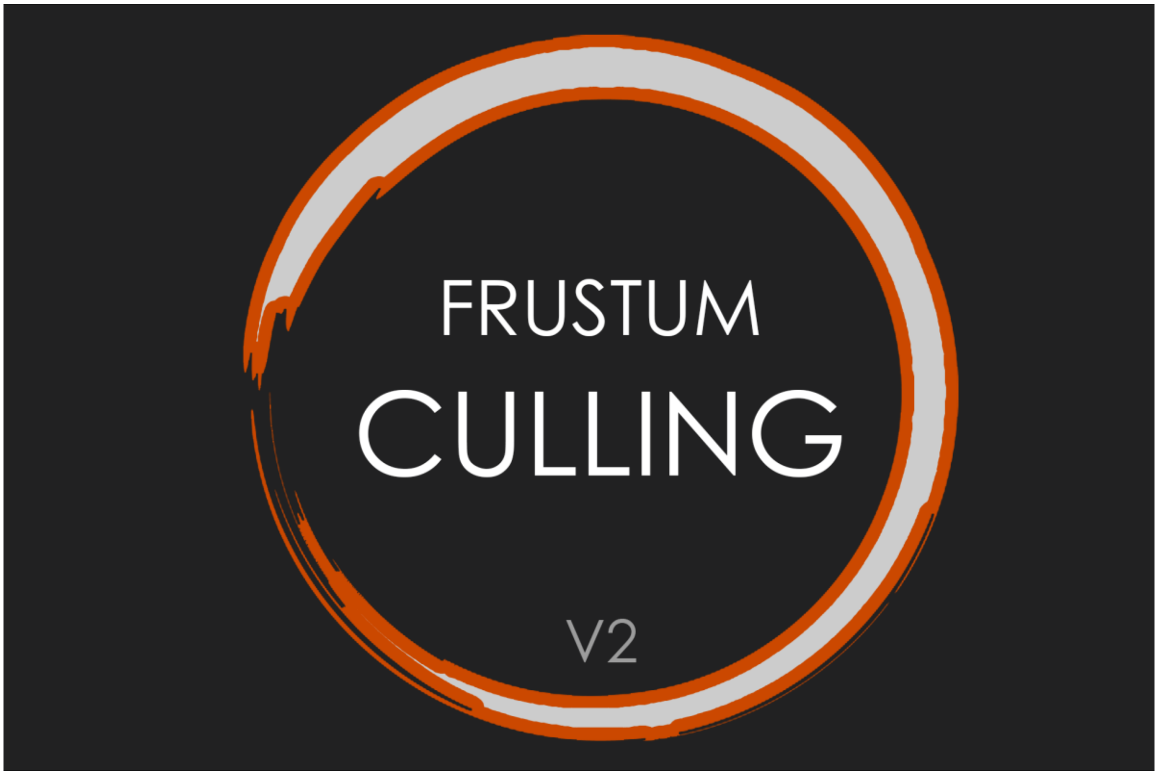


FRUSTUM CULLING DOCUMENTATION

Feel free to email me at ***pathiralgames@gmail.com*** *for support*

Find more packages from Pathiral here

ABOUT

Disable game objects completely that're out of the camera's view (frustum) or distance.

P.S: Unity already does *Frustum Culling* by default under *Occlusion Culling* for static objects but it disables the renderers only. You'll need a custom solution to completely disable the object. This is where this solution comes in. Extremely helpful to disable objects that are very costly to have on but you need them enabled when in view. Things like an AI standing still and playing animations, small volumetric lights, objects with highly intensive scripts, etc…

GETTING STARTED

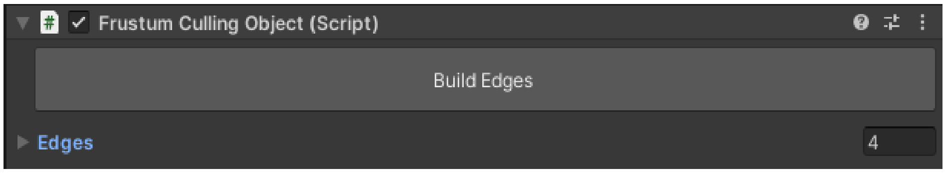
1. Create and add an empty game object to your scene and on that object add the FrustumCulling component.

2. Leave the default values of FrustumCulling properties as is.

3. Create a Cube and place it in front of the game camera. This will be the object we'll test on. When out of view, we'll make this cube disable.

