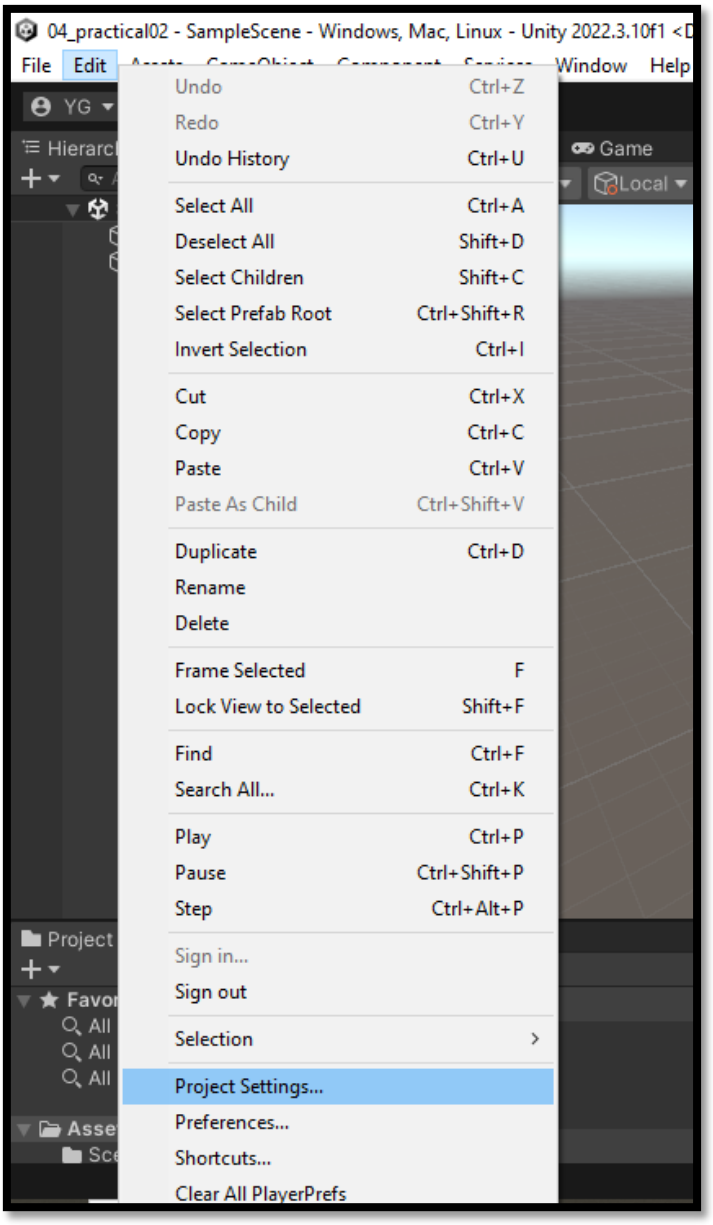
Practical No. 02

Aim: Implement XR simple Intractable in Unity using XR Interaction Toolkit.

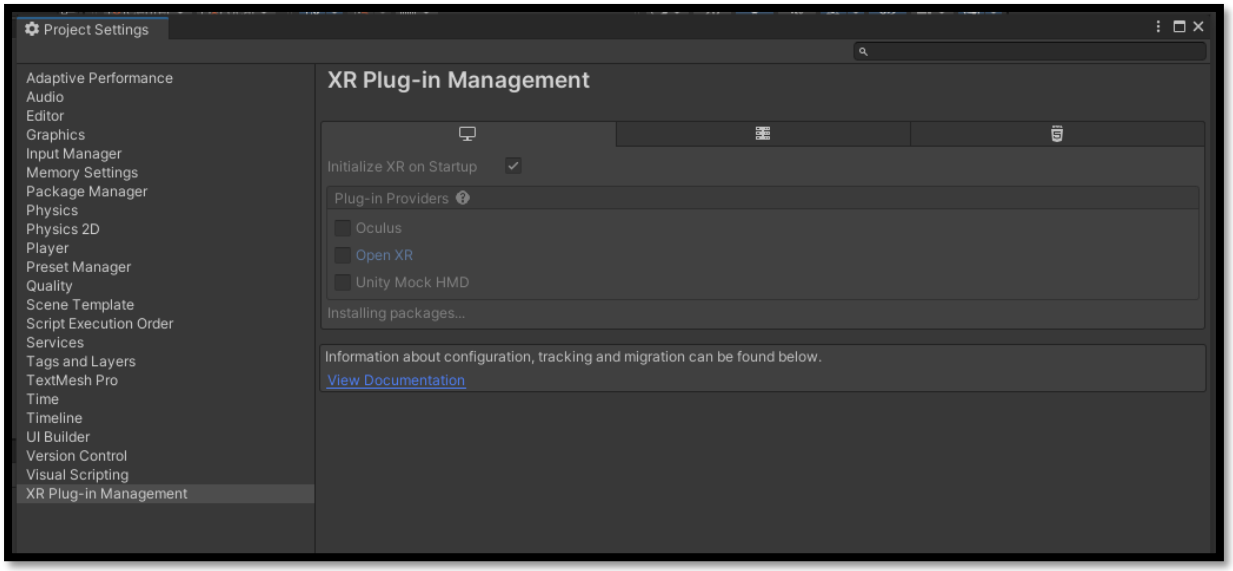
Step 1: Go to unity hub ➔Click on the New Project ➔ Click on the 3D and give the project name.

Step 2: Go to the Edit Option click on the project setting.

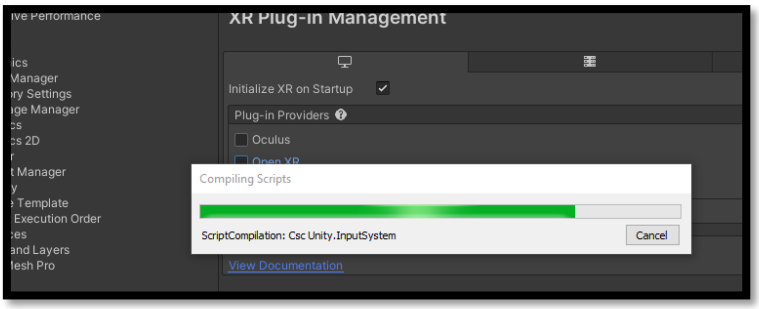


Step 3: Go to last option XR Plugin Management and click on Install.

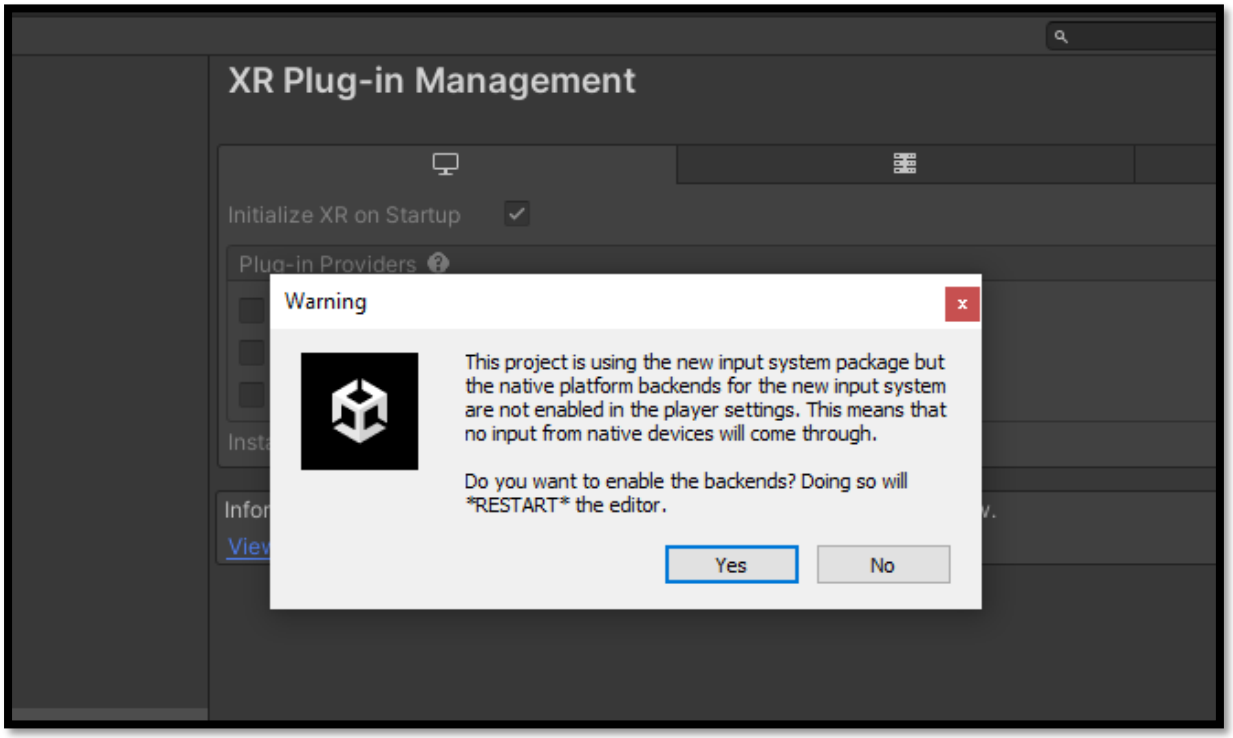
Step 4: Click on the XR Plug-in Management and check the box Open XR.

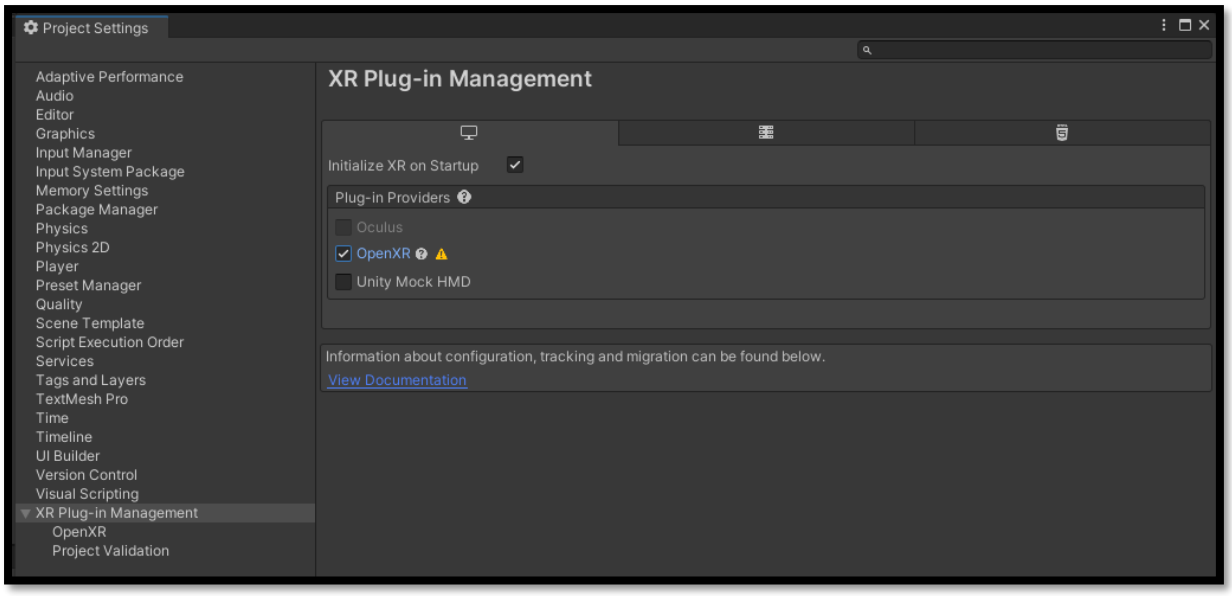


This will take little time and once done

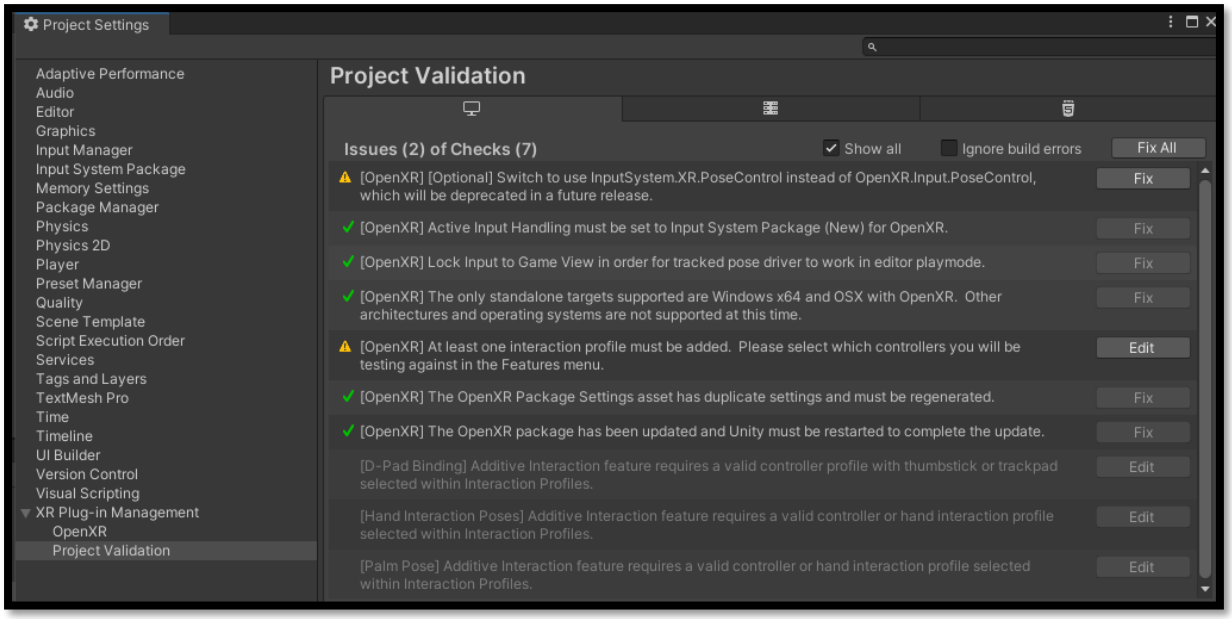


Step 5: You will be shown a warning dialogue box this is to implement all the preferences to your project in effect by restarting Unity ➞Here click on Yes and your Unity will restart.



