

Portals

Built for VR but perfect for non-VR Desktop and Mobile projects as well

Last update: November 07, 2022 - Version 1.0.6

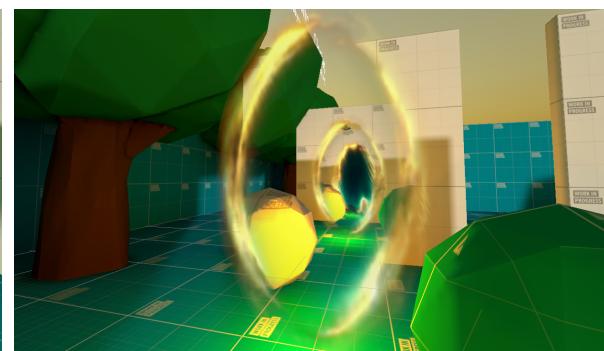
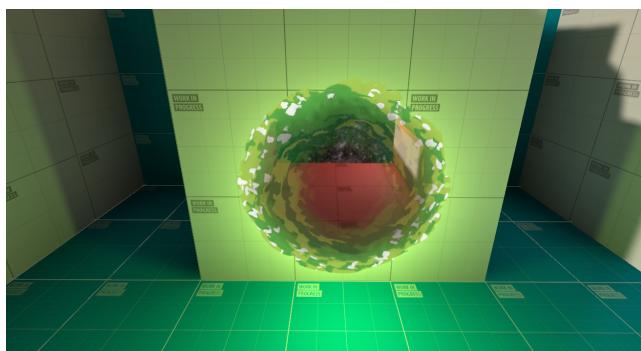
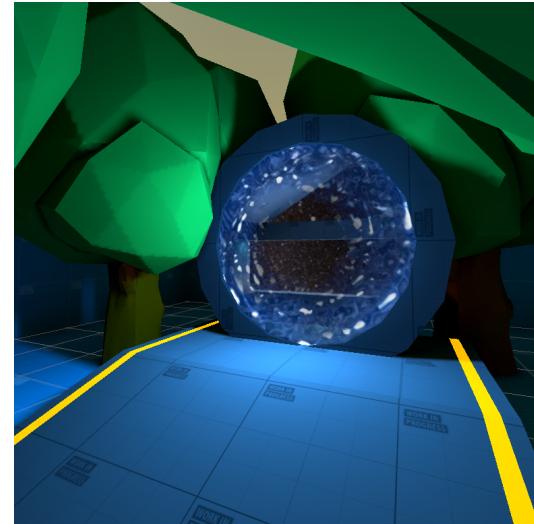
If you're looking at the pdf bundled with the asset, the documentation might have already been updated. You can find the most recent documentation here:

<https://docs.google.com/document/d/1H0ZYh-Rs5vwENm5eayC8ovxJjB5G0xh8EKm3325AQhM/edit?usp=sharing>

Warning!

Rendering portals is taxing on mobile hardware. Don't expect 2 portals seeing into each other 3 recursions deep at full screen resolution to hit framerate on an Oculus Quest 1. We're essentially rendering your scene multiple times over depending on how many portals and how deep you want the "reflections" to go.

That being said, this is most likely the fastest solution to do real-time portals on mobile VR hardware in existence using renderTextures.



Known issues

Pimax 8kx is reported to have issues with double vision inside the portal. Currently investigating but please don't buy for the moment if you're targeting this hardware.

HTC Vive has a glitch in your peripheral vision where you can see something flickering while in between 2 portals. Currently being investigated.

Compatibility

Are you working with the **URP rendering pipeline?** Then continue reading.

Are you using the standard rendering pipeline!? Request a refund.. This is **currently** not the package for you.

"Portals for VR" is created trying to be a great way to render real time recursive reflections. Everything is made to look good and to be as performant as I know how to make it.

Have a look at the below table.. Find your Unity / URP combination, then check which modes are available. **If your unity version is not included but higher than the lowest version number listed here, chances are good things will work.** It just means that combination was not verified working. This table is not complete so feel free to provide testing data by emailing.

Fragilem17@gmail.com

# Unity # URP	Non VR (PC and mobile games)	PC VR Single Pass Instanced	Mobile VR Multiview	Remarks
2022.01.05f1 URP 13.1.8	OK	OK	OK	All Good
2022.01.00f1 URP 13.1.8	OK	OK	OK	All Good
2021.03.02f1 URP	OK	OK	OK	All Good

2021.03.00f1 URP 12.1.6	OK	OK	OK	All Good
2021.02.06f1 URP 12.1.2	OK	OK	OK	All Good
2020.03.36f1 URP 10.9.0	OK	OK*	OK*	requires a skybox material to work in VR
2020.03.24f1 URP 10.9.0	OK	OK*	OK*	requires a skybox material to work in VR
2020.03.19f1 URP 10.6.0	OK	OK*	OK*	requires a skybox material to work in VR
2020.03.10f1 URP 10.5.1	OK	OK*	OK*	requires a skybox material to work in VR
2020.03.02f1 URP 10.4.0	OK	OK*	OK*	requires a skybox material to work in VR

*requires a skybox material to work. (Projection Matrices return identity Matrix when no Skybox is set.)

PS: make sure you switch the stereo rendering mode for desktop to your required setting even though you might be compiling for Android your development machine is using the setting for desktop during development.

Using it.

First, make sure you're using the URP rendering pipeline.

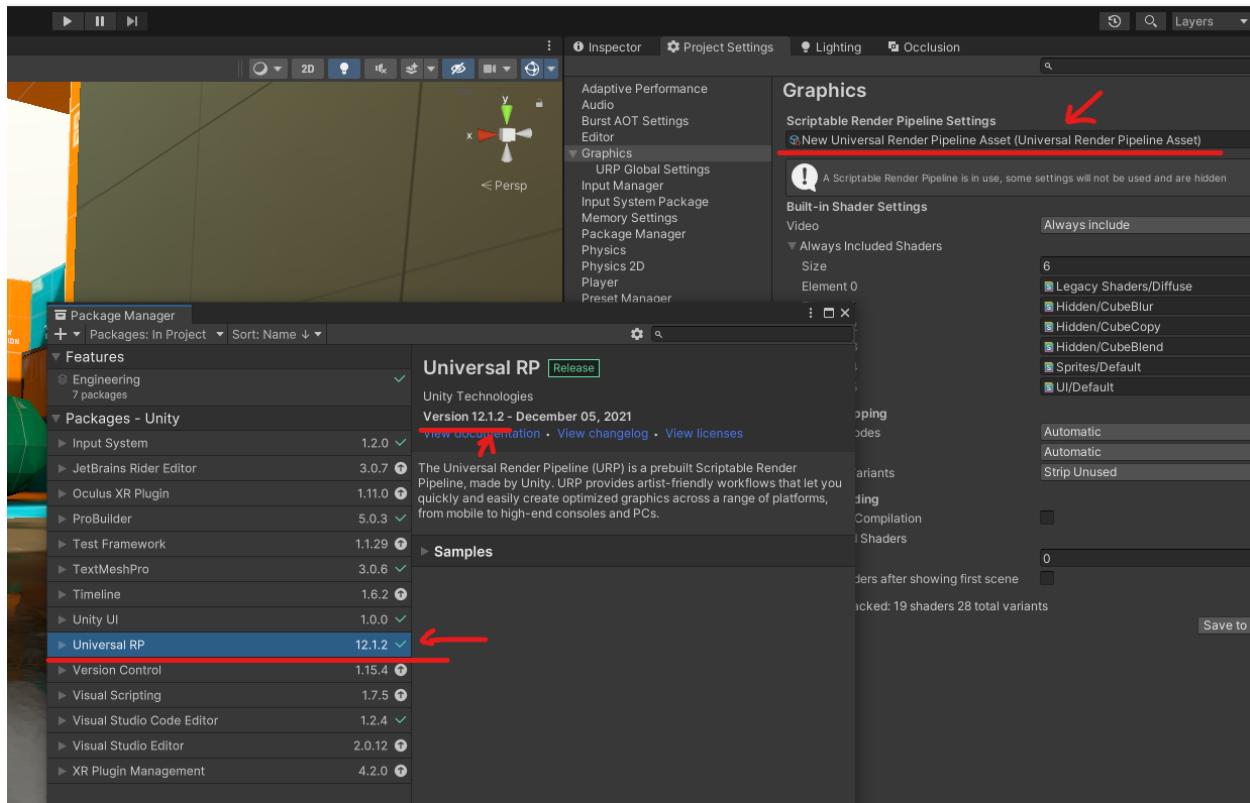
The asset was originally made using:

Universal RP 12.1.2

Unity 2021.2.6f1

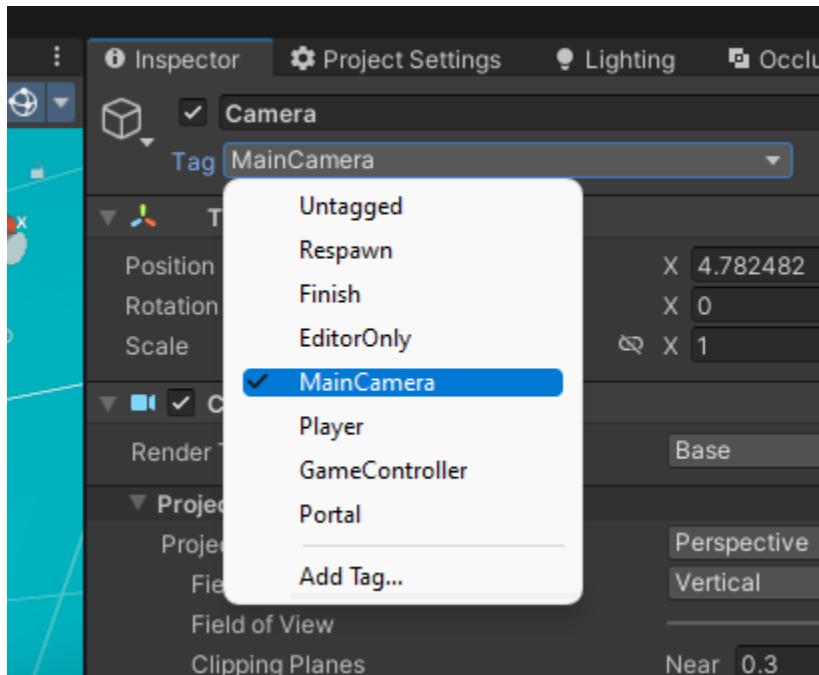
Using: OpenGL ES3

But newer versions will of course work. Check the table above for details.



