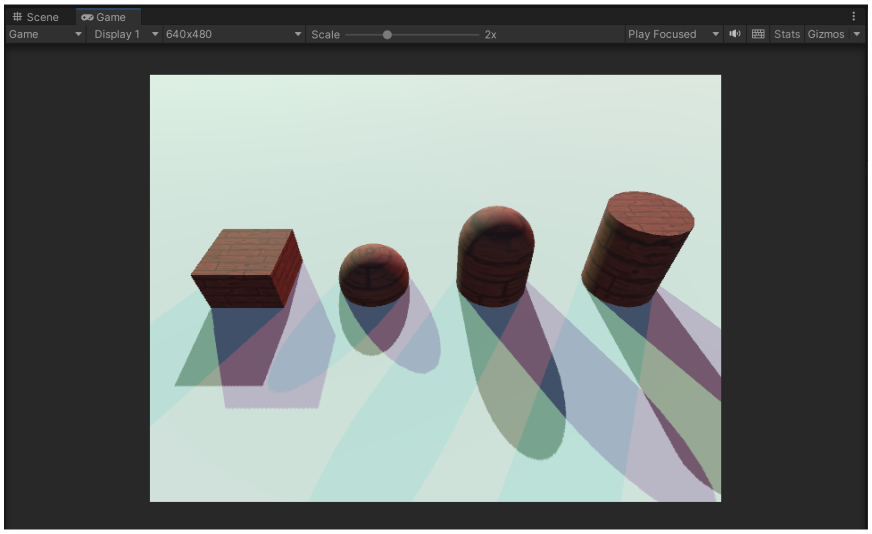
```
if(blackhole != null) {
    pixelRays[x,y].origin = pixelRays[x,y].origin + (pixelRays[x,y].direction.normalized);
    Vector3 direction = (blackhole.transform.position - pixelRays[x,y].origin);
    Vector3 normalizedDirection = ((direction.normalized) + (20 * pixelRays[x,y].direction)).normalized;

    pixelRays[x,y].direction = normalizedDirection;
}
//pixelRays.origin = pixelRays.origin + pixelRays.direction;
//pixelRays.direction = blackhole.transform.position - pixelRays.direction;
texture.SetPixel(x, y, Color.black);
```

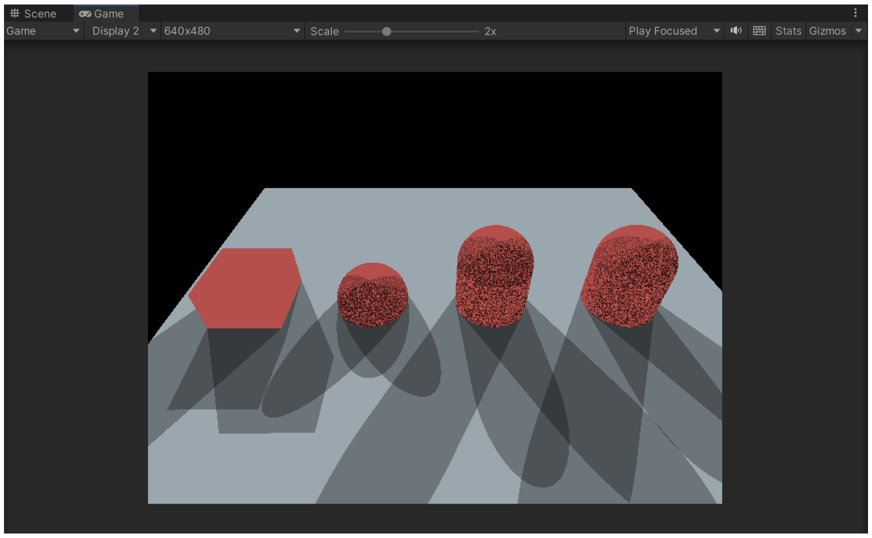
For the rotation, I used the code above.

Results

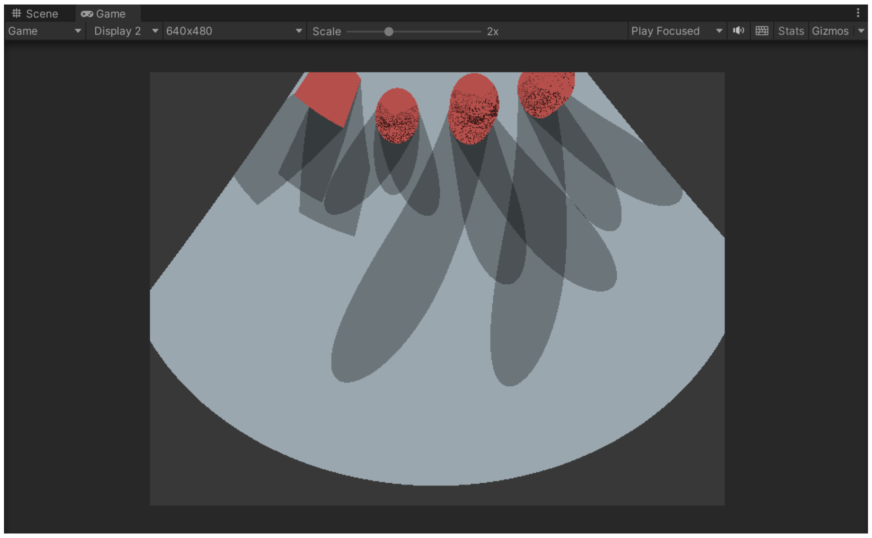
When Blackhole is Behind the Camera



(Game Camera)