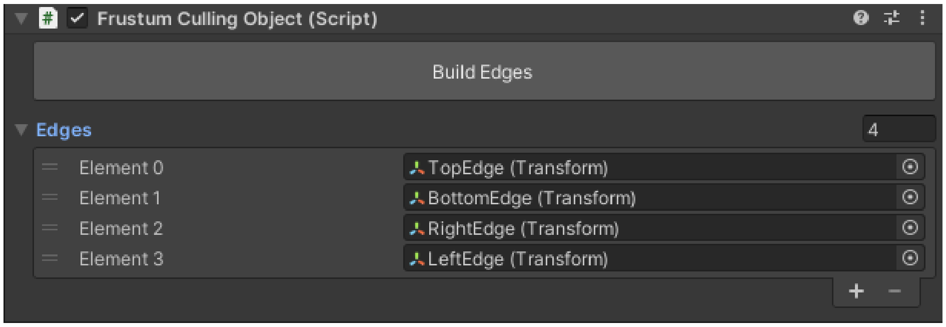
4. On that cube add the component FrustumCullingObject. Any object we want to cull, we need to add to it this script **WHERE THE MESH RENDERER IS**. This is very important.

5. After adding the script click on Build Edges button:



Edges 4

6. After clicking on the button you'll notice 2 things have happened. First thing is the Edges array has now been populated as so:

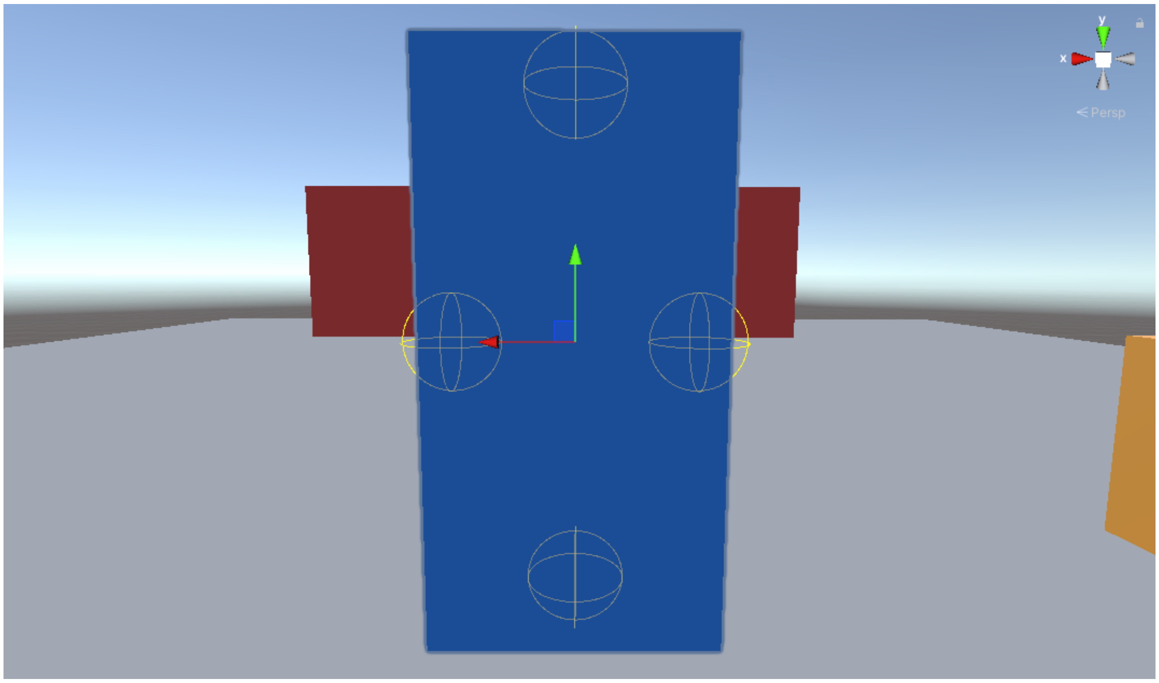


Edges 4

Element 0	TopEdge (Transform)
Element 1	BottomEdge (Transform)
Element 2	RightEdge (Transform)
Element 3	LeftEdge (Transform)

7. Second thing you should have noticed is that the hierarchal structure of the cube has been changed. It is now a parent to 4 other game objects. These 4 game objects represent edges. Each one is an edge.

8. Selecting the parent and opening the scene view, you'll find the generated edges have been placed as yellow spheres on each side of the cube. **Be careful though, automatic placement of edges aren't always accurate and sometimes you'll have to manually place them after generating them by moving each individual game object**.



9. You have now set the edges to your cube. Now start the game and move your camera away from the object. Keep your eye on the hierarchy or open the scene view side-by-side. You'll see the cube gets disabled when it's out of camera view.

10. That's it!

PUBLIC APIS

These are public methods you can call programmatically.

Add(GameObject) : This is automatically called by the FrustumCullingObject script on *Start()* to add the current game object to the list of objects to cull by the system. If for some reason you need to call it manually, there you go. *The FrustumCullingObject needs to be already added to the passed game object.*

Used:

```
FrustumCulling fcScript = GetComponent<FrustumCulling>();

fcScript.Add(gameObject);
```

Remove(GameObject) : If you would like for some reason to remove an object from the list of objects to cull, you can call this.

Used:

```
FrustumCulling fcScript = GetComponent<FrustumCulling>();

fcScript.Remove(gameObject);
```