Tag your camera

Portals will only be rendered for the scene view in edit mode and Cameras tagged as “**MainCamera**” or “**SpectatorCamera**”. So make sure your main camera has the tag “**MainCamera**”.

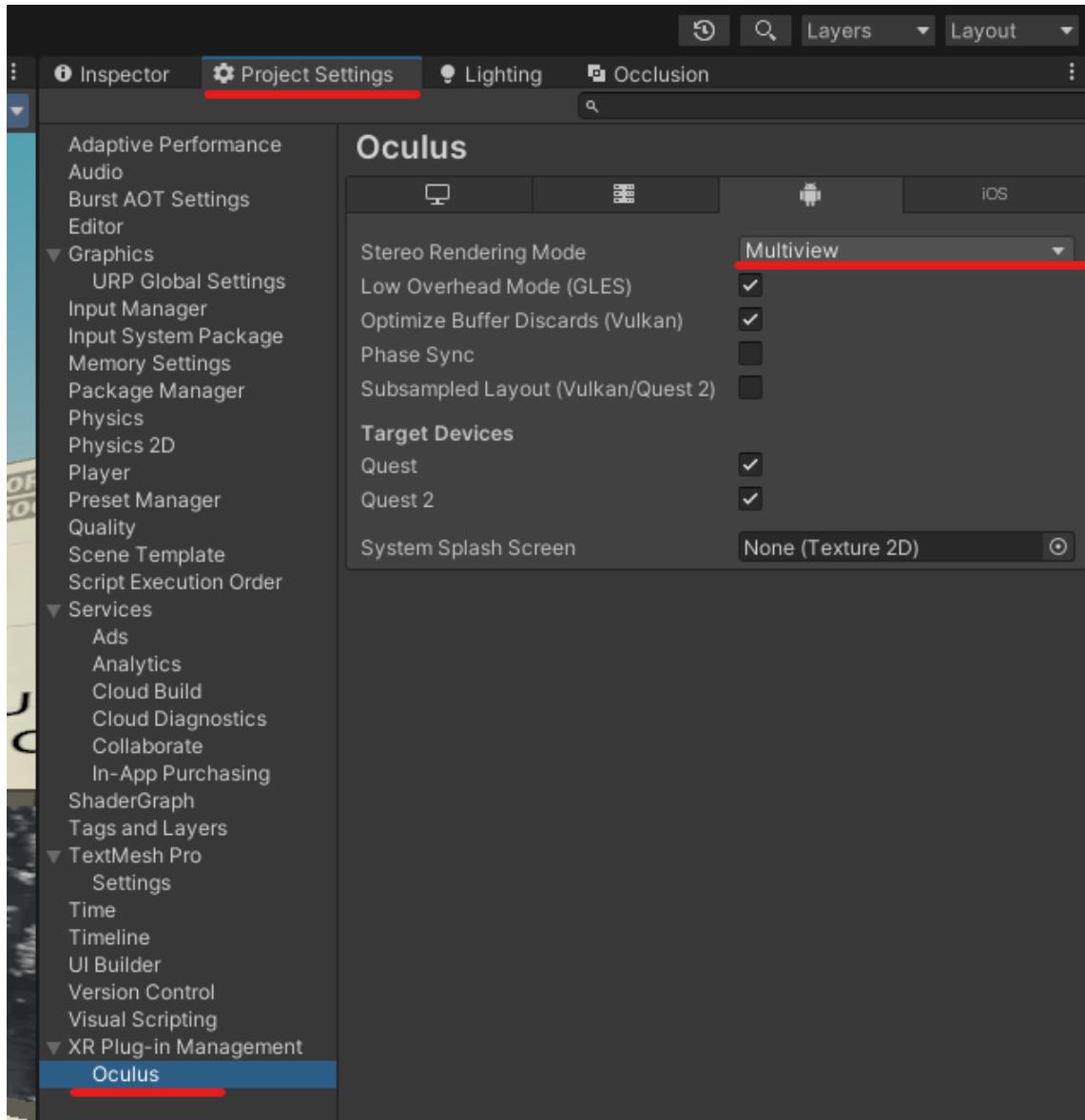


Spectator Camera

If you tag a camera as “**SpectatorCamera**” and set it to target “no” eyes then on a desktop build that camera will be used to render the view on the desktop monitor. The MainCamera will still be rendered in the VR HMD. If you make it so the Spectator Camera smoothly follows your head and has a wide FOV then people following along on the desktop monitor don’t get sick with the fast head movements. Only when the camera has the tag “**SpectatorCamera**” will the portals be rendered.

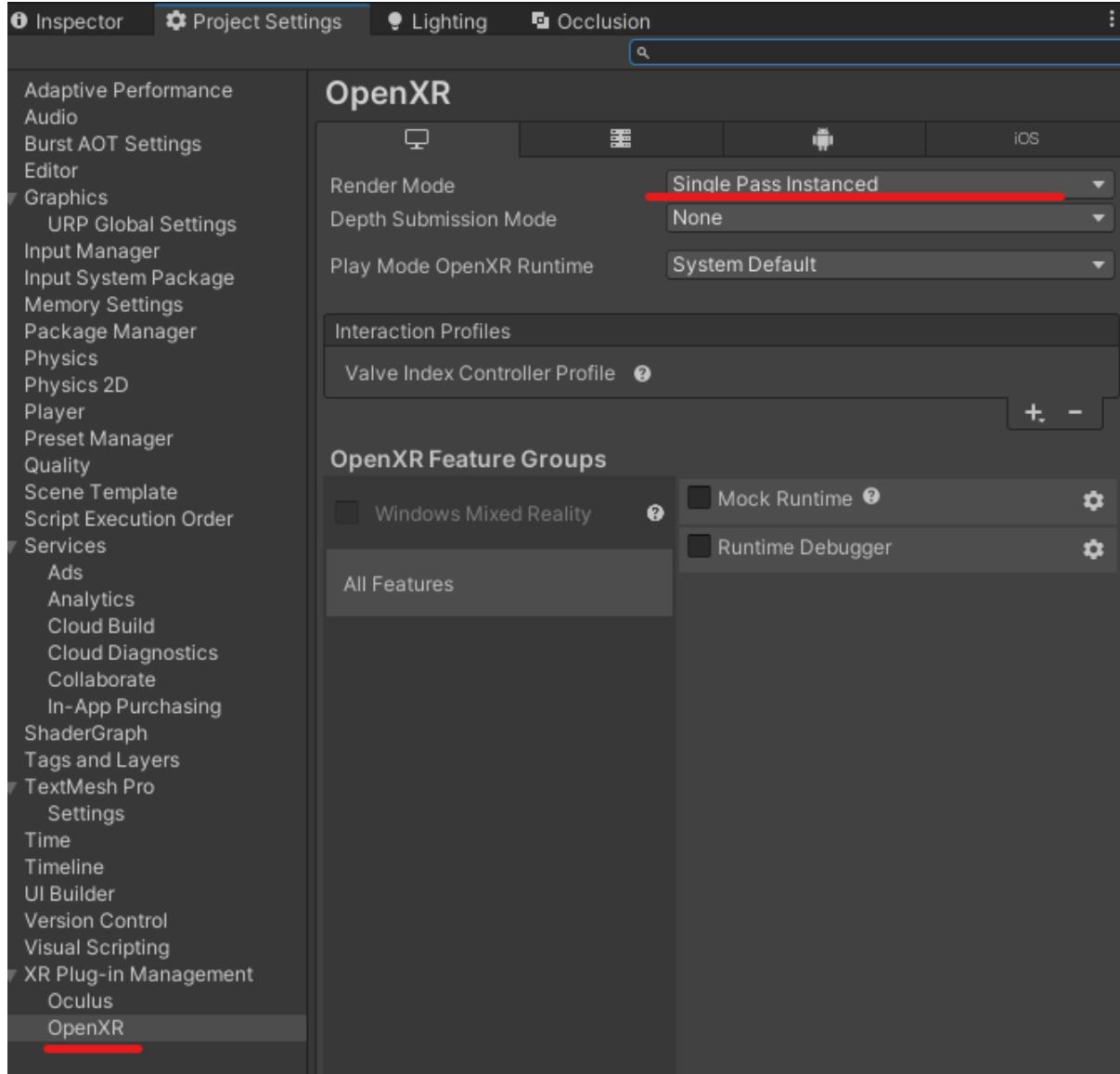
For Quest or Native Mobile VR (meta):

If you're using the Oculus integration, make sure you're rendering in **Multiview** mode! This was previously named **singlepass** and is significantly faster than Multipass. **Multi Pass is currently not supported!** NOTE: Even though you're building for Android, your development PC/Mac is a desktop.. So switch over to desktop, and make sure you've set the rendermode to **singlepass** instanced there as well.