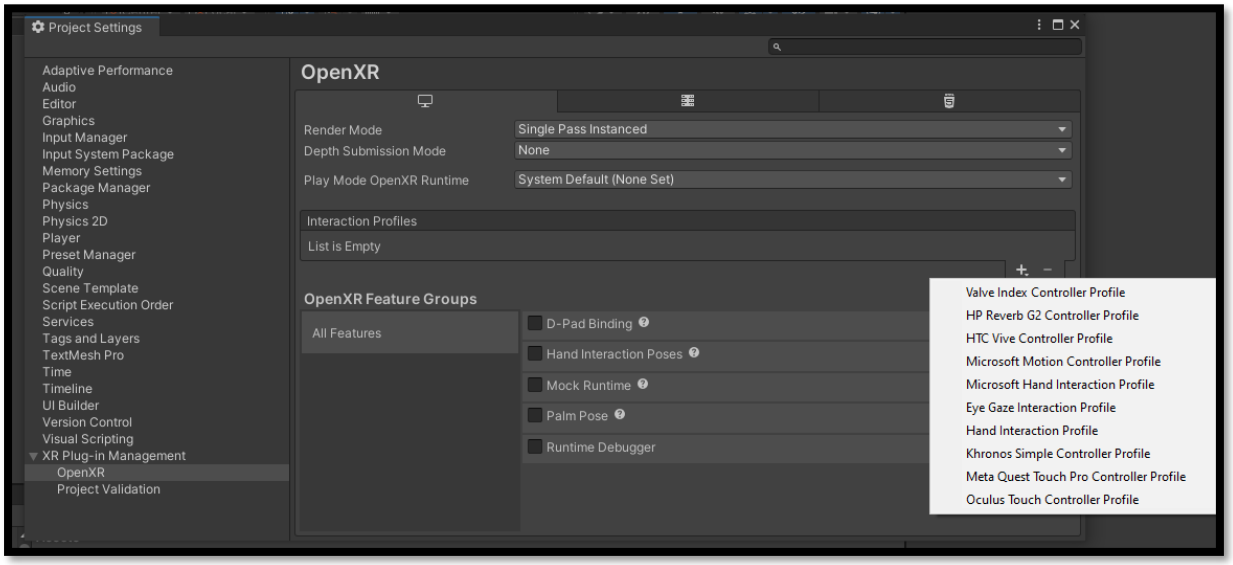


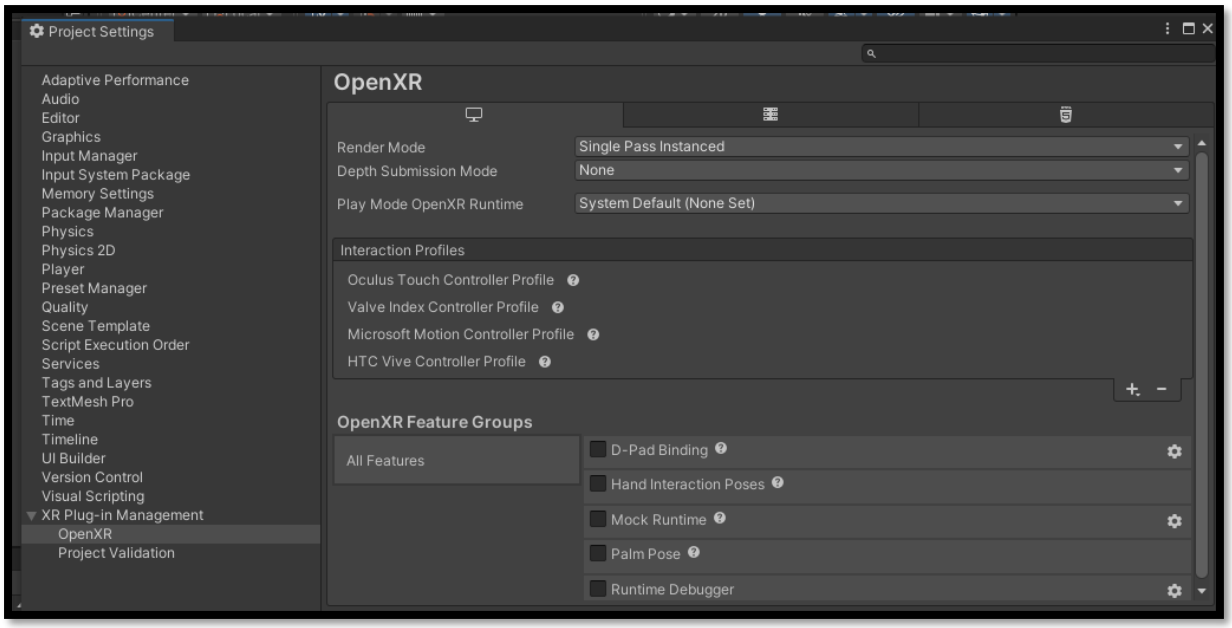
Step 6: Once Unity restarts ⬇ screen will be present, for the first issue click on finish**.**



Step 7: While for the second one, Click on edit, and the below dialogue box will appear *[The reason you are getting this issue is that Unity wants to know which interaction VR profile you will use use for testing your game]*

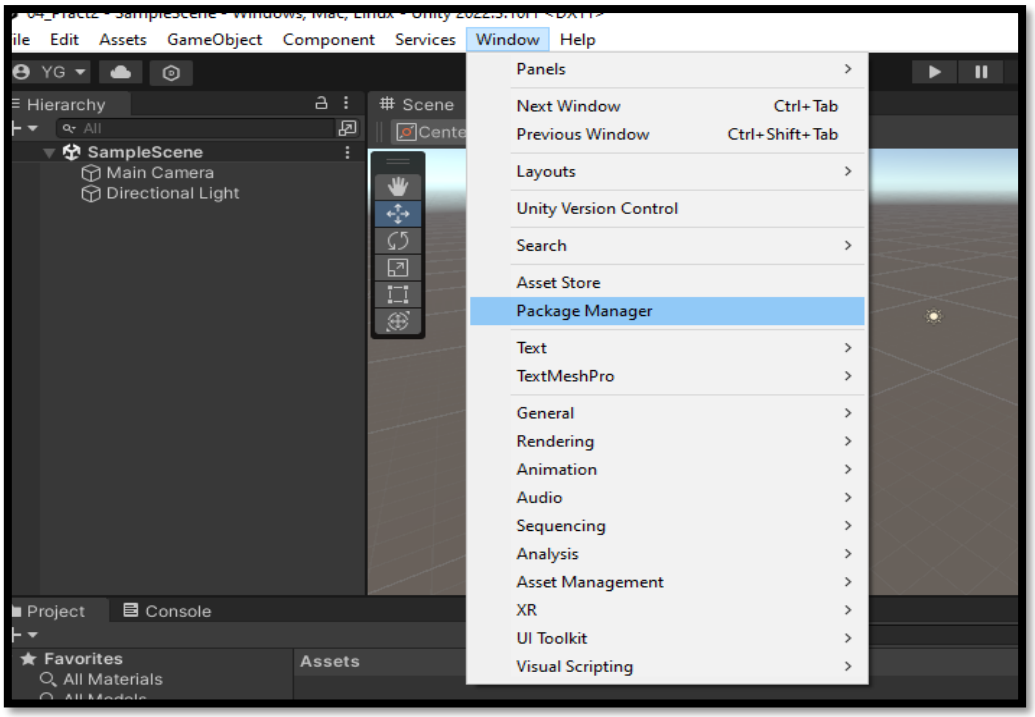
Step 8: In above dialogue box screen click on the + sign to add Interaction Profiles.



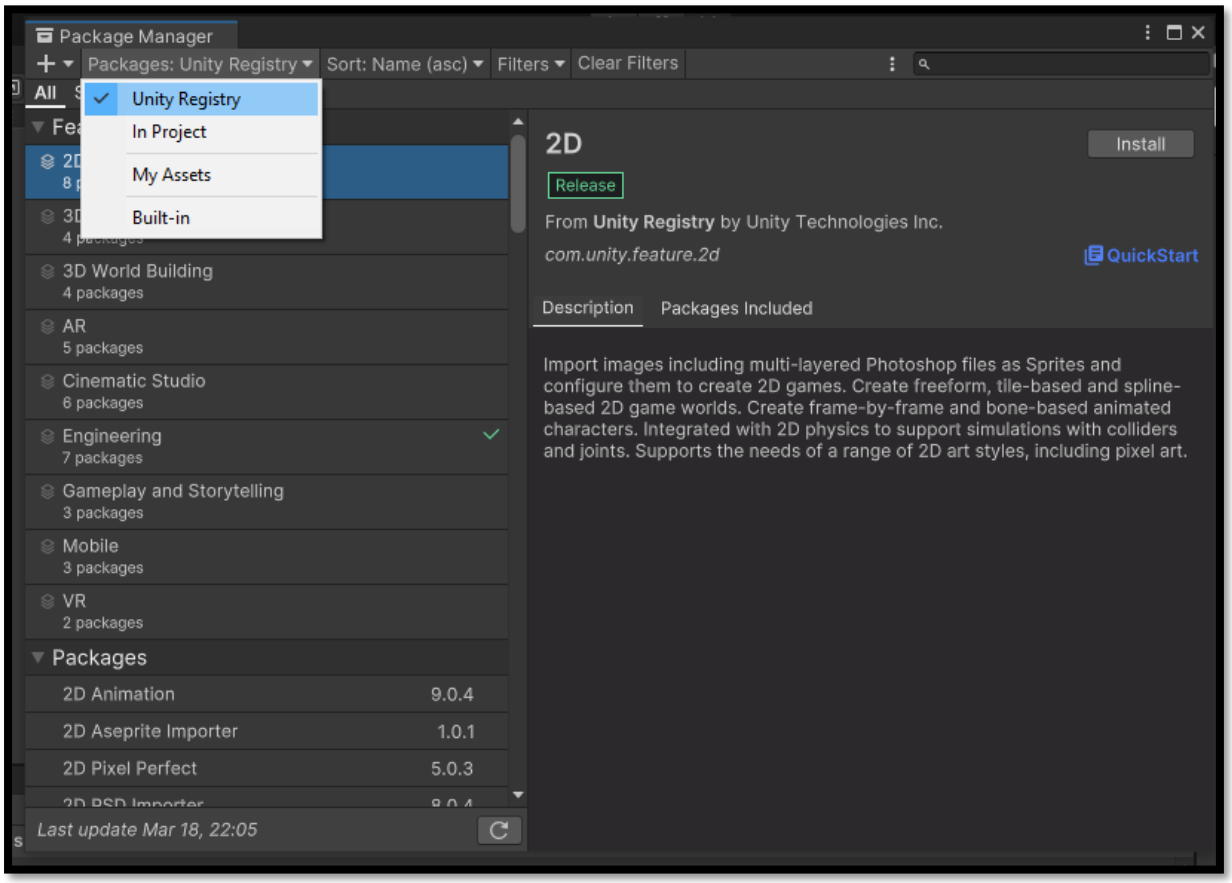


Step 9: For Adding XR Interactable Toolkit

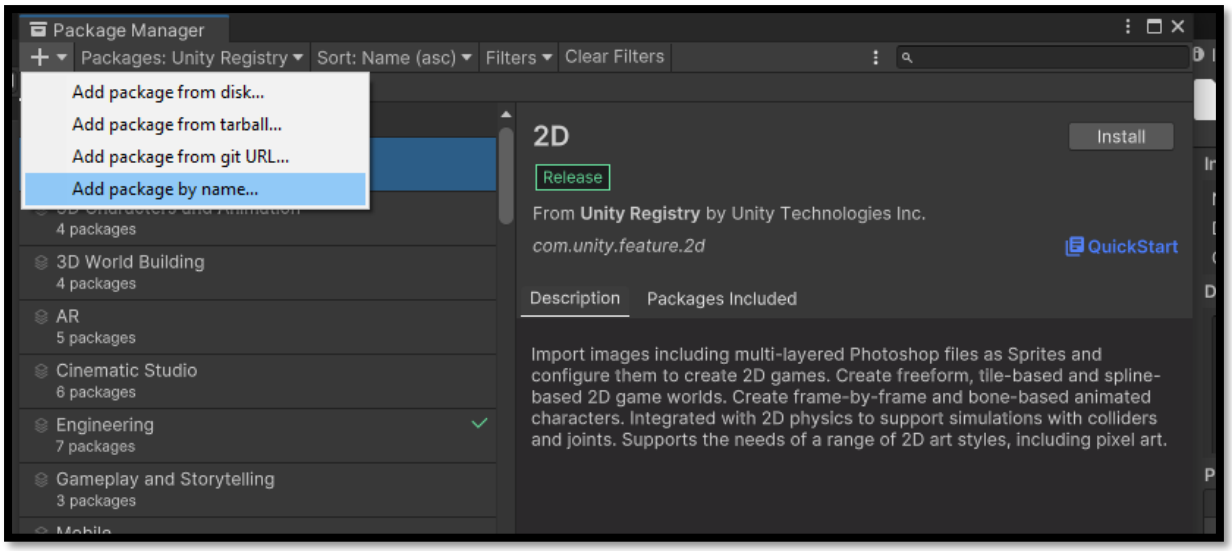
Go to Window ➞Click on Package Manager ➞



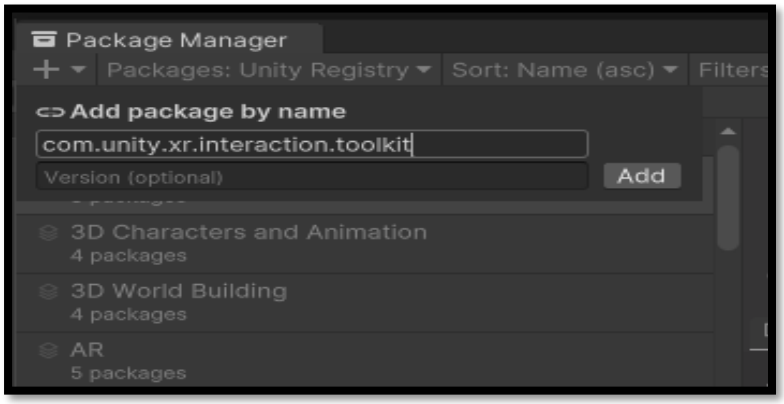
Step 10: In the ⬇ screen click on Package: In project ➞ select Unity Registry

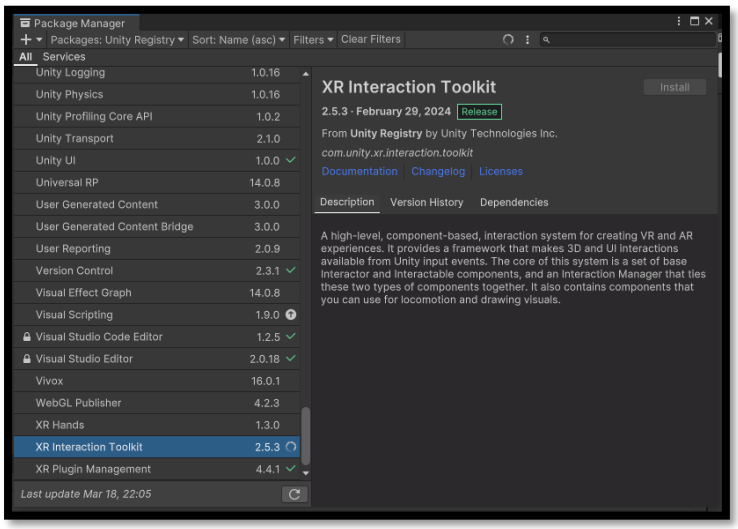


Step 11: Click on ✚ icon and click on Add package by name.