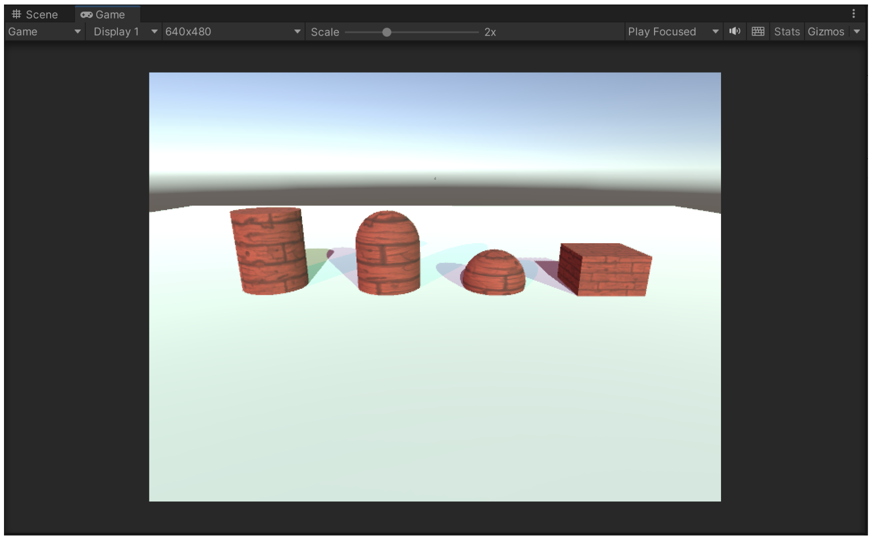


(Raw Image - No Blackhole)

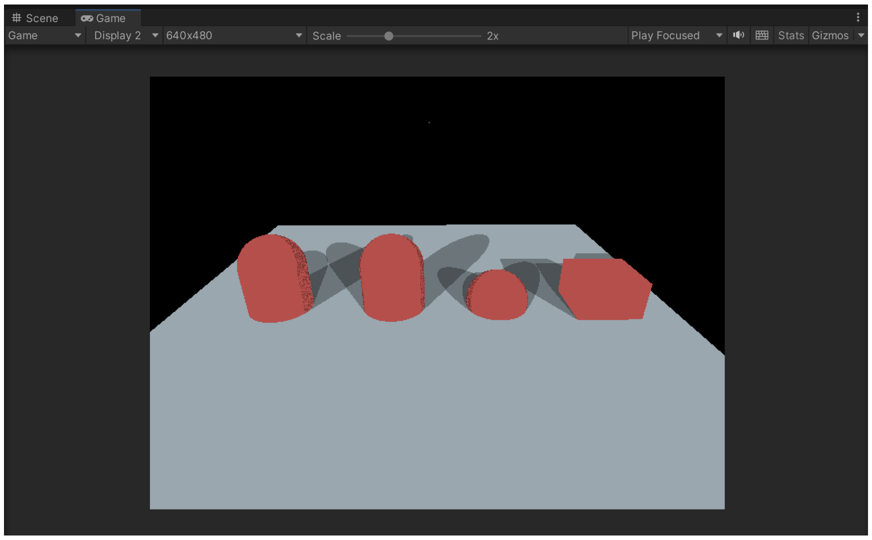


(Raw Image - With Blackhole)

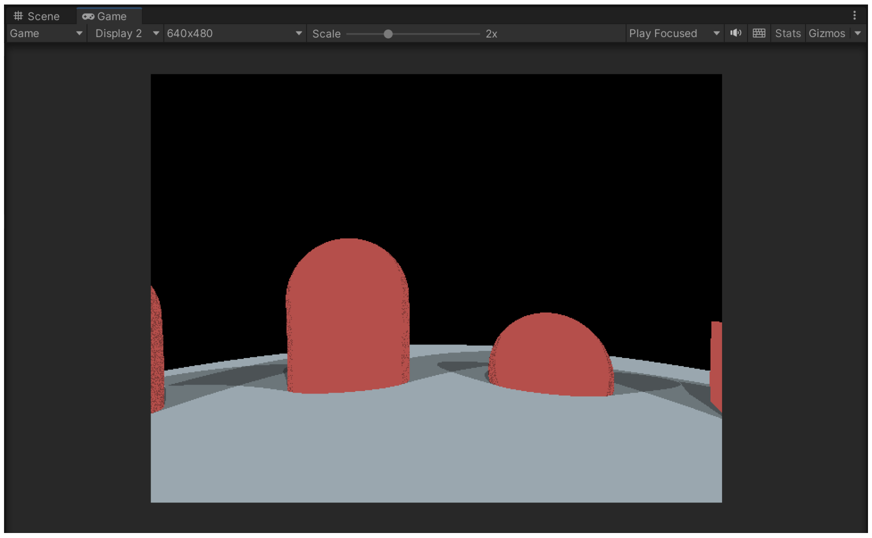
When Blackhole is in Front of the Camera



(Game Camera)



(Raw Image - No Blackhole)



(Raw Image - With Blackhole)