For Oculus Link / Valve Index / Vive / any steam VR OpenXR based headset

If you're using OpenXR then, make sure you're rendering in **Single Pass Instanced** mode. This is significantly faster than Multi Pass. **Multi Pass is currently not supported!**

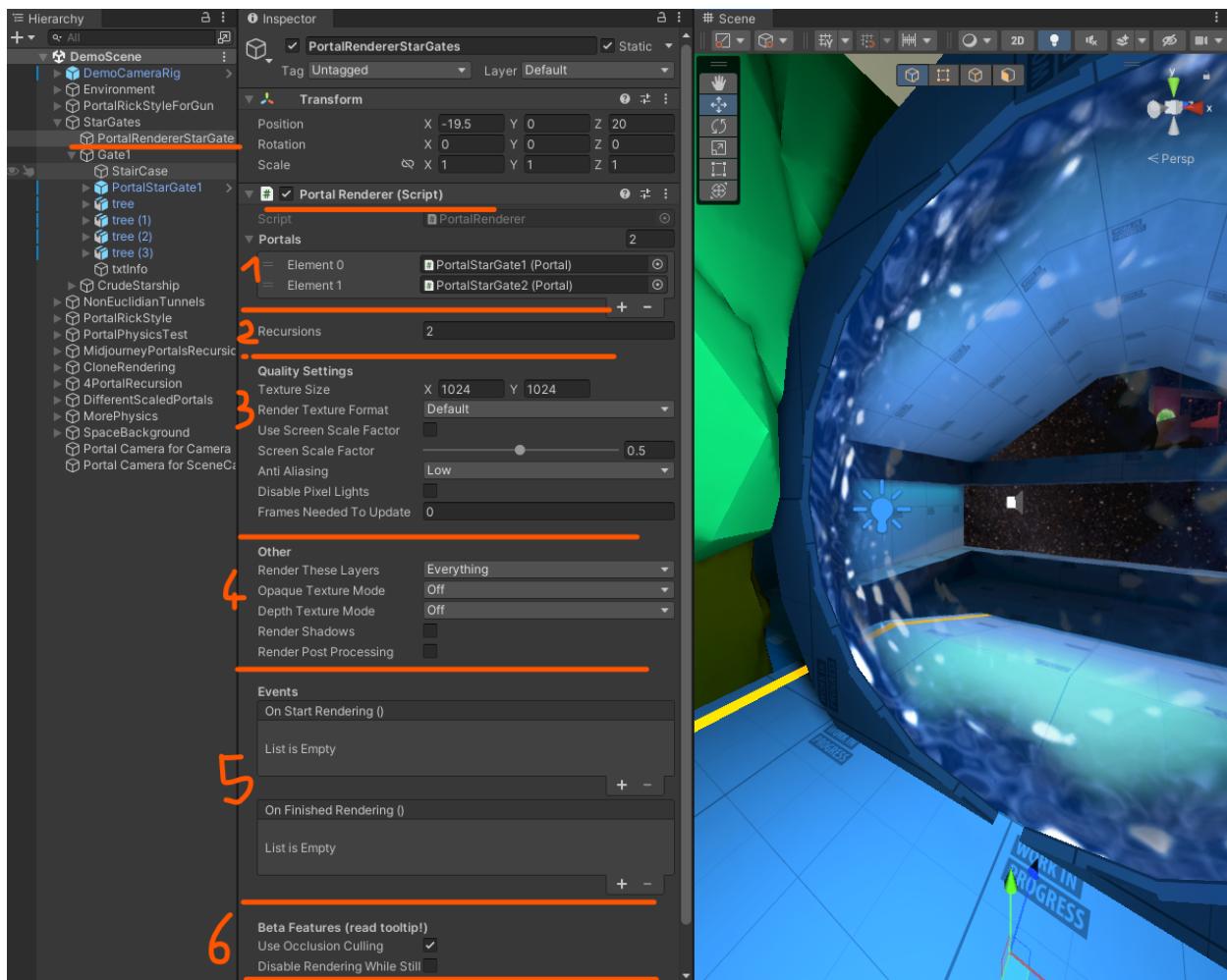


(Even though we're still stuck rendering the reflection as a pass per eye, i'm keeping an "eye" out for changes to URP to allow a rendertexture to be rendered as a double wide texture, I made a bug report explicitly showing that this is not working in URP while it used to in the deprecated old rendering pipeline)

Open the Demo Scene

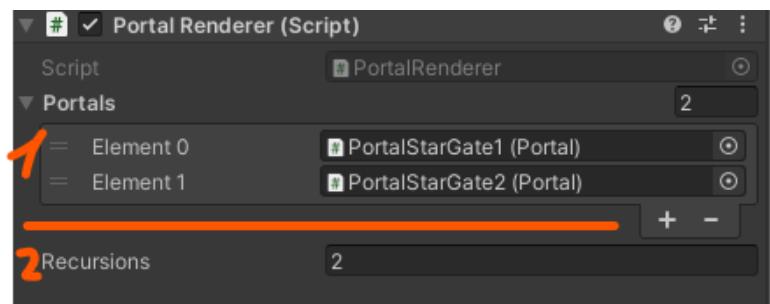
PortalRenderer

Have a look at "StarGates" or "MidjourneyPortalsRecusion", Select the Gameobject **PortalRendererStarGates**. It contains the **PortalRenderer**. Here you define which surfaces are to be rendered. Usually you'll only assign your 2 portals, but you can put as many here as you like. They will all be able to "reflect" each other with the settings defined in the renderer. (all the same resolution) You can have multiple PortalRenderers with their own settings, but **Portals** will only be able to "reflect" each other when they're in the same renderer.



9

1 Here we define which Portals are to be rendered.



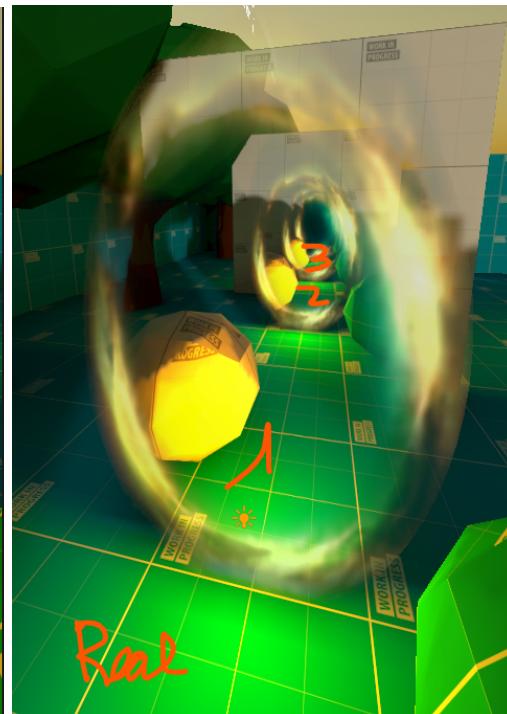
2 “recursions” define how many reflections of our own portal we can see in the other portal.

This defaults to 2. If a portal can't

“see” the other portal it won't render it even though the recursion depth is higher than 1. **For Portals to correctly render when you're “inside” them in VR you need to have this at a minimum of 2 because the left and the right eye both need a reflection at that time.** “PortalSurfaces” (keep reading) have a “Max Rendering Distance” that might stop you from seeing the deeper recursions because they're too far away. If they are, they won't be rendered either, saving you some precious GPU cycles..



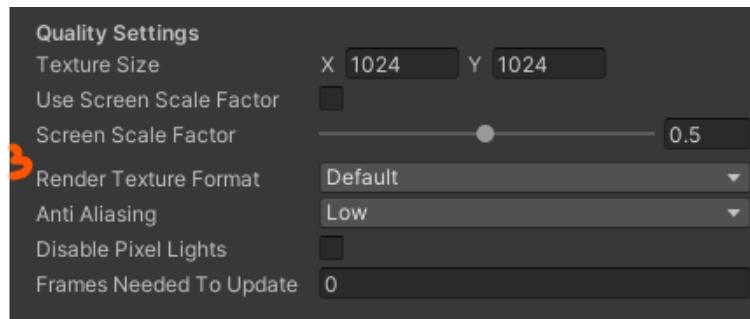
A recursion of 2



A recursion of 3

3 Quality Settings:

“Texture Size” defines the resolution at which the portal texture will be rendered. Square texture sizes seem to be a bit faster on Quest. I usually go with 1024x1024



“Use Screen Scale Factor” when enabled the texture size is calculated as a factor of the screensize. 0.5 is a good default value. On Quest 2 the display size is 3840x1832 a Factor of 0.5 would result in a texture resolution of 1920x916. I do prefer a fixed resolution if I know the target platform.

“Render Texture Format” Choose **Default** for 8 bits per color channel textures or Default HDR for 16 bits. **Default** is faster but might give some banding in the portal image.