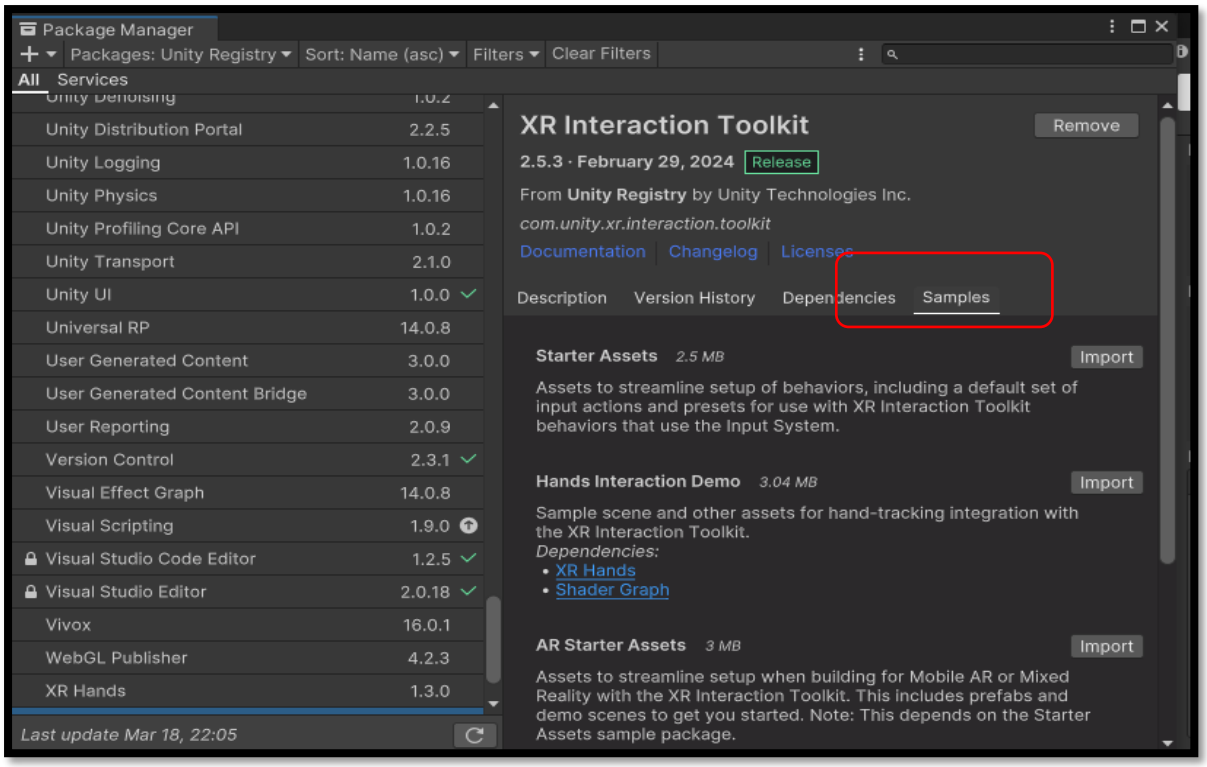


Type com.unity.xr.interaction.toolkit and hit enter (this way will help you install latest version of XR interaction Toolkit).

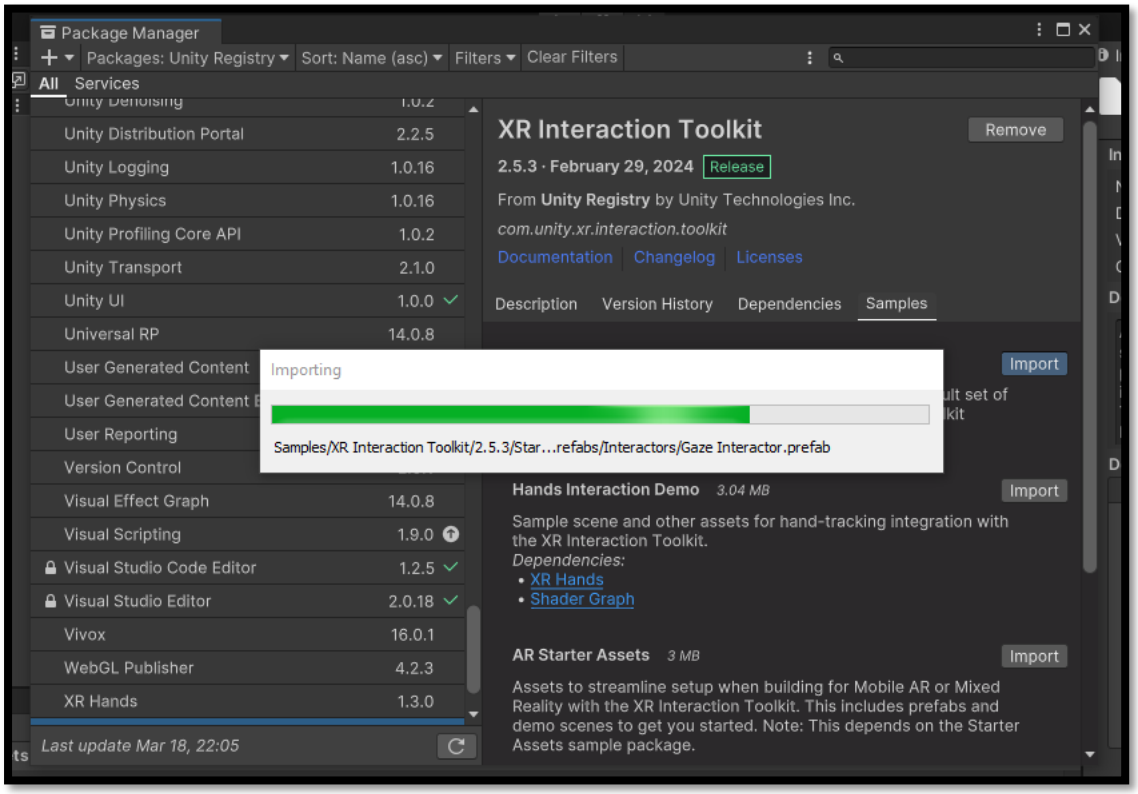


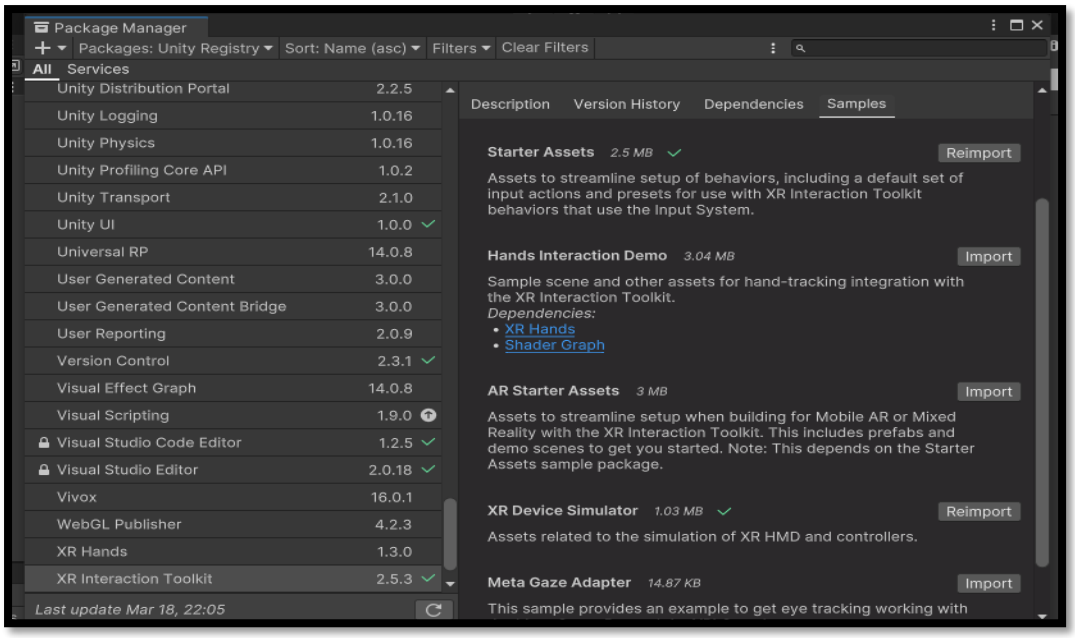


Step 12: After installation is, done ⬇ screen will appear click on Samples.

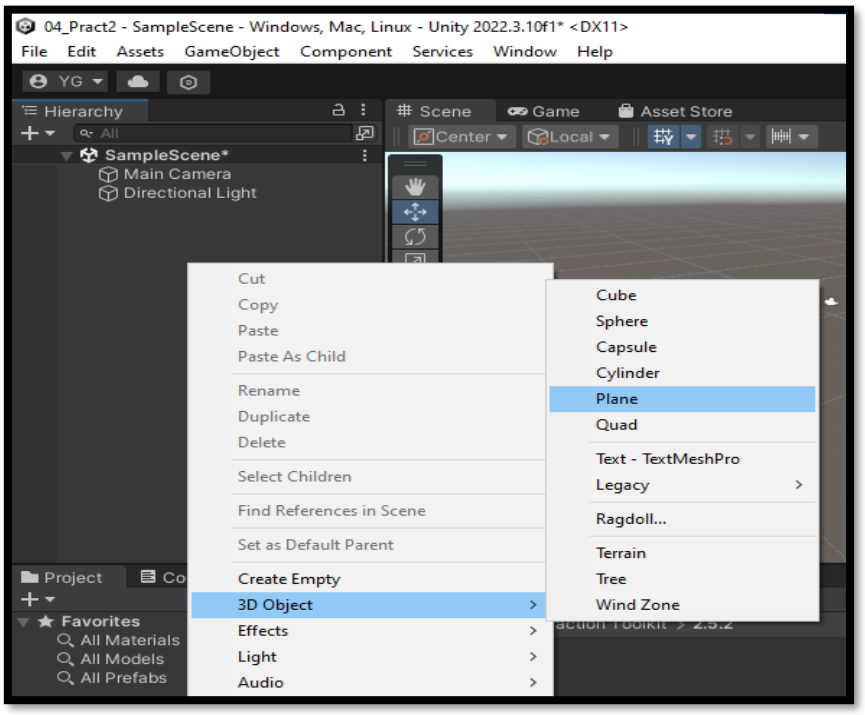


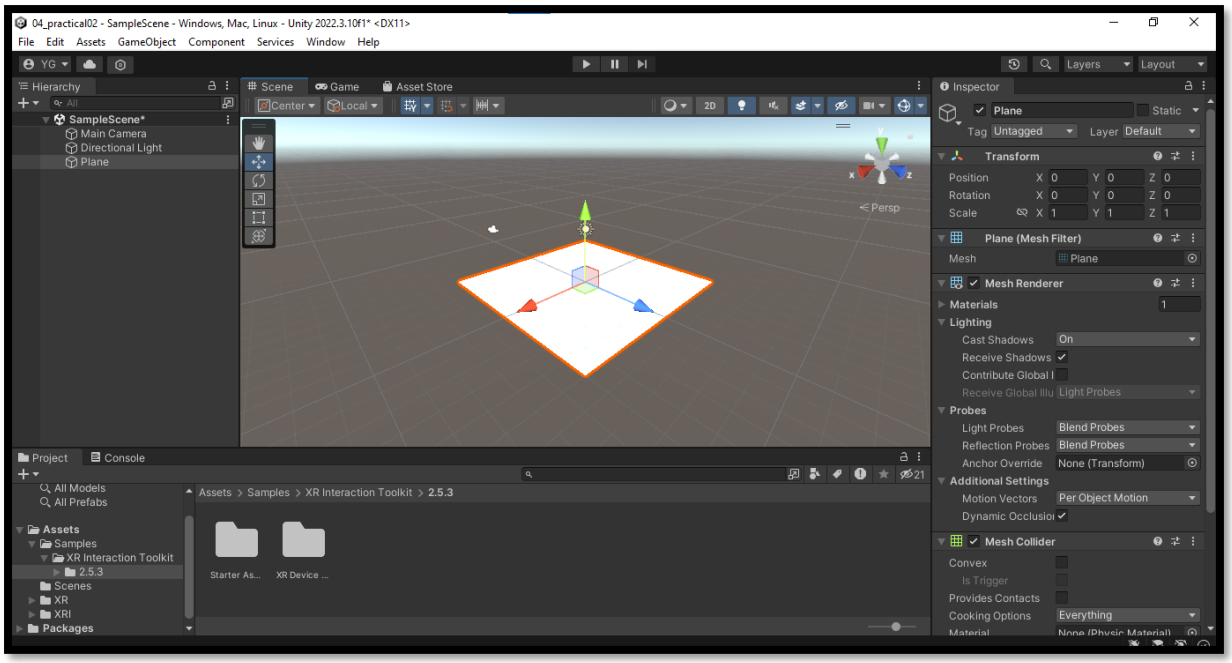
Step 13: Click on Import for Starter Asset and XR Device Simulator (We are using XR simulator since we cannot have our actual VR profile with us for testing purposes thus this simulator will help us in the job of testing our VR game)





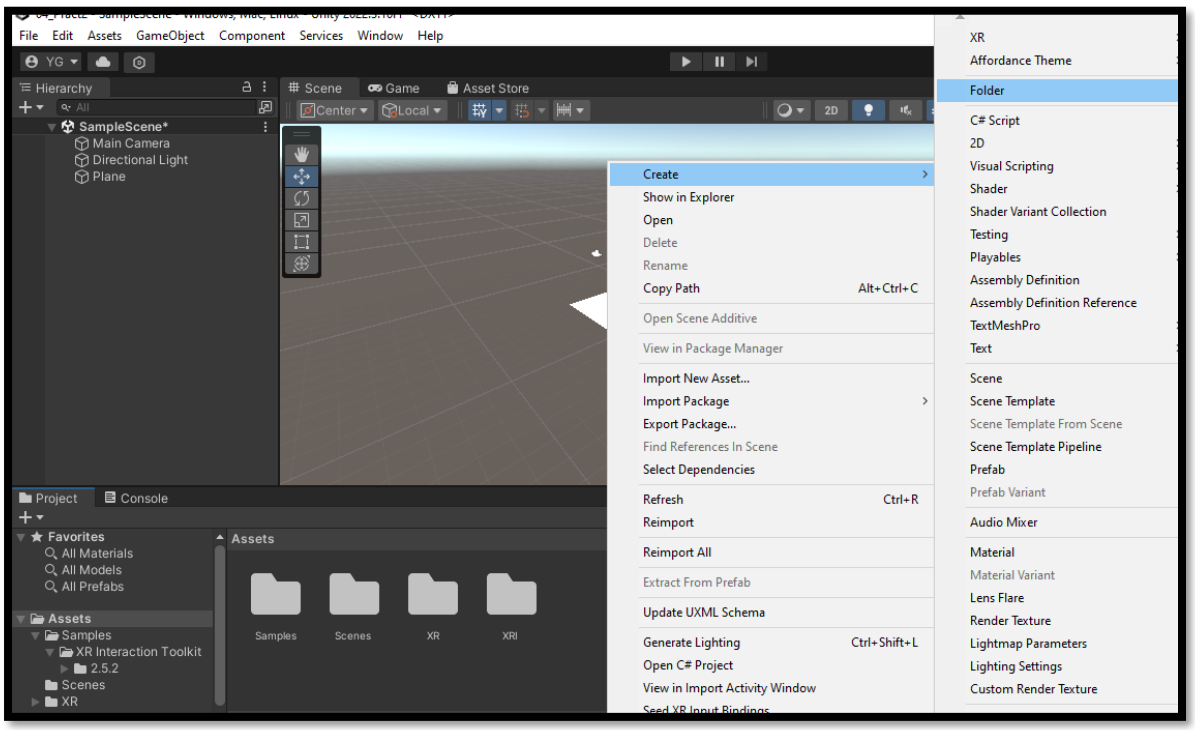
Step 14: Right Click on the Hierarchy area and click on 3D Object ➞ Click on Plane.