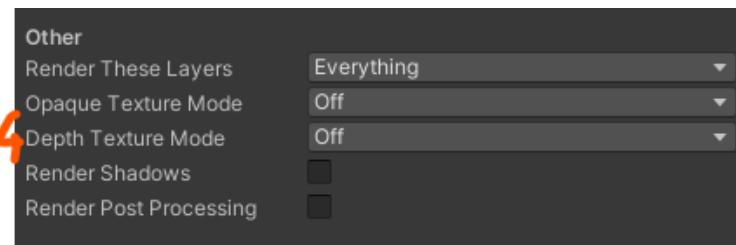
“Anti Aliasing” It’s not free but best to use at least a low setting.

“Disable Pixel Lights” If you can get away with real time pixel lights (you’re probably not doing mobile VR), You can turn them off in the portal reflections.

“Frames needed to Update” Leave this at 0, in VR even a 1 frame drop to save performance looks terrible. But it might be useful for mobile games outside of VR.

4 Other:

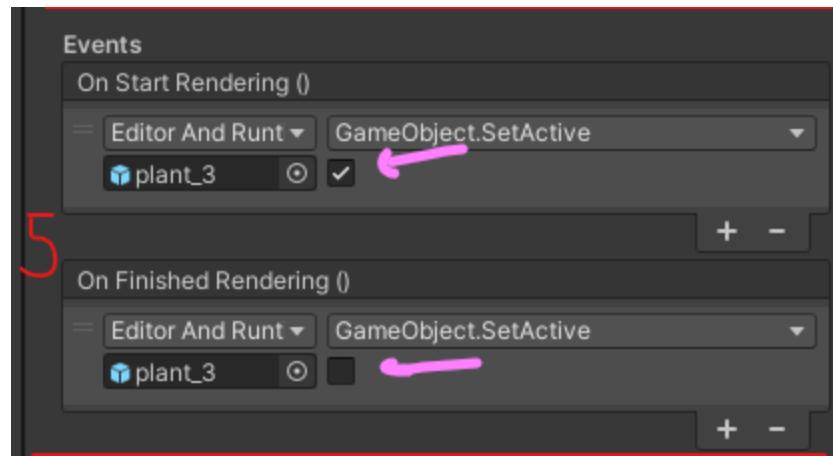
“Render These Layers” limit the things to render by changing the layer mask of the Portal Camera.



The rest of these settings speak for themselves and reflect the settings any other camera would have. If you’re doing mobile VR you’ll want all of these things to be the default values. (Off)

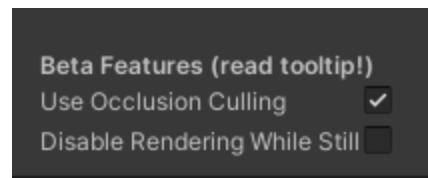
5 Events:

These 2 events allow you to change the scene just before the reflection gets rendered and then change it back. Use it for example to scale your head down in the VR camera and scale it up for the reflection camera. Can be used as an alternative for Culling masks.



6 Beta Features:

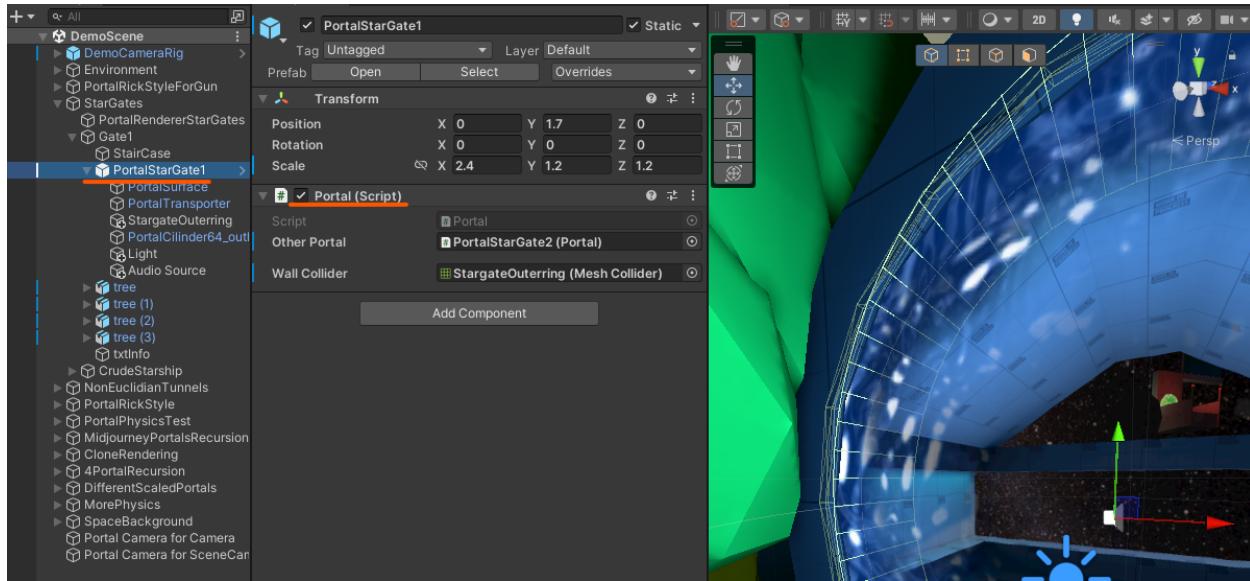
"Use Occlusion Culling" whether or not the camera that renders the portal is using Occlusion culling. Sadly there's a bug in Unity where an Oblique Occlusion Culling Matrix is not working correctly. This option is here mainly in the hope some Unity version will have a fix and we'll be able to turn this on.



"Disable Rendering While Still Updating Materials" When checked, the portal will stop rendering but the materials will still update their position and blending settings.

Portal

Our **PortalRenderer** needs something to render. In comes the **Portal** Script.



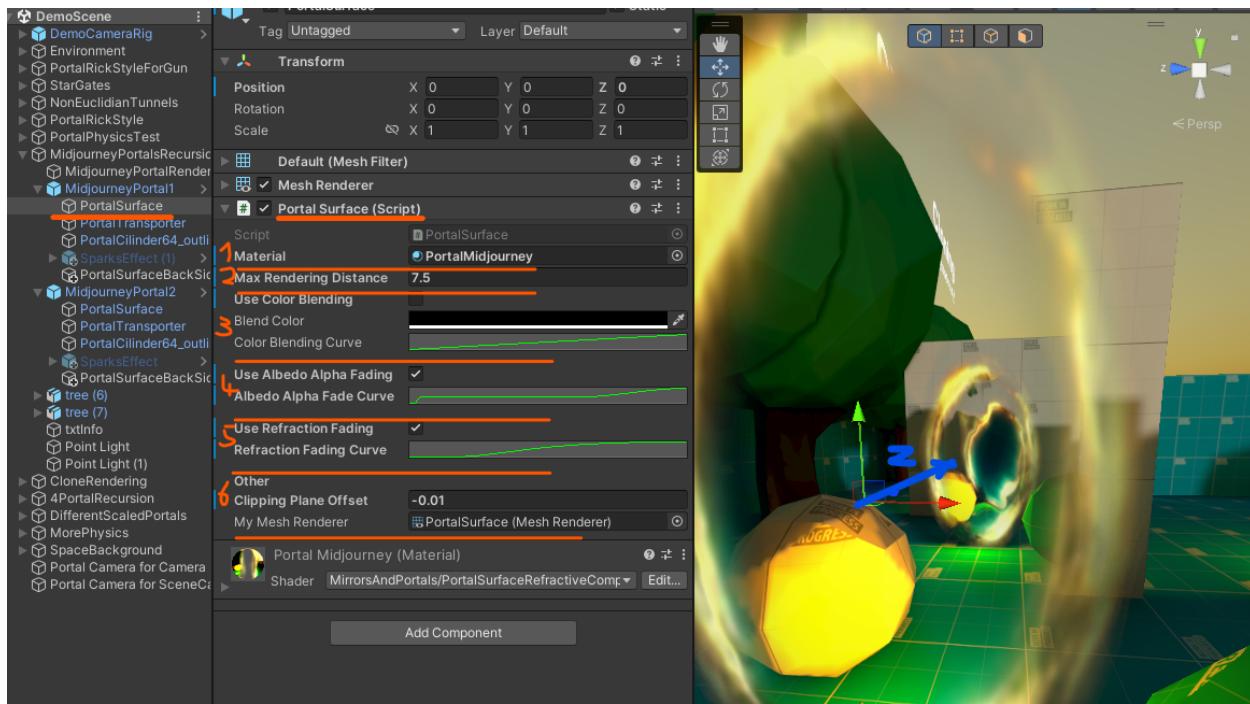
“Other Portal” A Portal always points to another Portal, just drag a reference in the scene to the other Portal in this slot.

“Wall Collider” If a portal is against a wall with a collider you can't just walk through that wall.

Putting the collider in this slot allows your “Portalable Object” to walk right through.

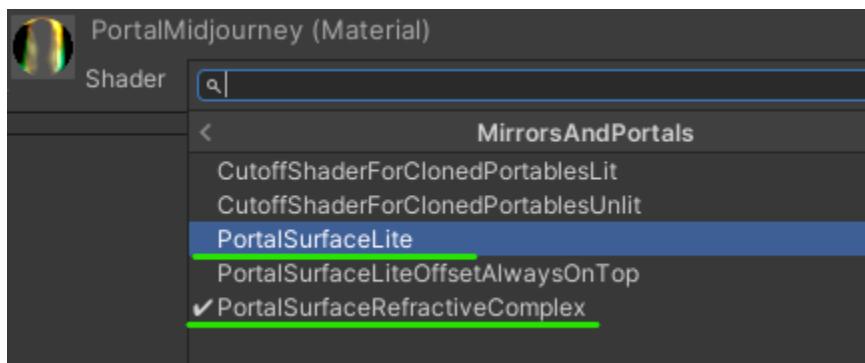
PortalSurface

Each **Portal** can have 1 **PortalSurface** (either as a sibling or a child of the Portal GameObject)



1 "Material" A surface needs a Material. Start by making a material and dragging it into the slot.
Both Portals can use the same material instance of different ones, does not matter.

There are 2 PortalSurface shaders. a real simple unlit shader for simple portals and another unlit advanced shader that allows you to do refractions transparency and other nice effects.



2 "Max Rendering Distance" an important one! The portal stops rendering the "reflection" when you're further away from this value. This includes recursions!

So if 2 portals are **3 meters apart** looking at each other and the **Max Distance is set to 4**. You are standing in between the portals then you'll see the reflection of 1 portal but the other will be too far away to also reflect your back. (Distance $1.5 + 3 > \text{maxDistance } 4$).

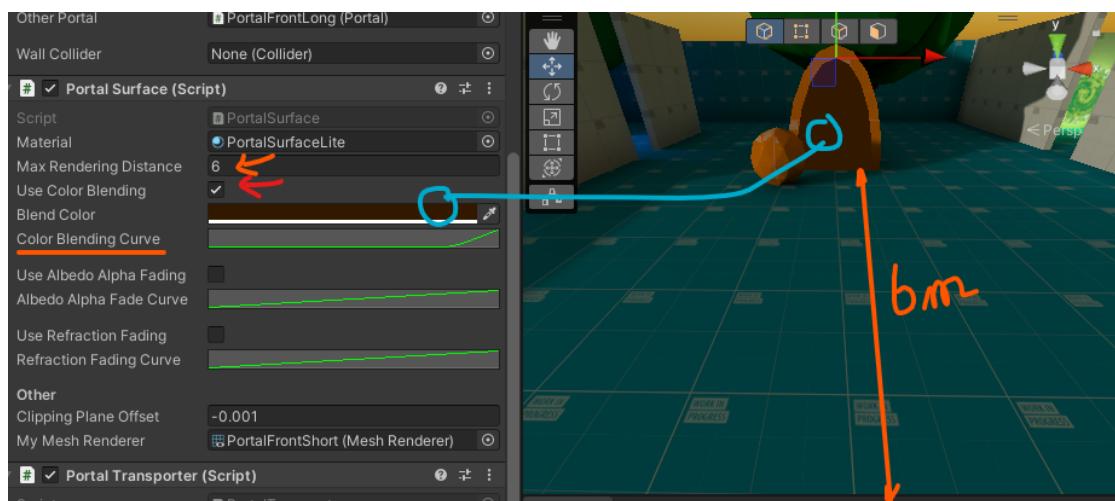
As you walk closer to P2 the reflection of your back will start appearing through P2 in P1 once you get within **1 meter** of p2. (Distance $1 + 3 == \text{maxDistance } 4$)



3 "Use Color Blending"

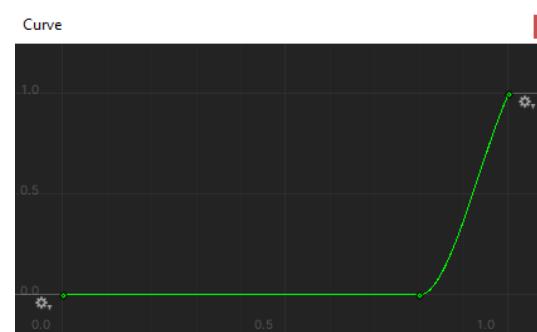
Supported by both the simple and the complex shaders.

When checked the portal will fade the "reflection" to a solid color over distance until it's at the Max Rendering Distance.



How much color is blended in over distance can be controlled with the "Color Blending Curve"

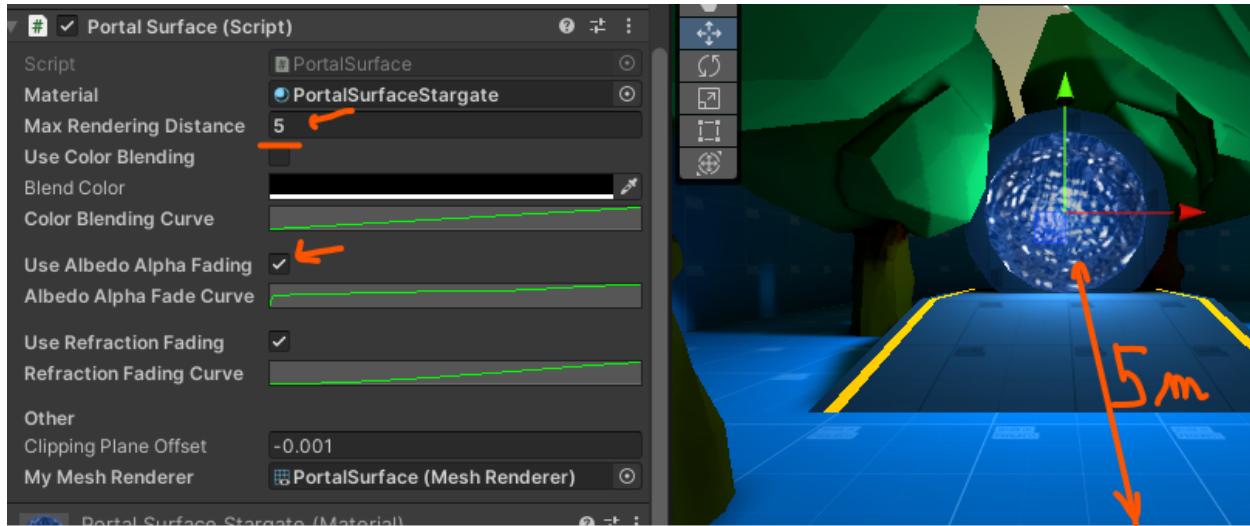
Zero on the left means zero color will be blended in at distance 0% up until 80% where it goes smoothly to 1, meaning at $0.8f * \text{MaxDistance}$ the color will start to blend in.



4 "Use Albedo Alpha Blending"

Supported by both the simple and the complex shaders.

When checked the portal will multiply the “albedo” texture’s alpha over the reflection in function of distance to the portal. This allows the “portal opening” effect as you get closer.



In the screenshot you'll see the stargate texture is fully opaque. In the curve you can see the alpha is multiplied with 1.5 (right side as we're more than 5 meters away)

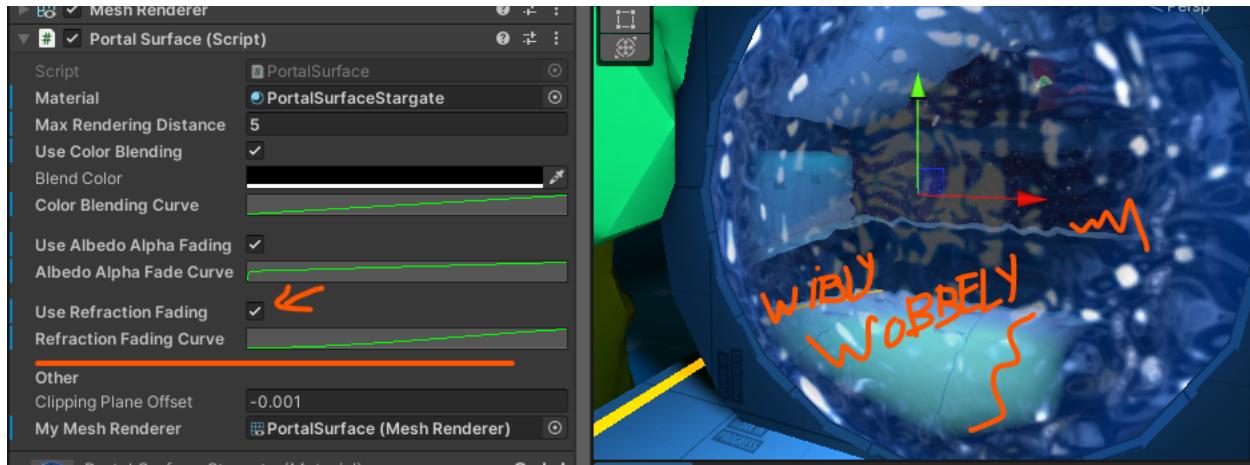
As we get closer the alpha will get lower until it drops quickly to 0. (if we're up close with our nose against the texture) You'll only see the reflection at that point.



4 “Use Refraction Fading”

Supported by the complex shader.

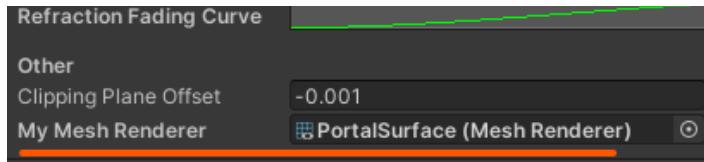
When checked the portal will use the provided normal map to calculate refractions in the “reflection” The curve allows you to control how much refraction is happening over distance.