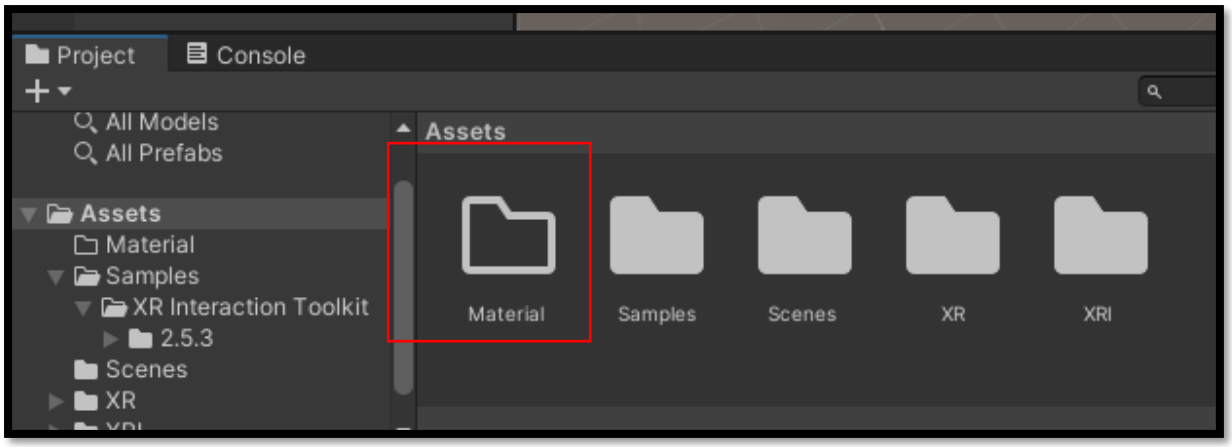




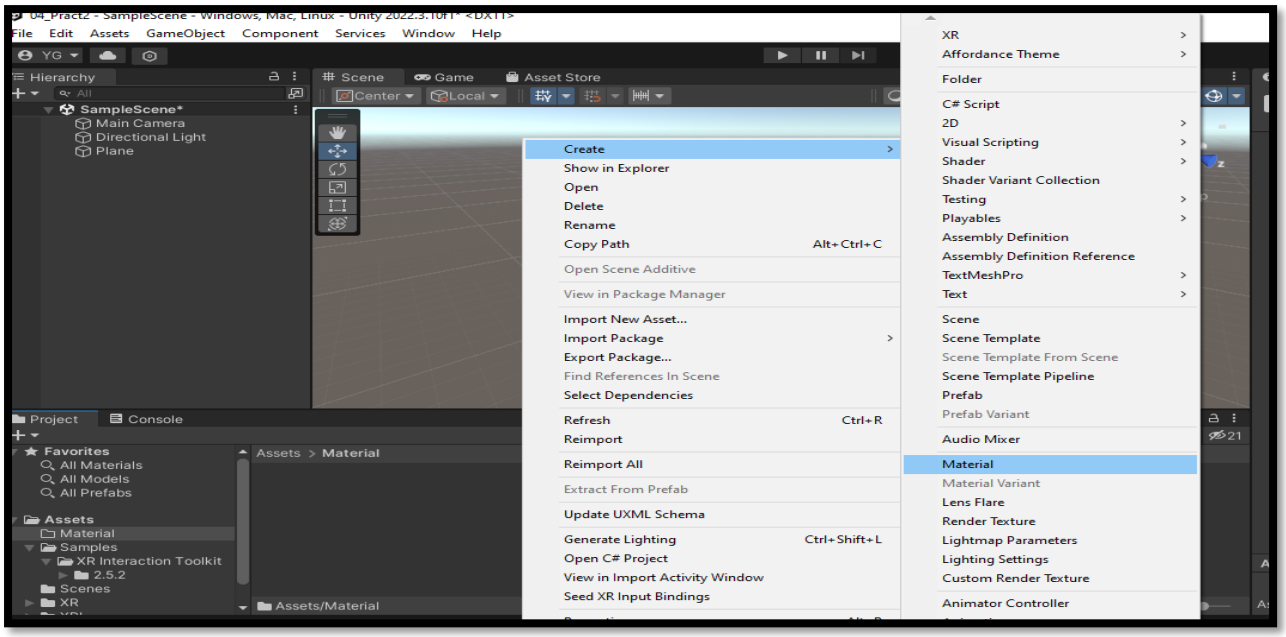
Step 15: For adding color to the plane we are using Material for that to keep things simple and structured, we will create folder called Material in our asset

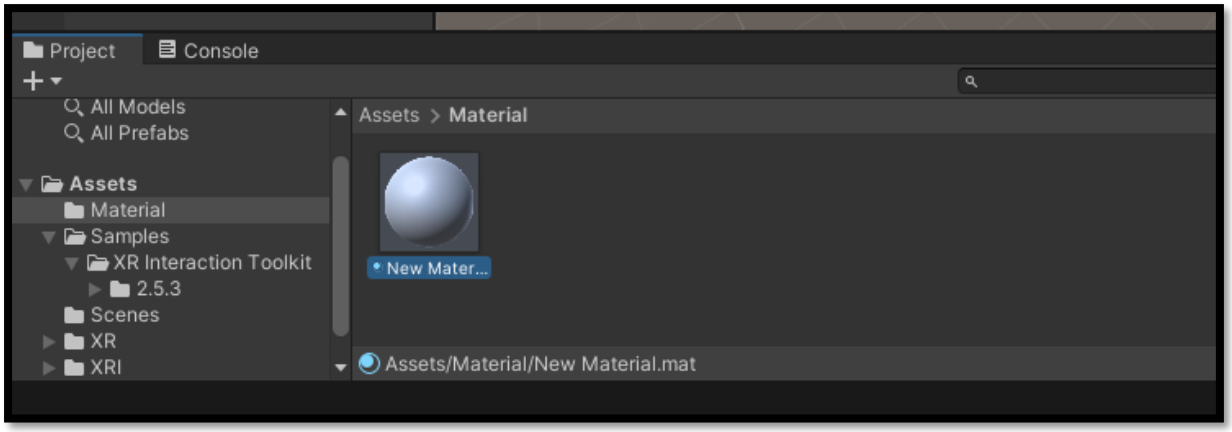
So to do that right click on Assets area ➞ click on Create ➞ Click on Folder ➞ rename your New Folder to Material.





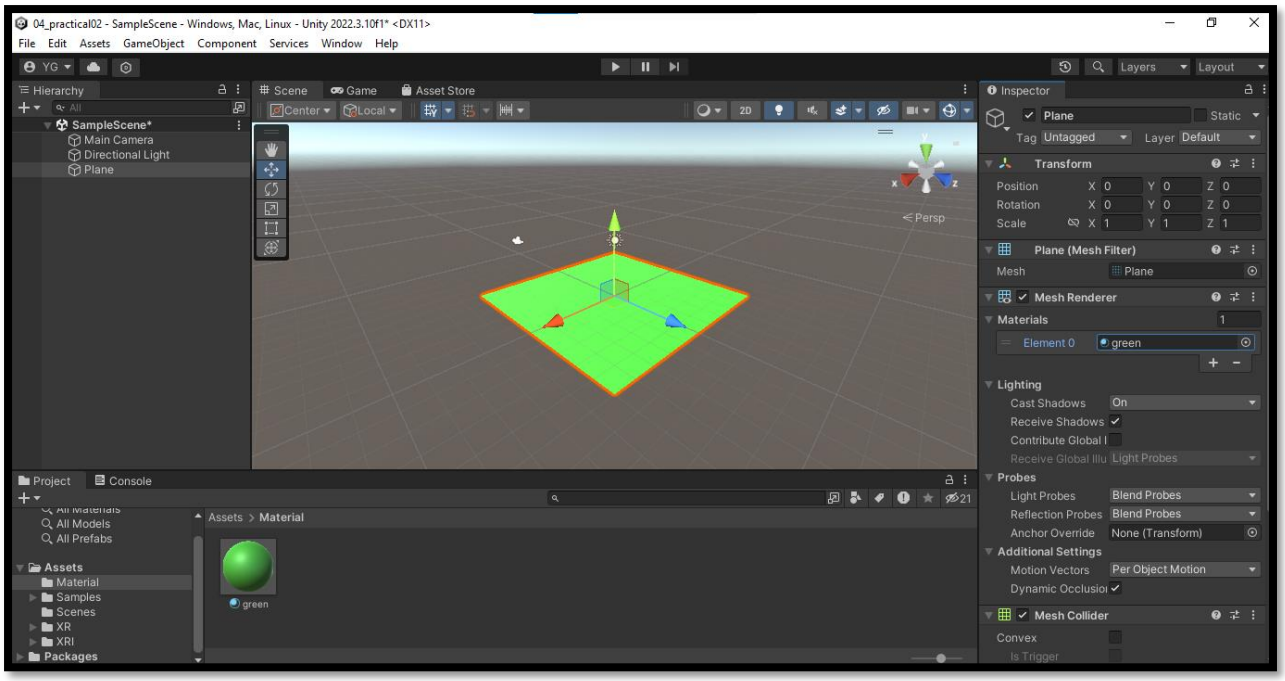
Step 16: Click on Material Folder ➞Inside Material folder ➞ right click ➞click on Create ➞Click on Material.



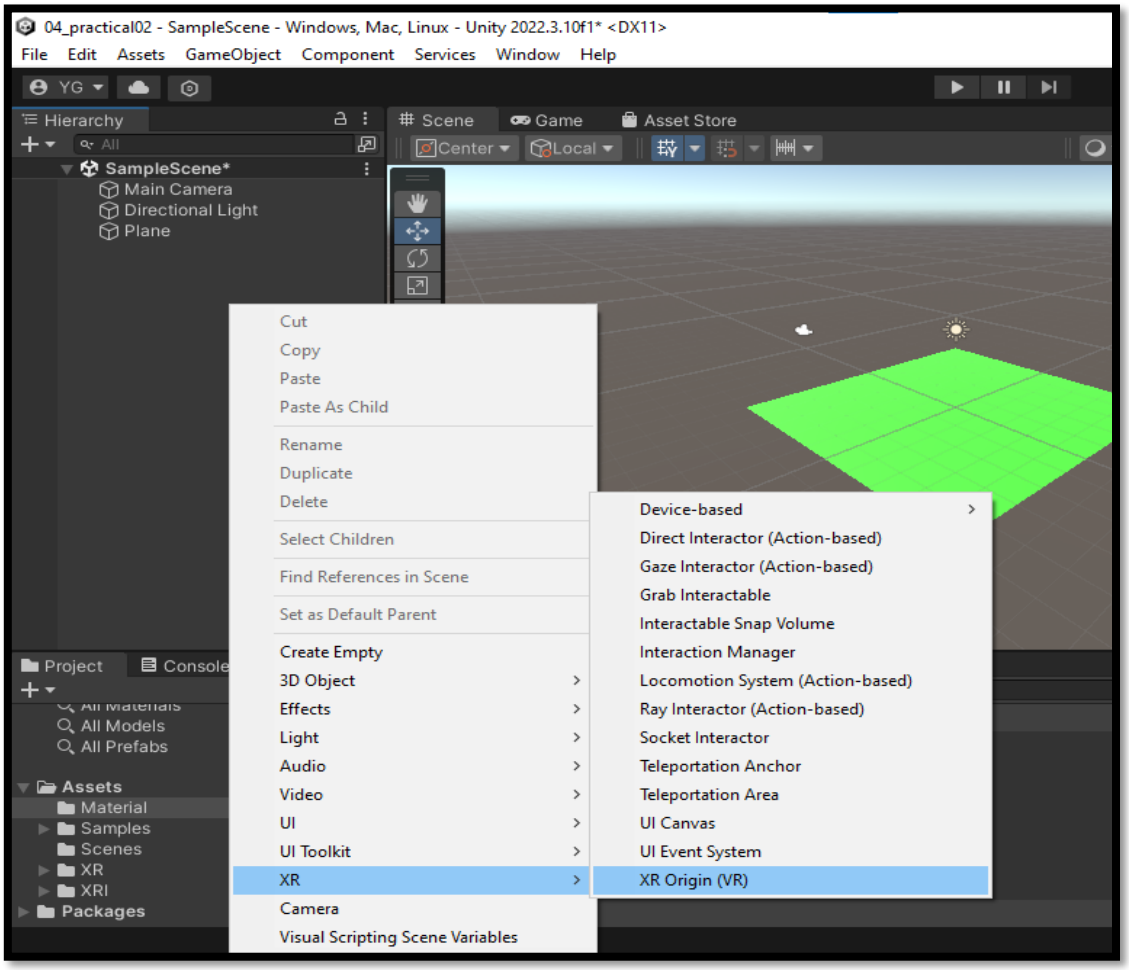


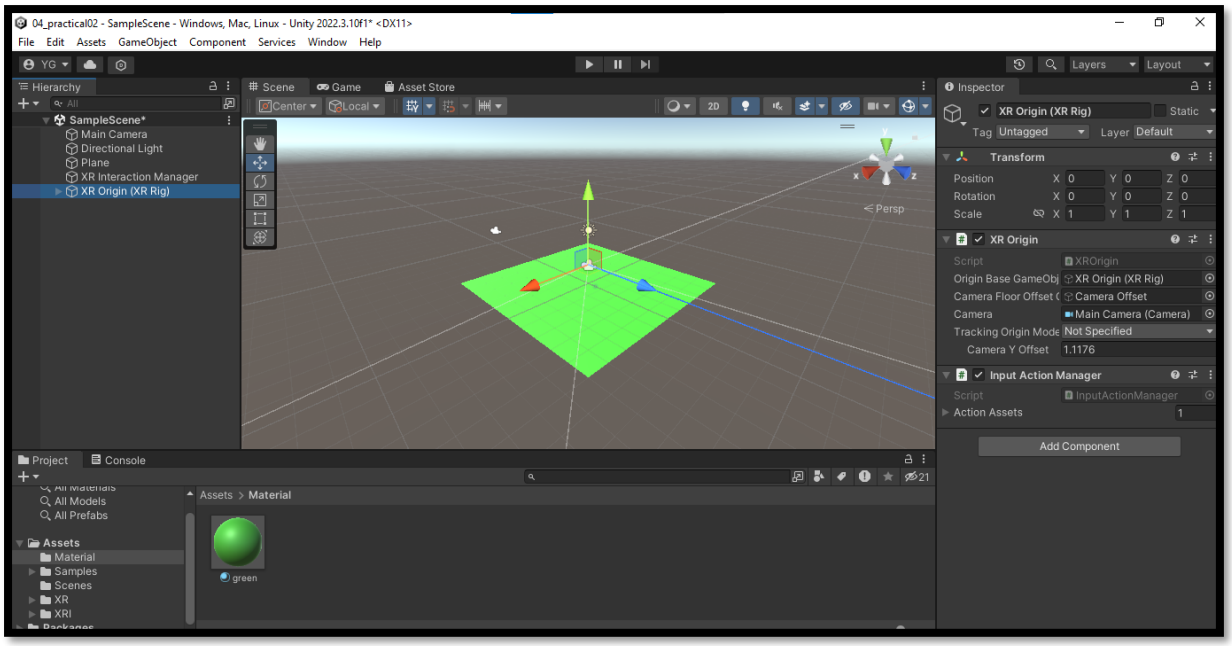
Step 17: Rename the name of material with your desired name, On Inspector panel click on Albedo to change color to desired one.