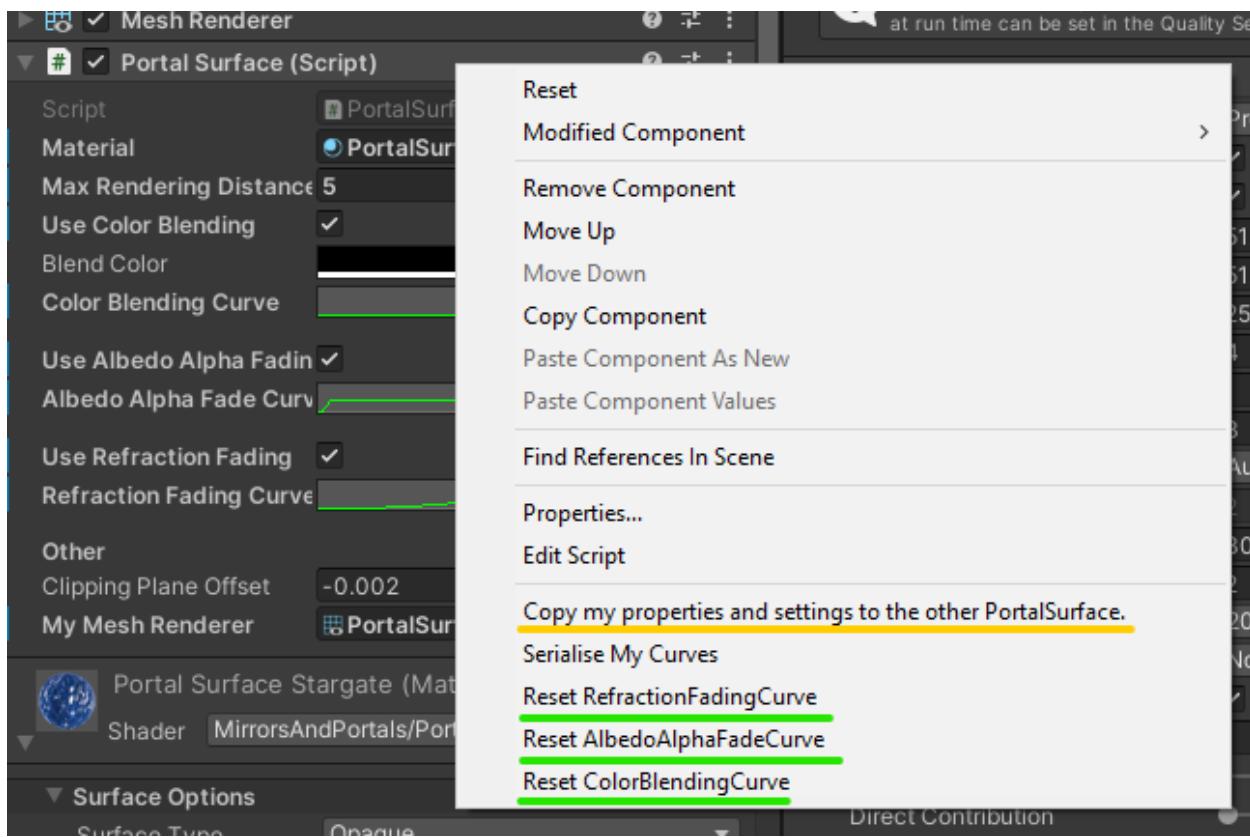
In the screenshot you'll see the reflection being all wibby wobbly, as we get closer the ripples will get less and less depending on how you set your curve. You want it to be 0 when you're up close.

5 The “Clipping Plane Offset” allows you to move the reflection plane forward or backward from the actual Surface plane, a value if -0.001f usually gets rid of some small artifacts.

“My Mesh Renderer” If your model does not have its pivotpoint set correctly you can create an empty GameObject with the PortalSurface script attached to it. Rotate and position that gameobject correctly with its blue arrow pointing away from the surface. Then drag the actual model in the “My Mesh Renderer” property. I do not recommend doing this as it messes up selecting your material instance.



In case you completely mess up the curves and want to get back to something sensible you can right click on the PortalSurface component.

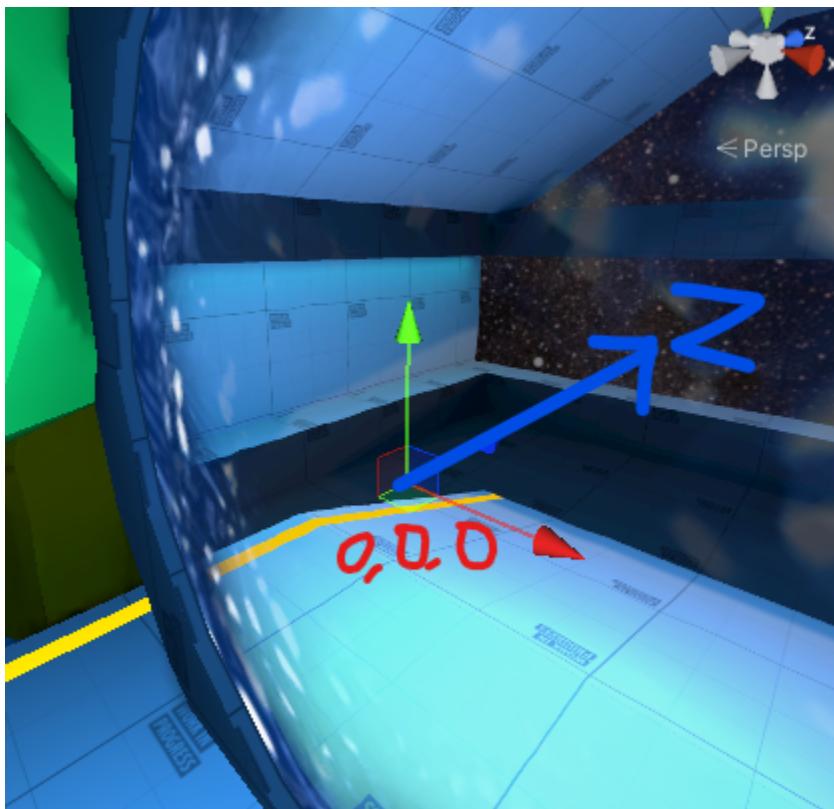


Creating your own portal geometry

Forward direction

You can use any shape planar model. (a square, a circle, a heart, egg shaped) As long as it's a planar surface and the pivot point and normal are facing the correct direction.

The "reflection" will be calculated looking back from the blue arrow of the model. So make sure to be looking at your local coordinates and that the blue arrow is looking away from the visible side of the plane. This is the default for the Unity 3D Quad.



The Portal Surface Shaders

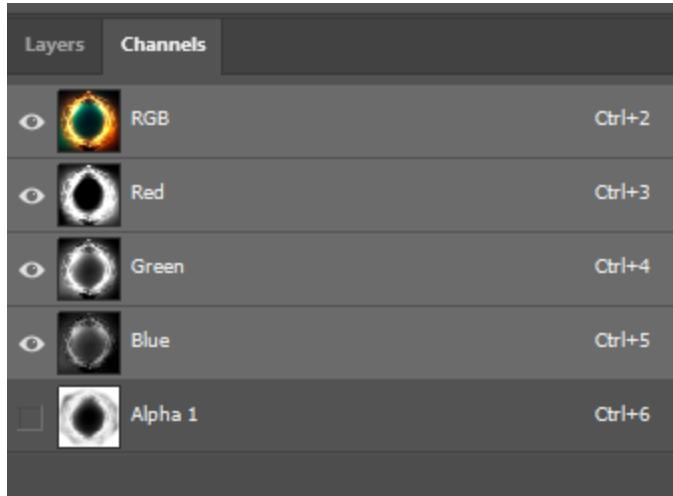
Both PortalSurface Shaders are pretty self-explanatory however:

This effect where the portal opens up requires an explanation on how to prepare your texture.



You barely notice in the screenshot but there's a gradient from almost black to whiter towards the edges.

When “**Use Albedo Alpha Fading**” is turned on in your **Portal Surface** the alpha channel of your albedo texture will control how transparent it is. The easiest way I know of to create an alpha mask is by using Photoshop and adding an alpha mask there.