Step 18: Once done click on to plane game object➞on inspector panel click on material➞click on dropdown and select material that you created.

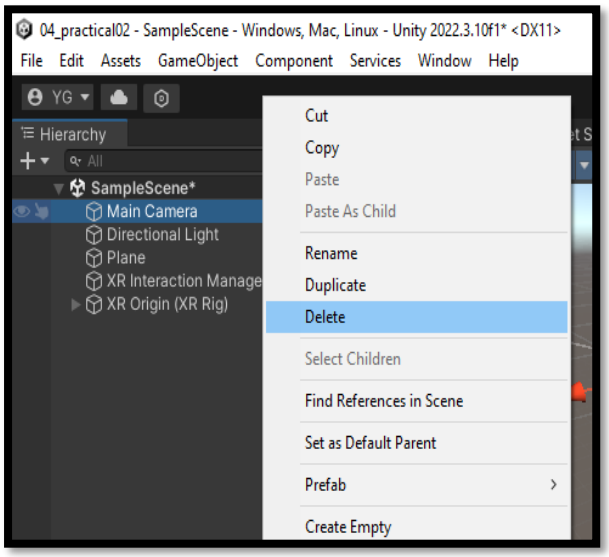


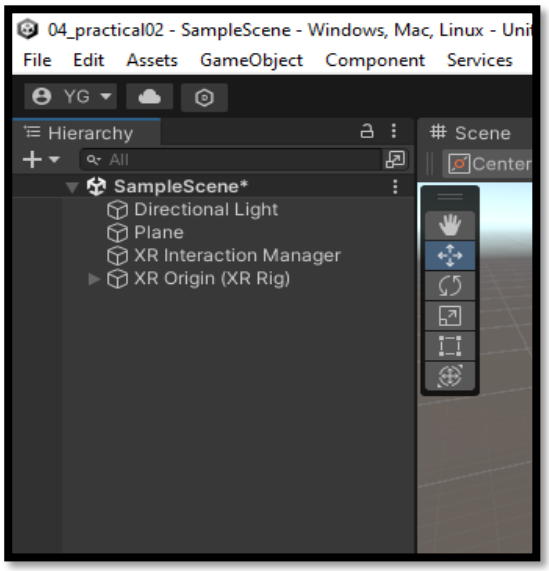
Step 19: Lets add our XR origin before adding more game object on scene Right click on Hierarchy panel➞Click on XR➞Click on XR Origin (VR)⬇



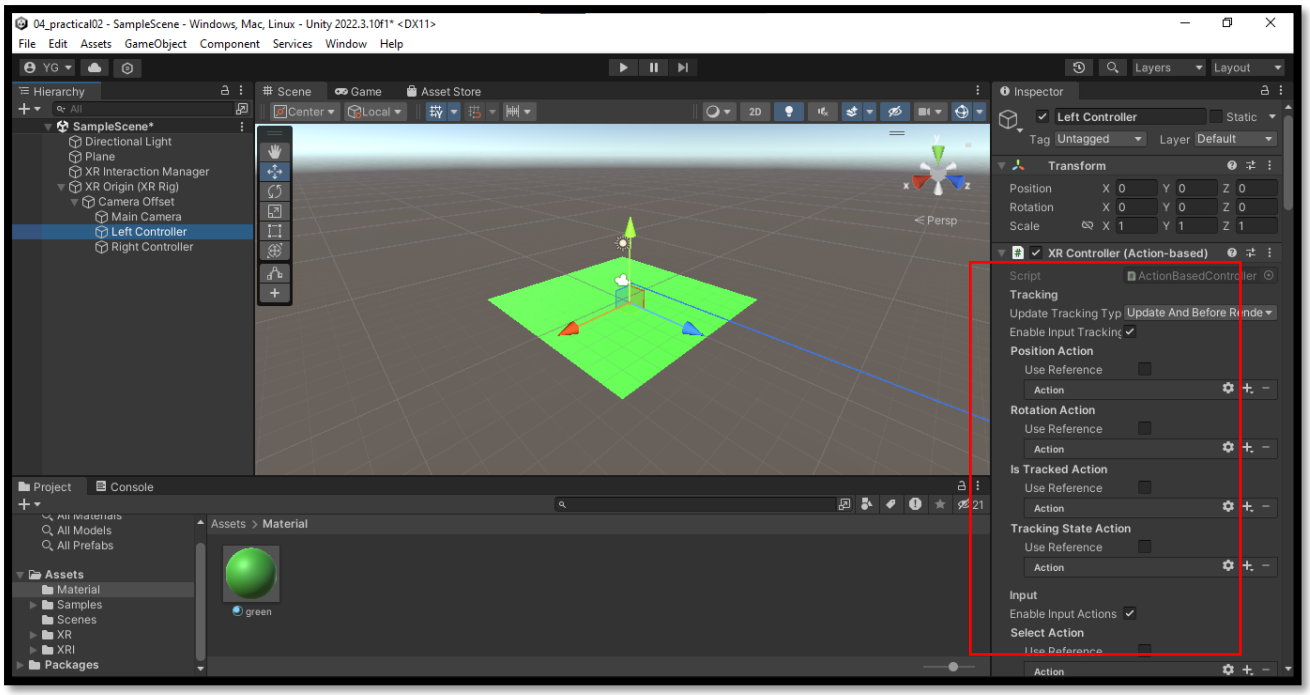


Step 20: Delete Main Camera above Directional light➞Right click on Main Camera➞Click on Delete
(Since we already have Main camera in XR origin having both camera's would clash and the Simulation won't provide proper result)

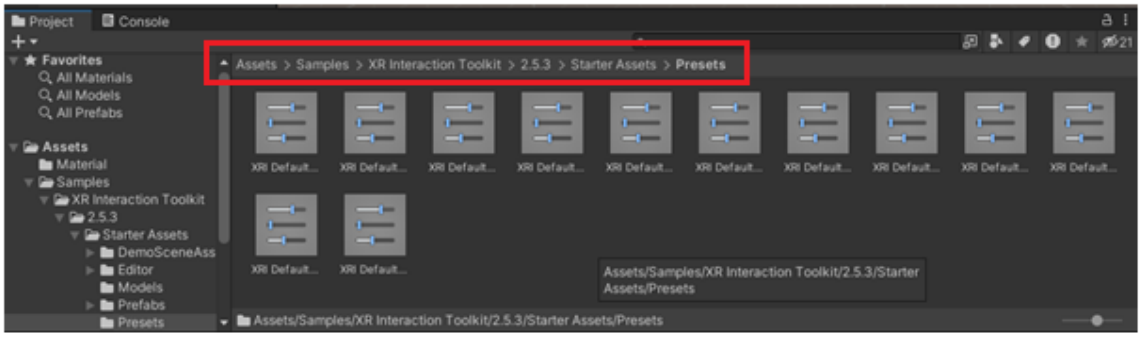




Step 21: When you get inside XR Origin folder➞Camera Offset➞Left Controller the Inspector panel shows controller option however for left controller it should show XRI Left but here it doesn't show the same thus for this lets make some settings so that Left and Right controllers have proper attribute values assigned to it.

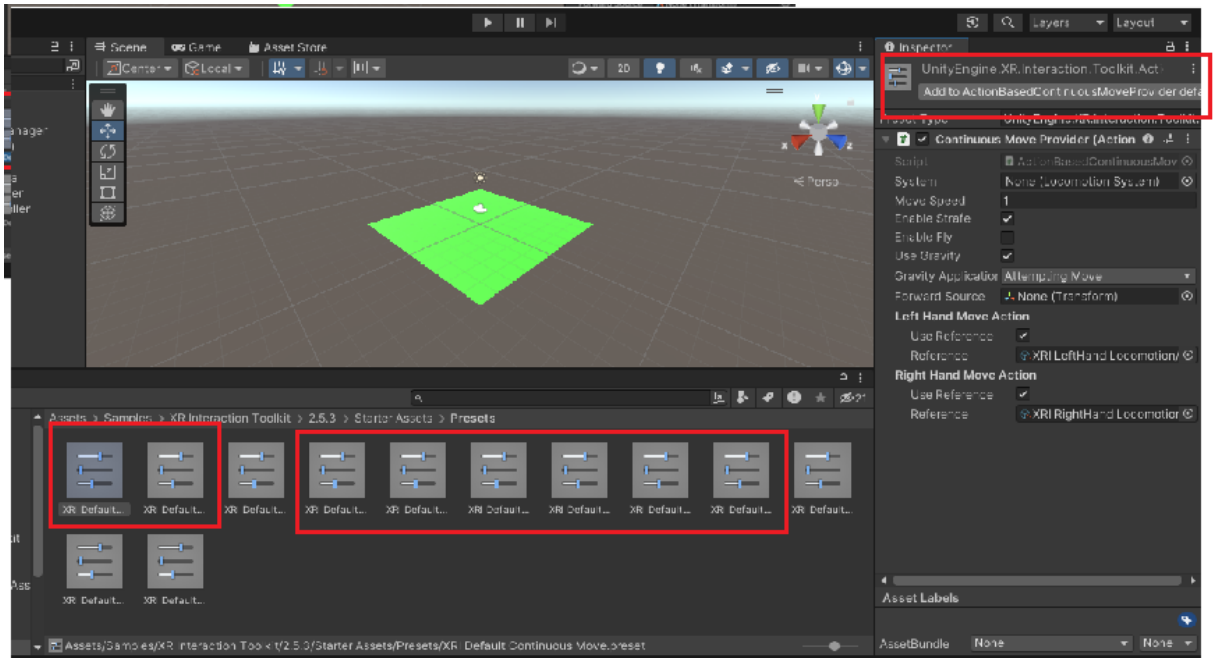


Step 22: First delete XR Origin from Hierarchy panel➞Go inside Samples folder➞XR Interaction Toolkit➞2.5.2➞Starter Assets➞Presents.

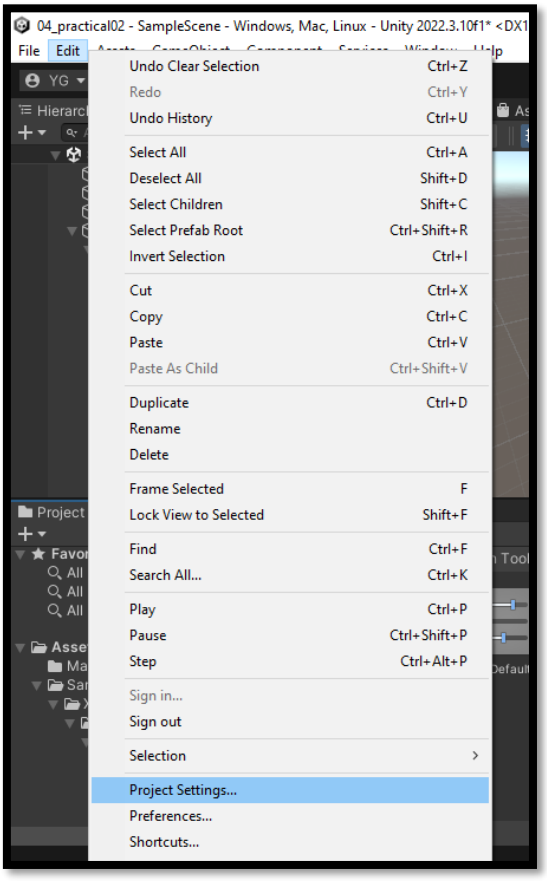


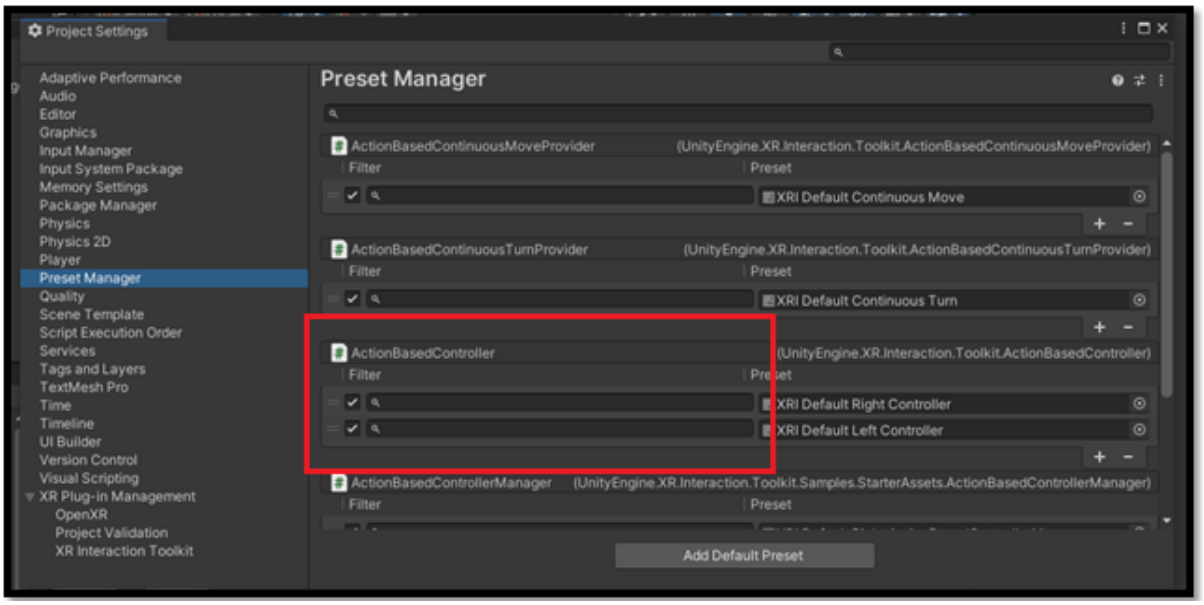
Step 23: Click on⬇marked presets➡click on Add to ActionBasedController do same job for rest of the marked presets.

-

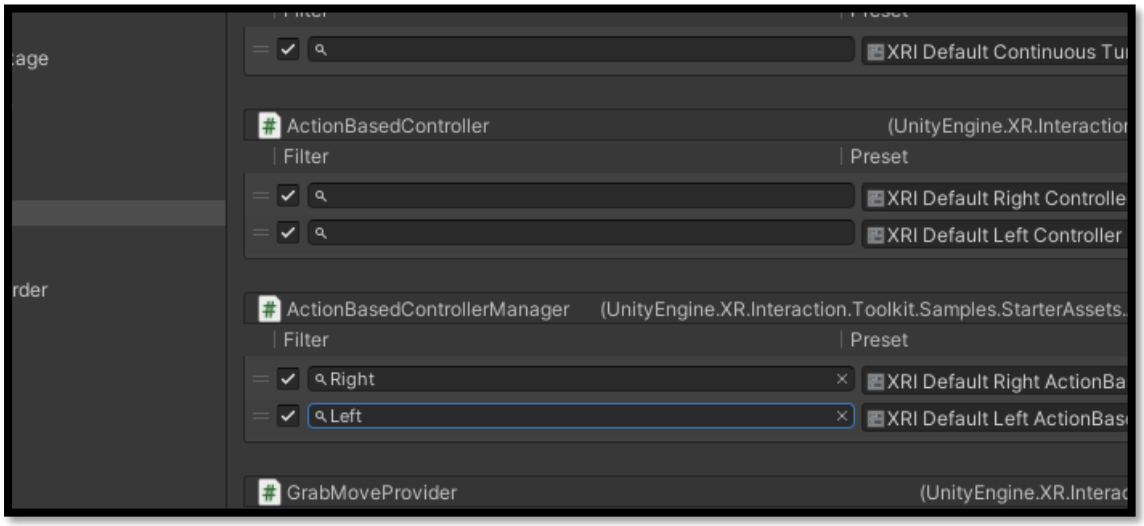


Step 24: Once added for marked Presets Go to Edit➡Project Settings➡Click on Presets manager.

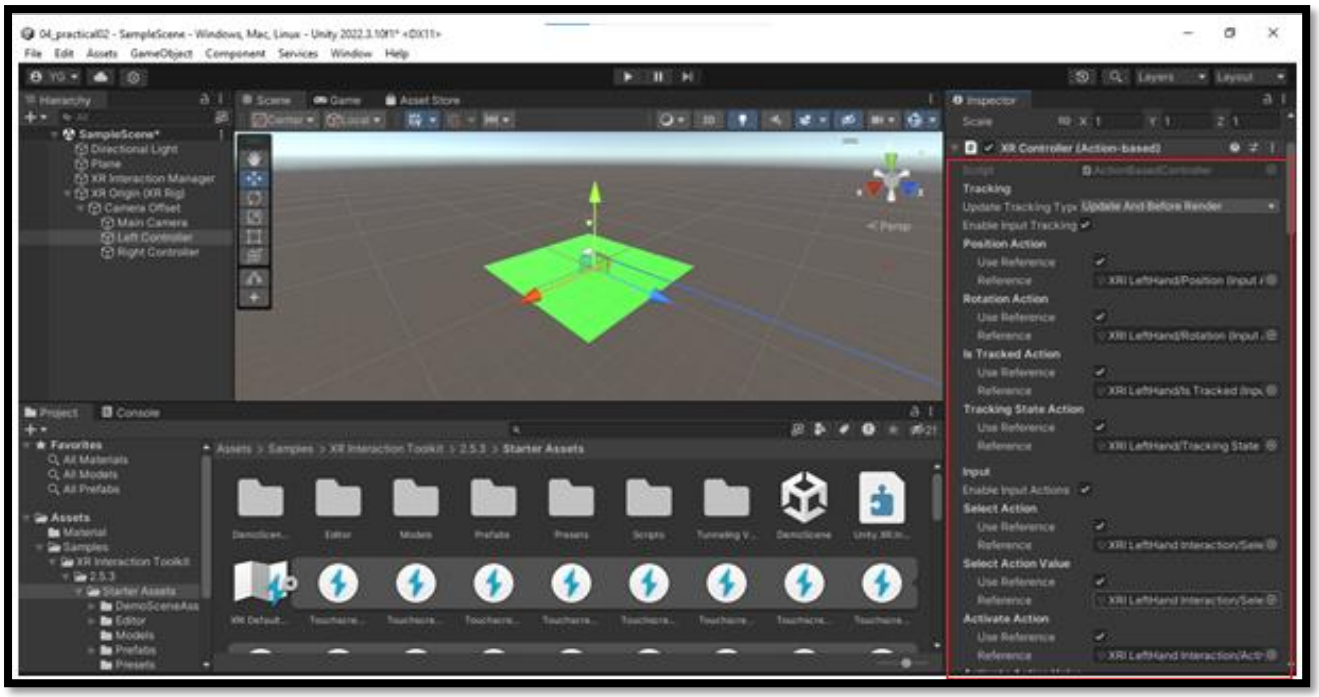




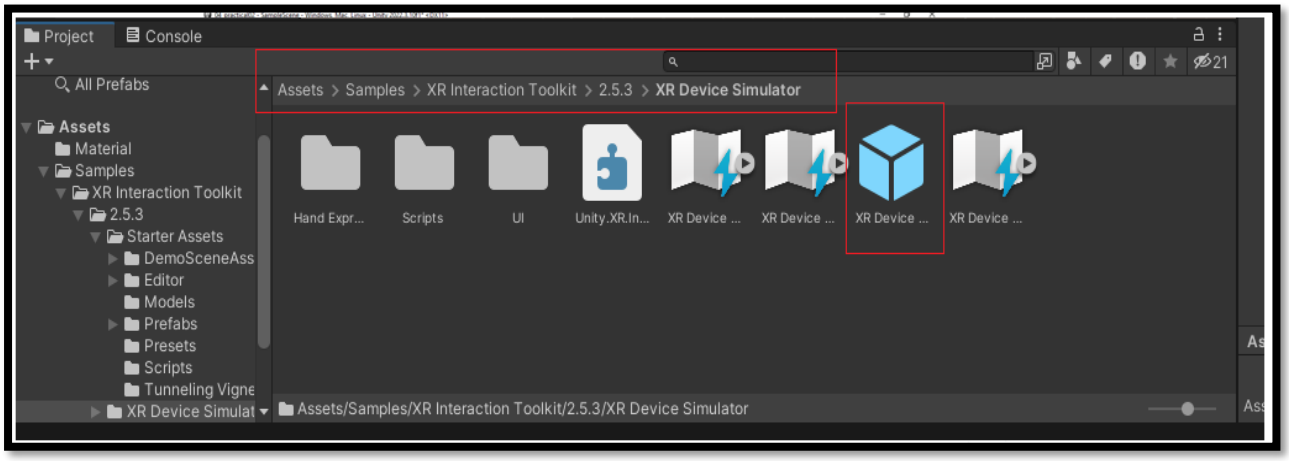
Step 25: As you can see fields are blank to the left of Preset XRI Default Right Controller and XRI Default Left Controller➞In this field type manually right and left for the respective controller.⬇and close this dialogue box.

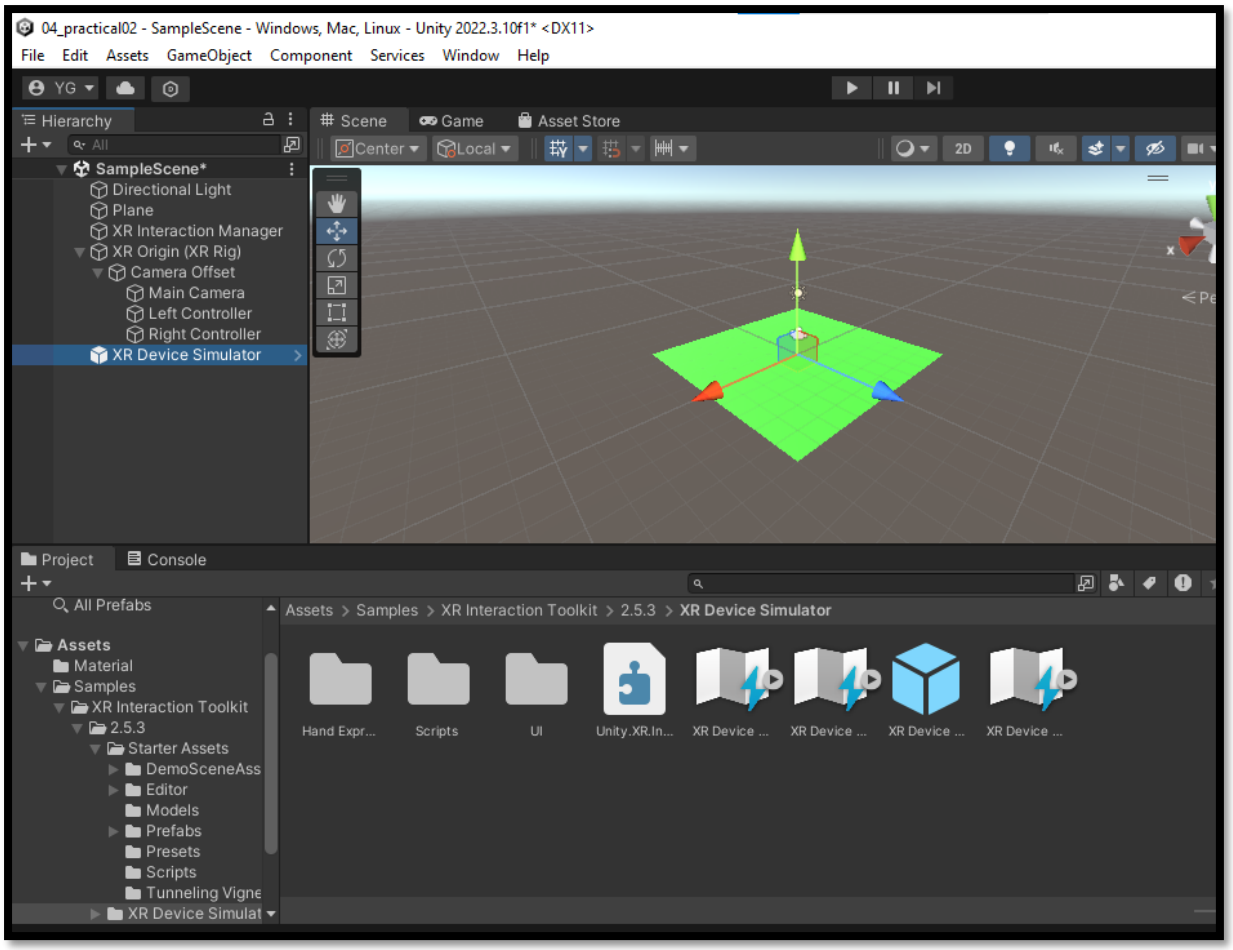


Step 26: Now again add XR Origin (VR) as in the earlier step [Right click on Hierarchy panel → click on XR → XR Origin (VR)]
Now go to left controller and check if Inspector panel appears same as that of below screen⬇.

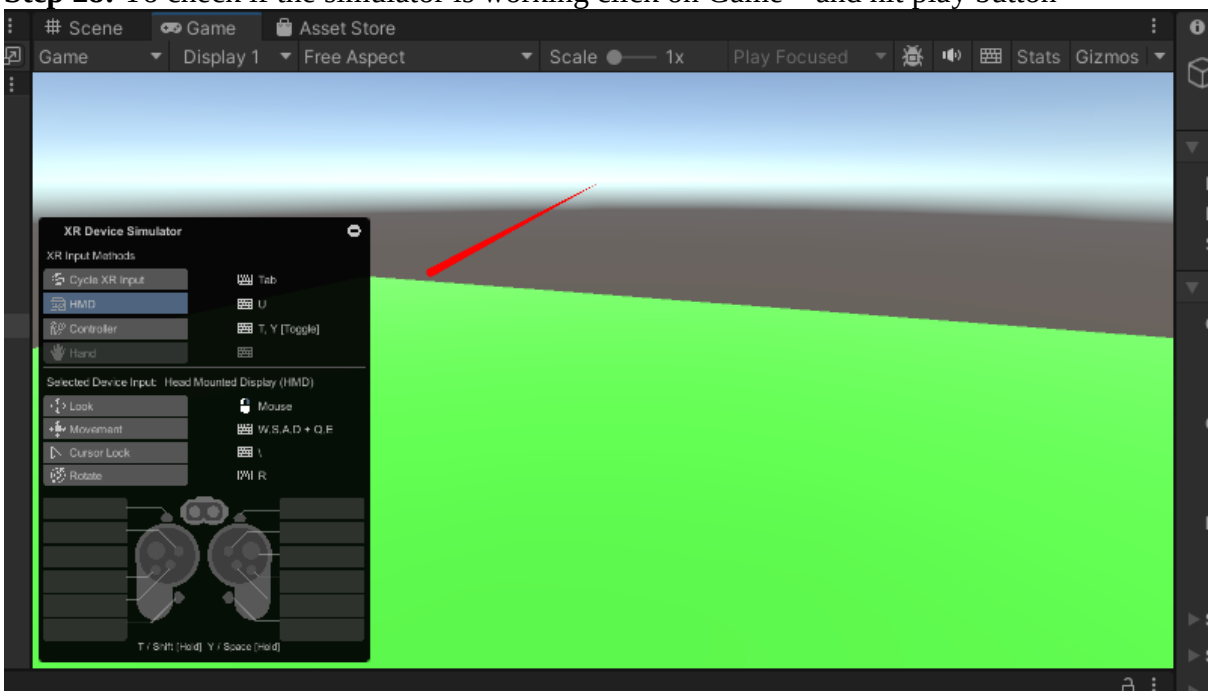


Step 27: Let us add our XR Device simulator as well For this go to Assets > Samples > XR Interaction Toolkit > 2.5.2 > XR Device Simulator > Click on XR Device Simulator > Drag and drop this prefab to Hierarchy Panel.