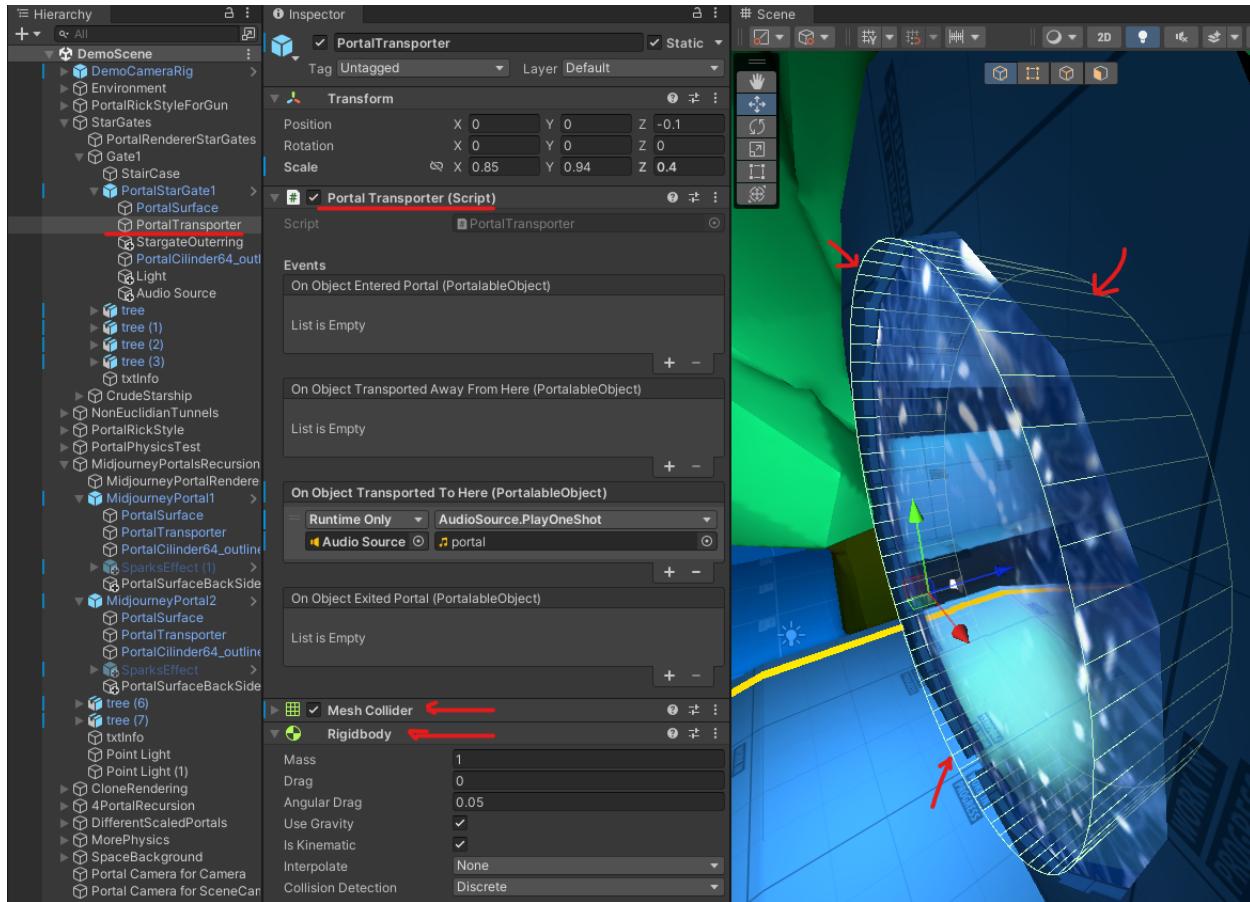


(<https://www.instructables.com/How-to-use-Photoshop-to>Create-Textures-With-Alpha/>)

The rest of the shader is pretty self-explanatory. Don't forget to mark your normals as normal maps.

Portaling Through a Portal

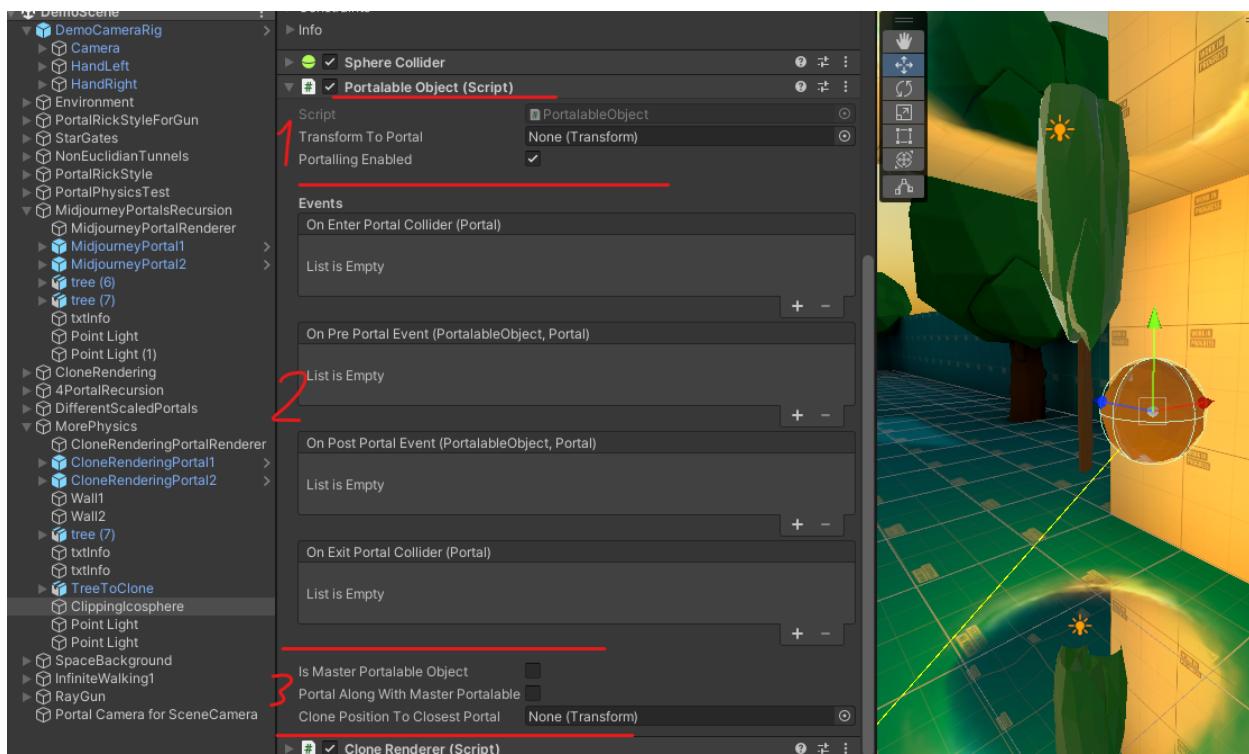
PortalTransporter



For an object to be able to transport through a portal the portal needs a **PortalTransporter** along with a **Collider** and a **Rigidbody**. The collider needs to be able to capture the object before it touches the potential wall the portal is attached to behind it. (It needs to stick out a bit). It also needs to extend a bit behind the portal. (Things will only **Clone** themselves while touching the collider, keep reading)

PortalableObject

Objects that want to be able to “Portal” (transport) through a portal need a “**PortalableObject**” component along with a Collider. You **playerController** would also need to have a “**PortalableObject**” component.



1 “Transform to Portal” You can leave this empty if you want to target the transform from this object. More complex objects like your **playerController** might need a different “root” element to move when you pass through.

Important to realize is that the object will transport once it’s **pivot point** is on the other side of the Normal of the **PortalSurface**.

In the case of a **playerController** the **PortalableObject** needs to be on the camera’s pivot point but in most cases you don’t want just your camera to transport but the whole **playerController**. That’s where you can enter the transform of the controller in the “**Transform to Portal**” slot.

Have a look at the DemoCameraRig for it to make sense.

So when Camera passes through the normal of the Portal's Surface, DemoCameraRig will be transported taking the camera with it as it's parented to it.

"Portalling Enabled" used to disable the portalling of an object (safer than than disabling the whole component as cloning will continue to work)

2 "Events"

OnEnterPortalCollider: When the portalableObject enters the collider.

OnPrePortalEvent: Just before the object is warping, the transform is still at the first portal

OnPastPortalEvent: Just after warping, the transform has been repositioned

OnExitPortalCollider: When the object leaves the portal collider (does not mean it has been portalled)

3 "Is Master Portable Object"

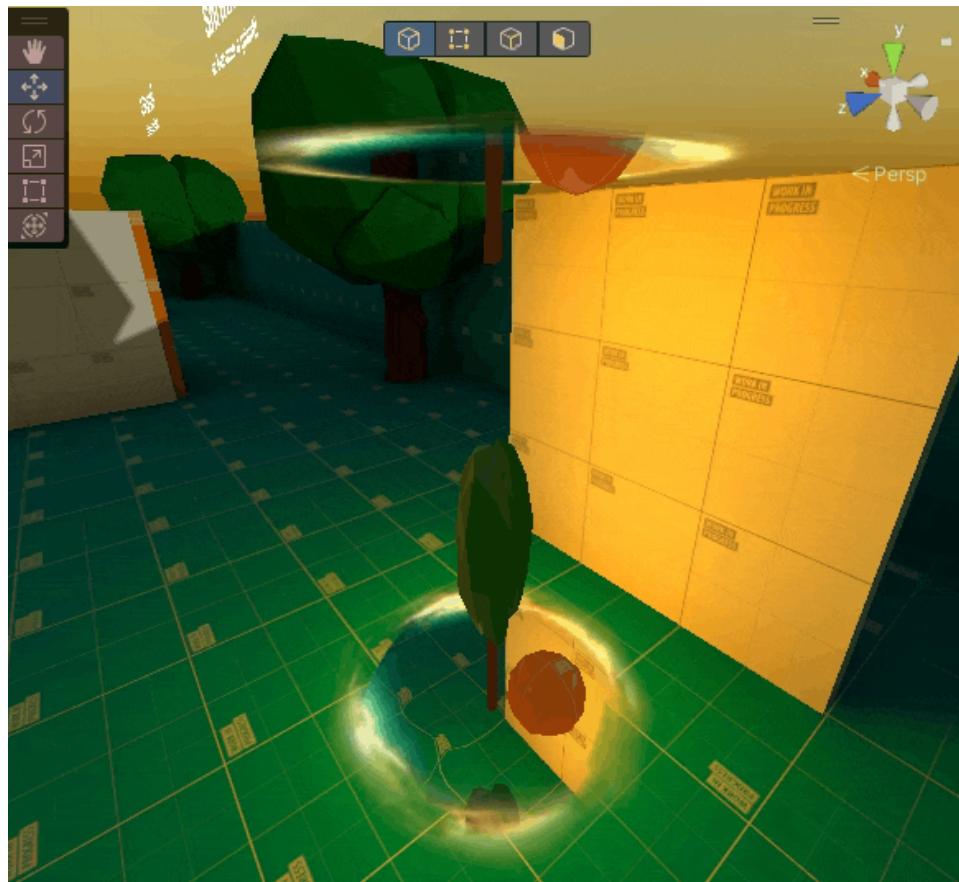
Set this to true if this is your character controller. All items that have