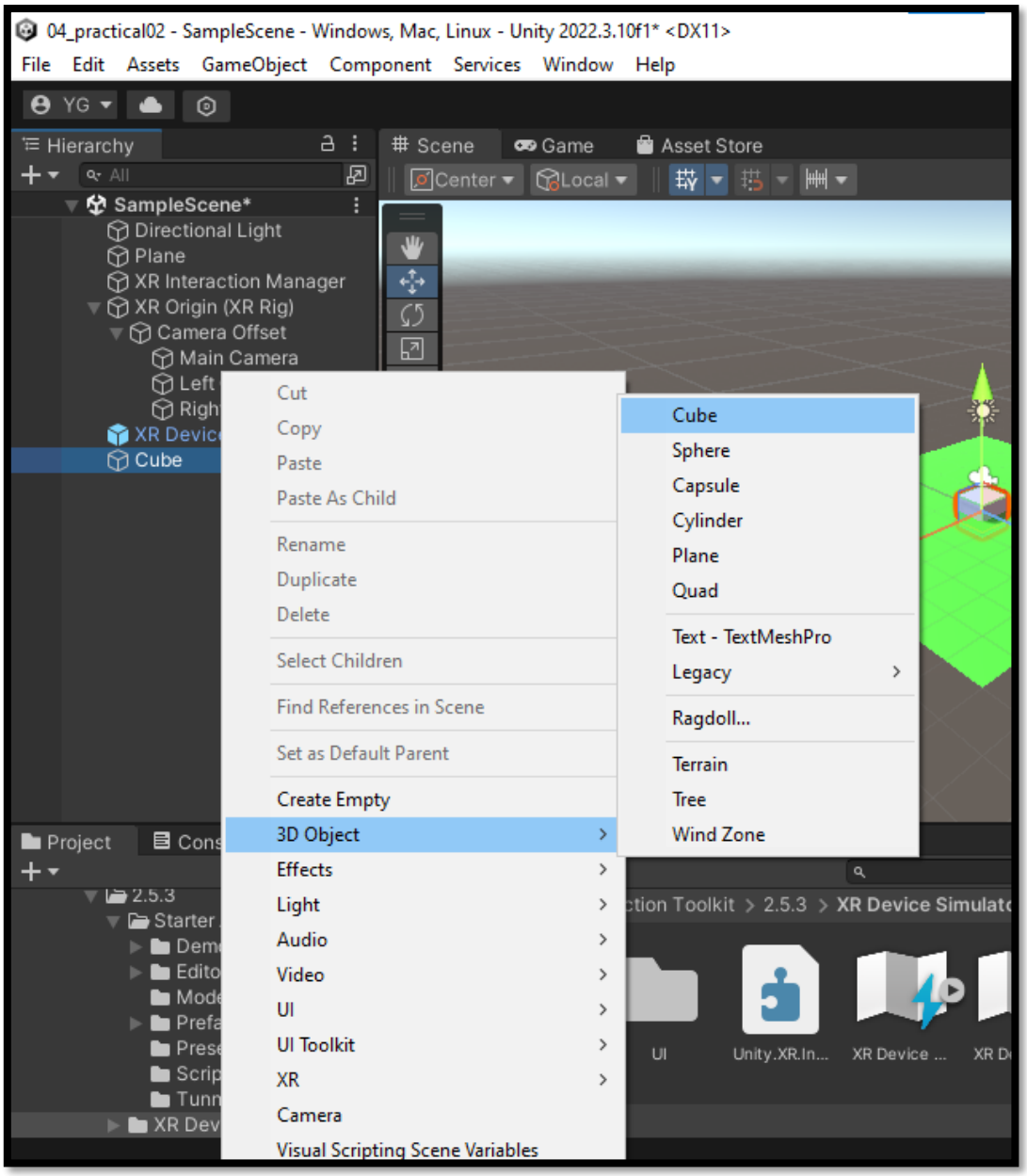


Step 28: To check if the simulator is working click on Game➞and hit play button

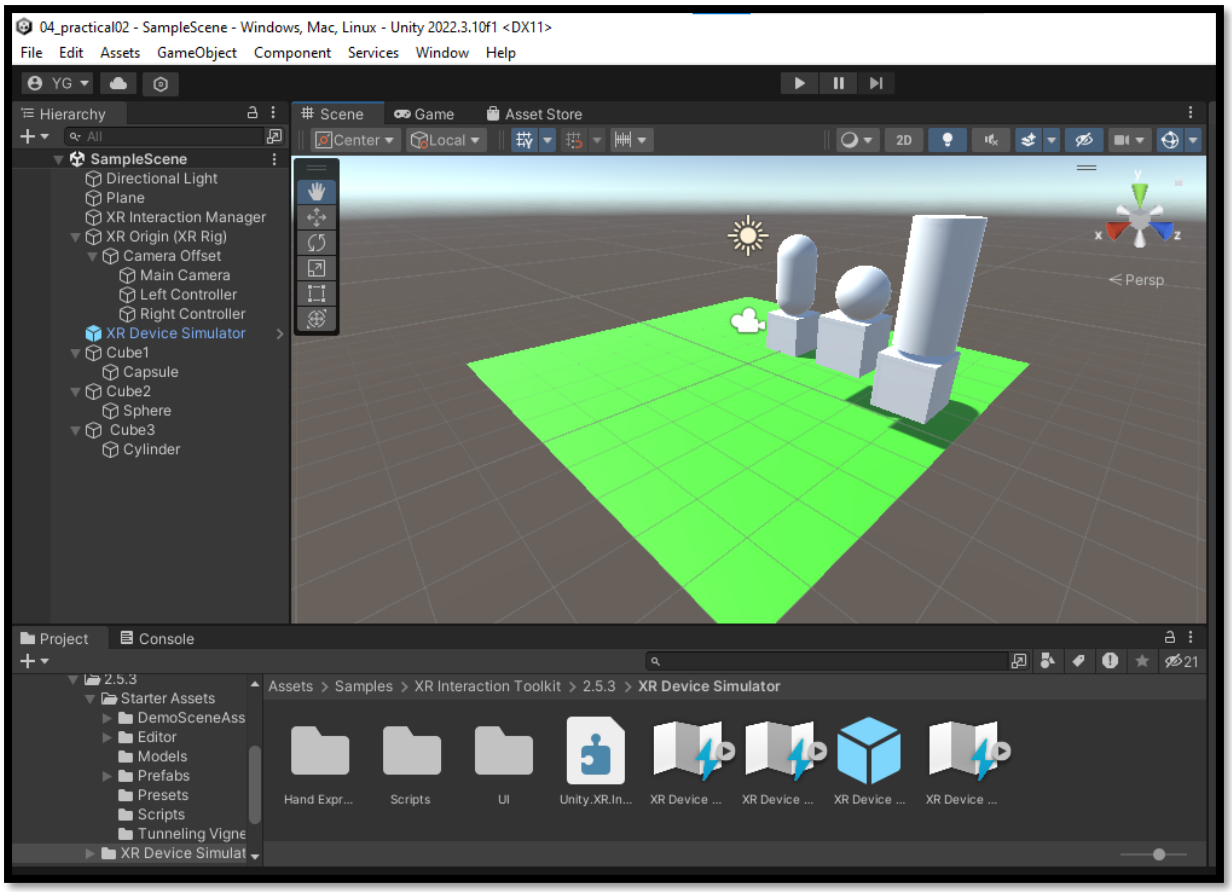


Since everything is working fine we are good to go with adding some more game objects in our scene [You can come out from game mode by clicking ESC button on keyboard]

Step 29: Let's add 3 cube game object in our scene

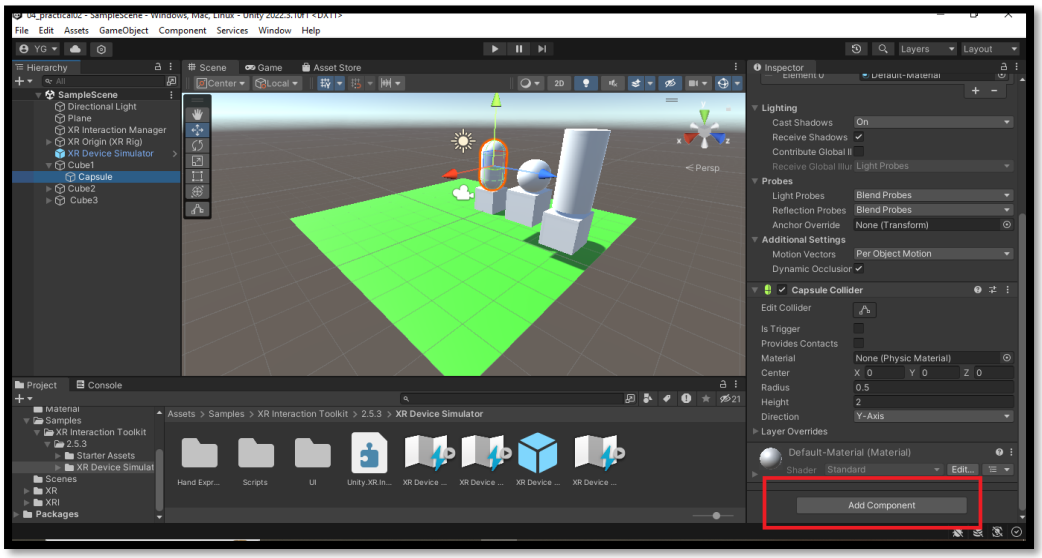


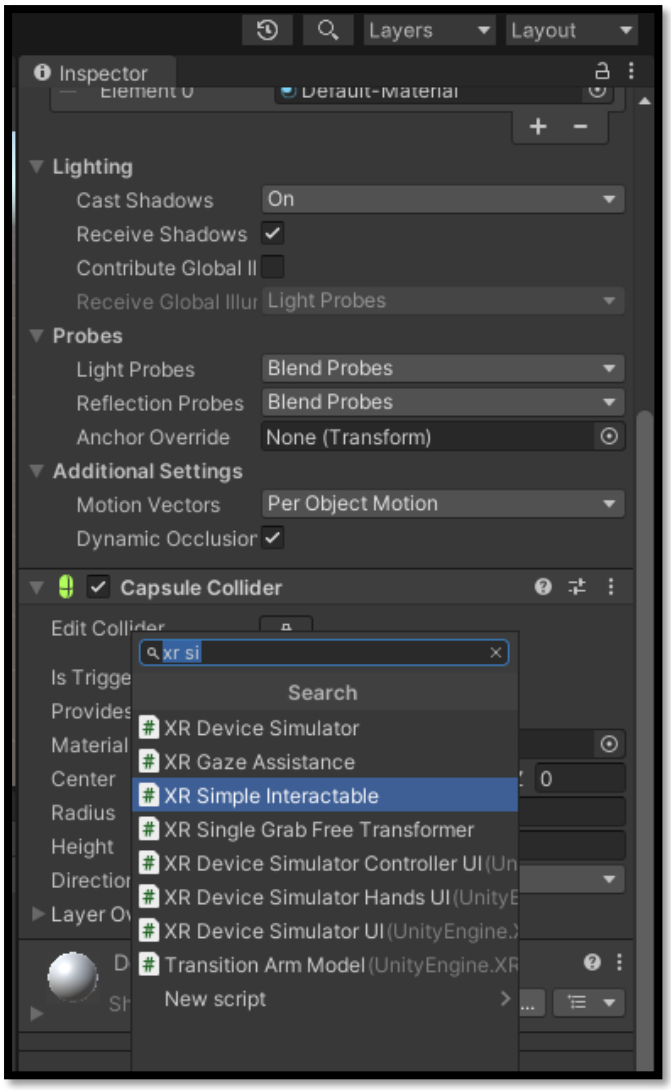
And this is how entire scene will look like⬇



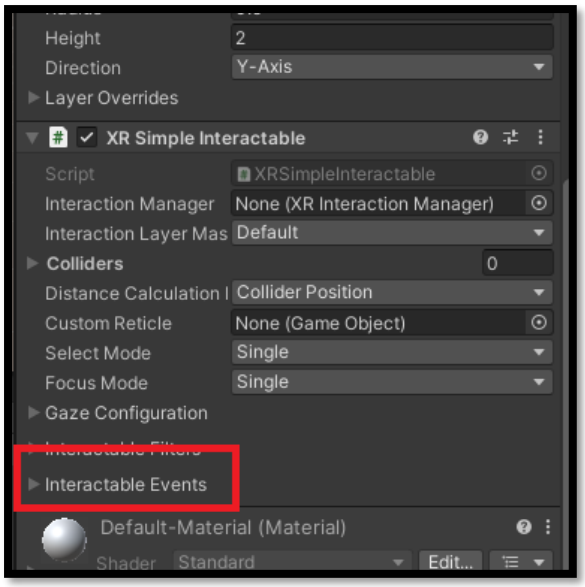
Lets add interactability i.e. as ones VR simulator rays hover on upper small cube it should change its color to Red, for this we will need to create material (how to create material so mentioned earlier)
For this let us check the steps ⬇

Step 30: Go to Capsule gameObject ➞ Click on Add Component ➞ Click on XR Simple Interactable.

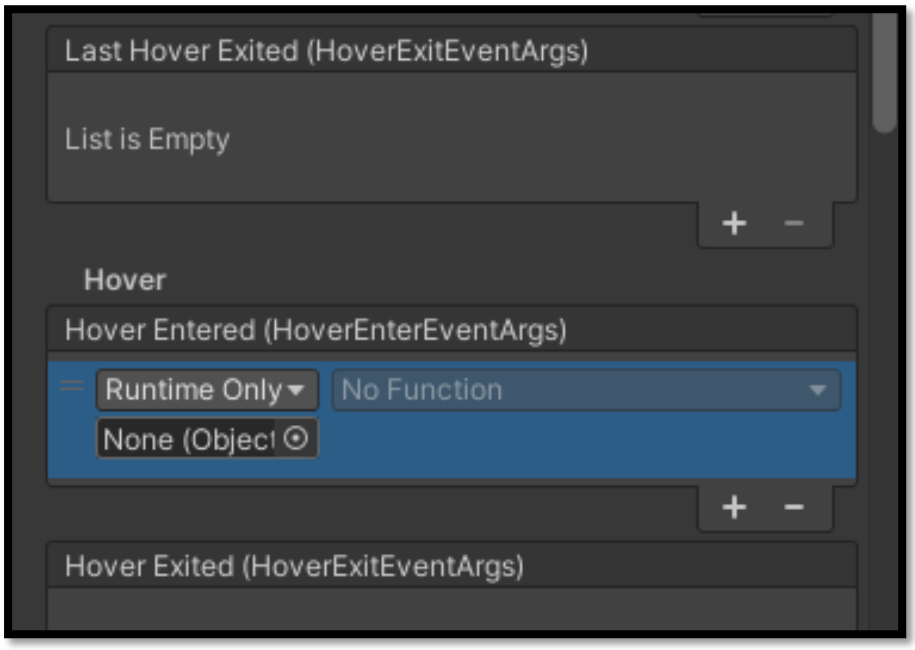




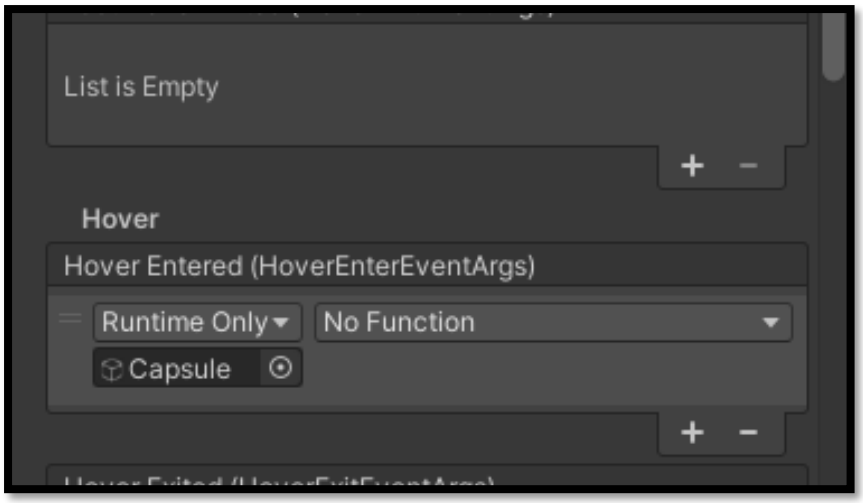
Step 31: Then Click on Interactable events for adding color to cube once hovered by simulator.