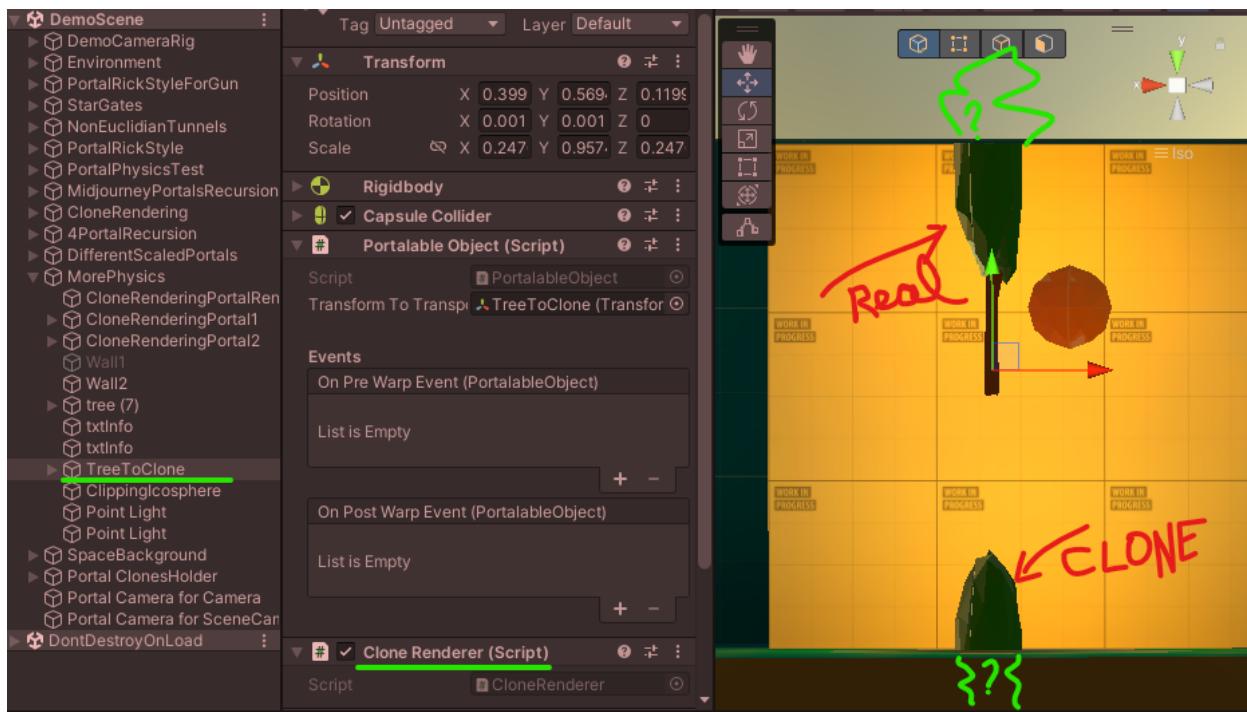
"PortalAlongWithMasterPortalableObject" enabled will transport along when the master portals.

"ClonePositionToClosestPortal" If a transform is present in this slot, that transform will always be at the nearest Portal's OtherPortal in relation to your current position. Comes in handy for third person characterControllers (look at the CinemachineExampleScene for an example) or here: <https://youtu.be/1-q7OfoB-II>

CloneRenderer

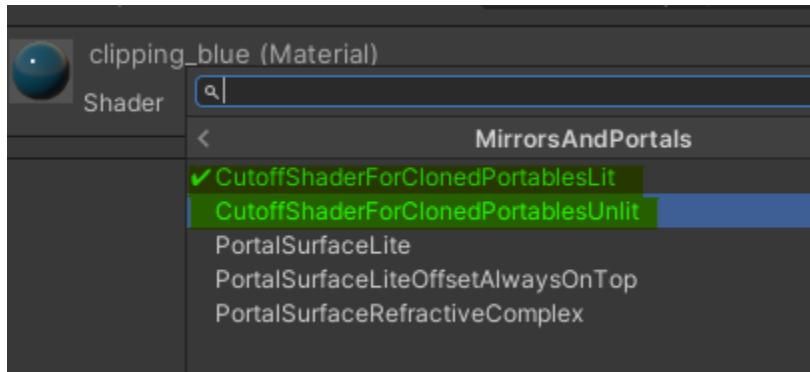
The CloneRenderer is the last piece of the puzzle. Have a look at the animation below (if you're looking at the PDF, you're not missing much) The tree and the ball exist twice in the scene, once as the real object and once as a clone. (This effect only runs at playTime)



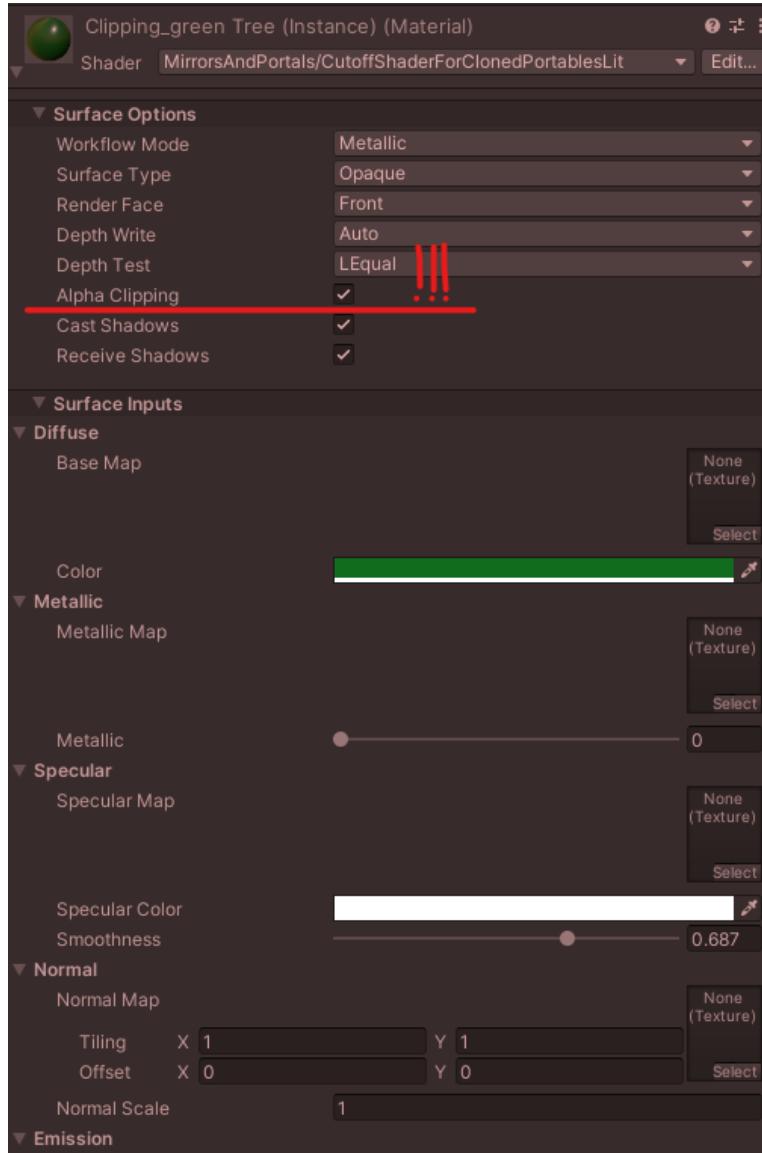


The **CloneRenderer** will copy all child objects and materials to render them at the other side as long as they're in the **PortalTransporter's** Collider.

Notice how the top of the top tree is missing and the bottom of the cloned tree! The Materials are being clipped. For this to work you need to use one of these shaders.

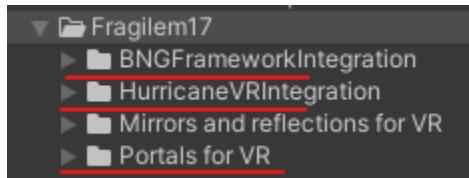


Make sure you check the Alpha Clipping checkbox in the Surface Options of your material.



BNG VRIF & Hurricane

Both VRIF and Hurricane have an example scene contained in a zip. **Unzip** the files in a folder **above** the “Portals for VR” folder. (Otherwise because of the assembly definition file the BNG or Hurricane files won’t be found)



For the Hurricane Integration to portal smoothly you need **version 2.91 of Hurricane** and up.

Hurricanes integration currently does not support scaling the player by going through differently sized portals.

Hexabody integration is currently not supported but next on the list to be looked at.

The example scenes contain elements only available when you purchase Hurricane or BNG. These assets contain materials that are not set to URP, you can convert them to URP using Unity’s built in converter. Some materials should use the CutoffShaderForClonedPortables Shaders if you want the clipping to work. Don’t forget to check the Alpha Clipping checkbox. Should your object disappear, check the alpha on the color of the material.

Thanks!

If you have any questions, don't hesitate to send me a mail at

Fragilem17@gmail.com

I hope you create beautiful things with this and don't forget. Framerate is life! Kill your darlings and if you can't hit framerate it's better to not use it.

I've spent an **enormous** amount of personal time creating this so thanks for your support in buying and not pirating this asset! It is very much appreciated.

Tom