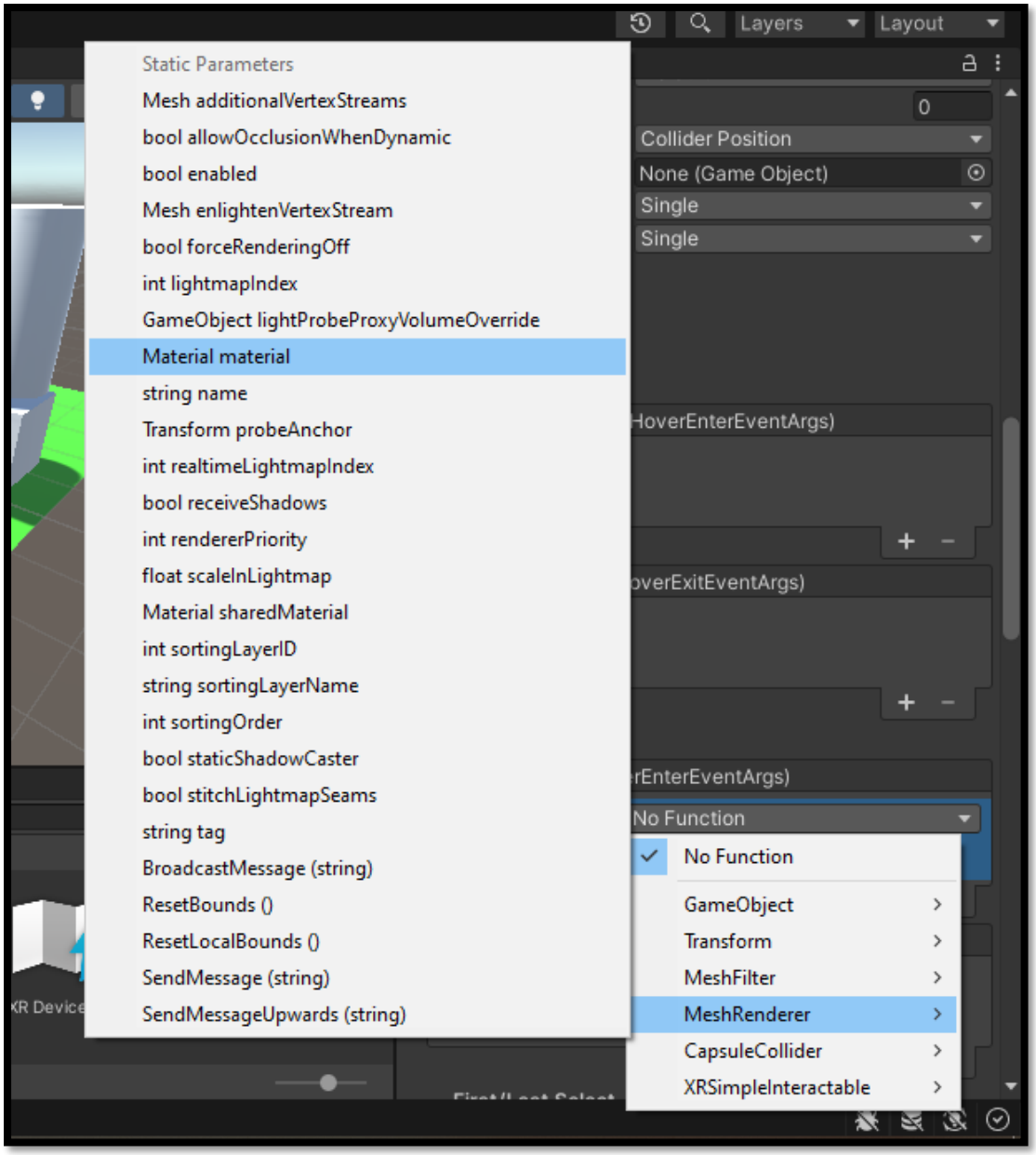
Step 32: Since we need on hover will go for Hover Entered event on the list ➞ then click on ✚ icon to add functionality.

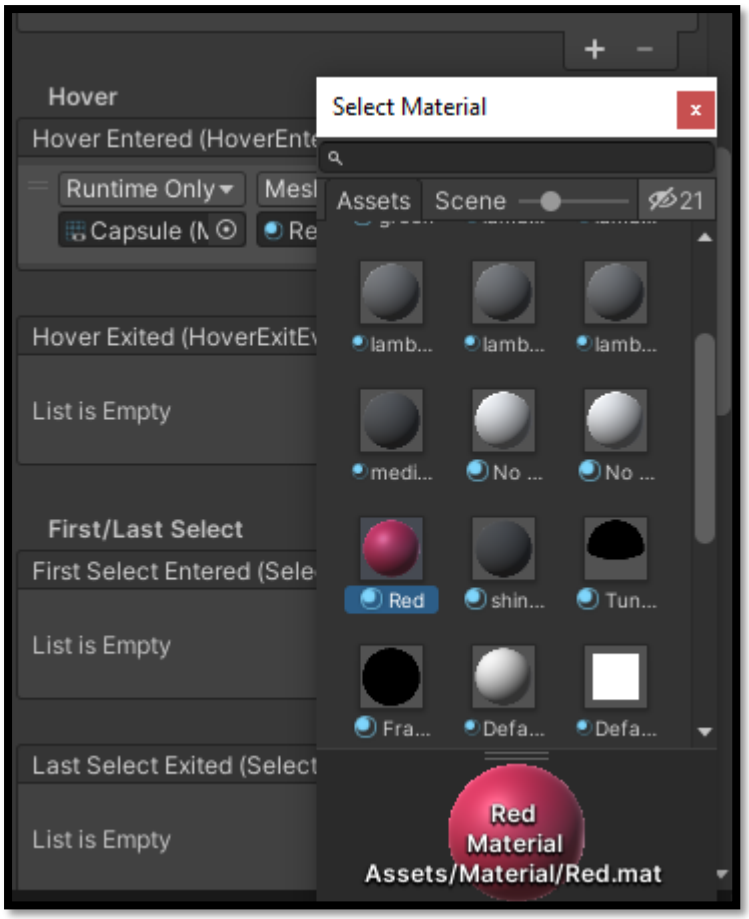


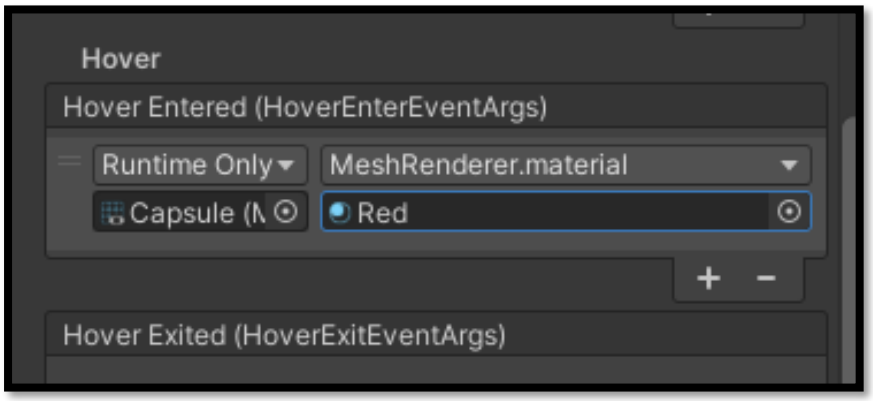
Step 33: Drag and drop your Capsule gameObject on None(Object) [Basically it is asking to add a gameObject on which this functionality should happen]



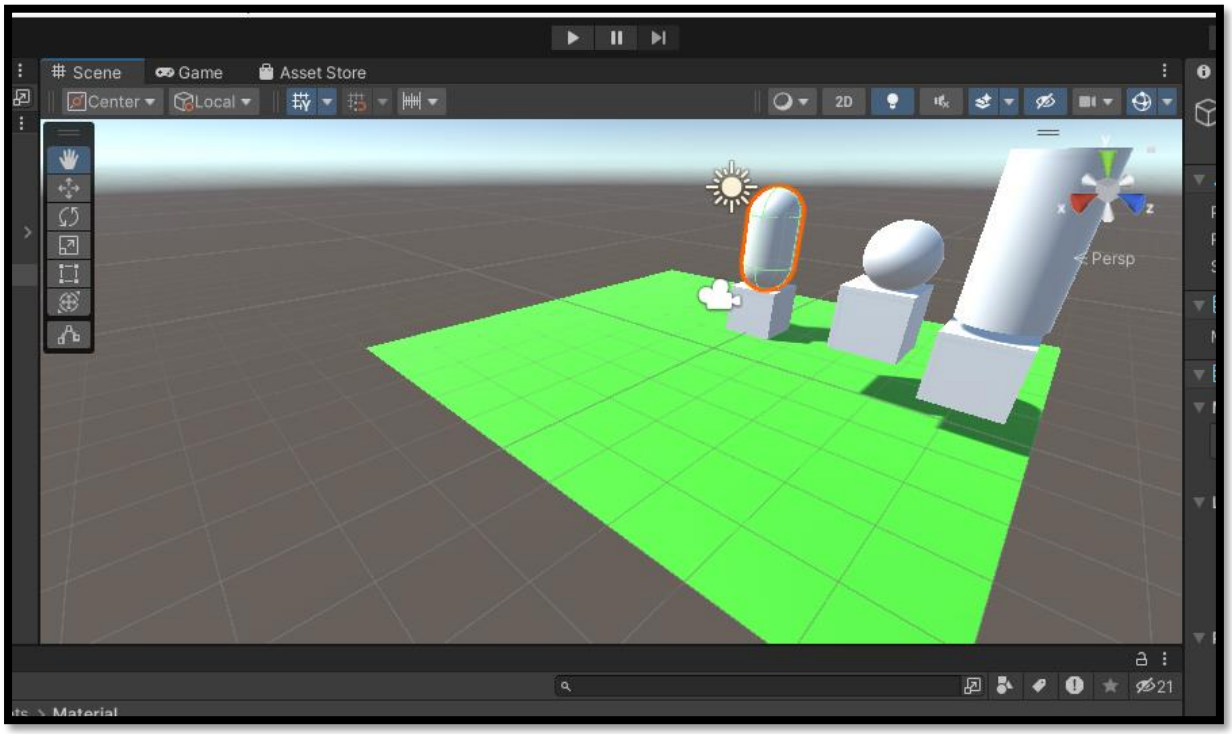
Step 34: Click on No Function for adding Red material ➞ click on Mesh Renderer ➞ then click on Material Material and below dialog box will pop up ➞here select your created material.

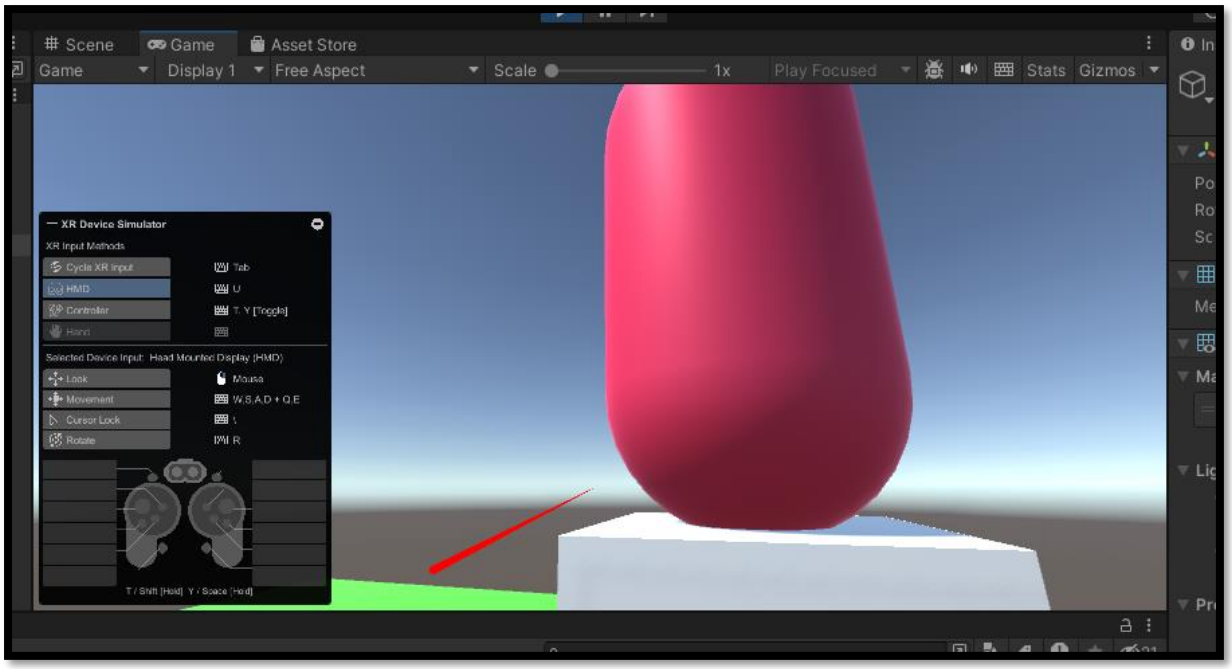



Hover
Hover Entered (HoverEnte
Runtime Only
Mesl
Capsule (M
Re
Hover Exited (HoverExitEv
List is Empty
First/Last Select
First Select Entered (Sele
List is Empty
Last Select Exited (Select
List is Empty
Select Material
Assets
Scene
21
lamb...
lamb...
lamb...
medi...
No ...
No ...
Red
shin...
Tun...
Fra...
Defa...
Defa...
Red
Material
Assets/Material/Red.mat


Hover
Hover Entered (HoverEnterEventArgs)
Runtime Only
MeshRenderer.material
Capsule (M
Red
Hover Exited (HoverExitEventArgs)

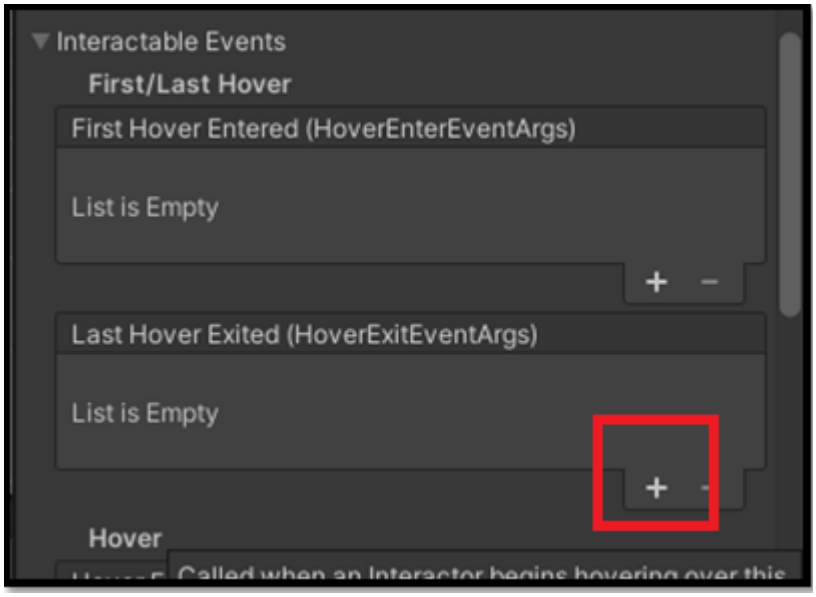
Step 35: Let's test this in game mode

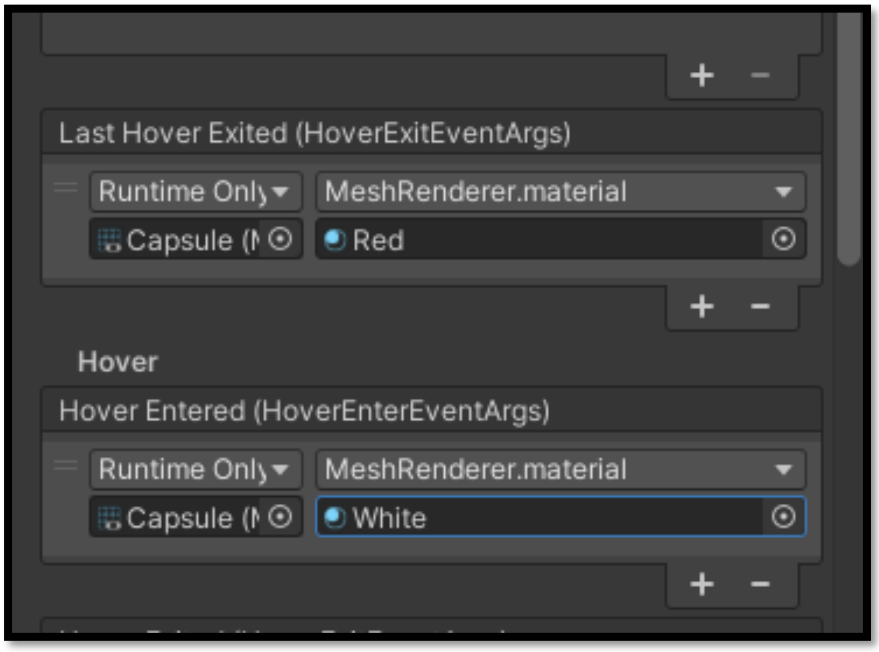




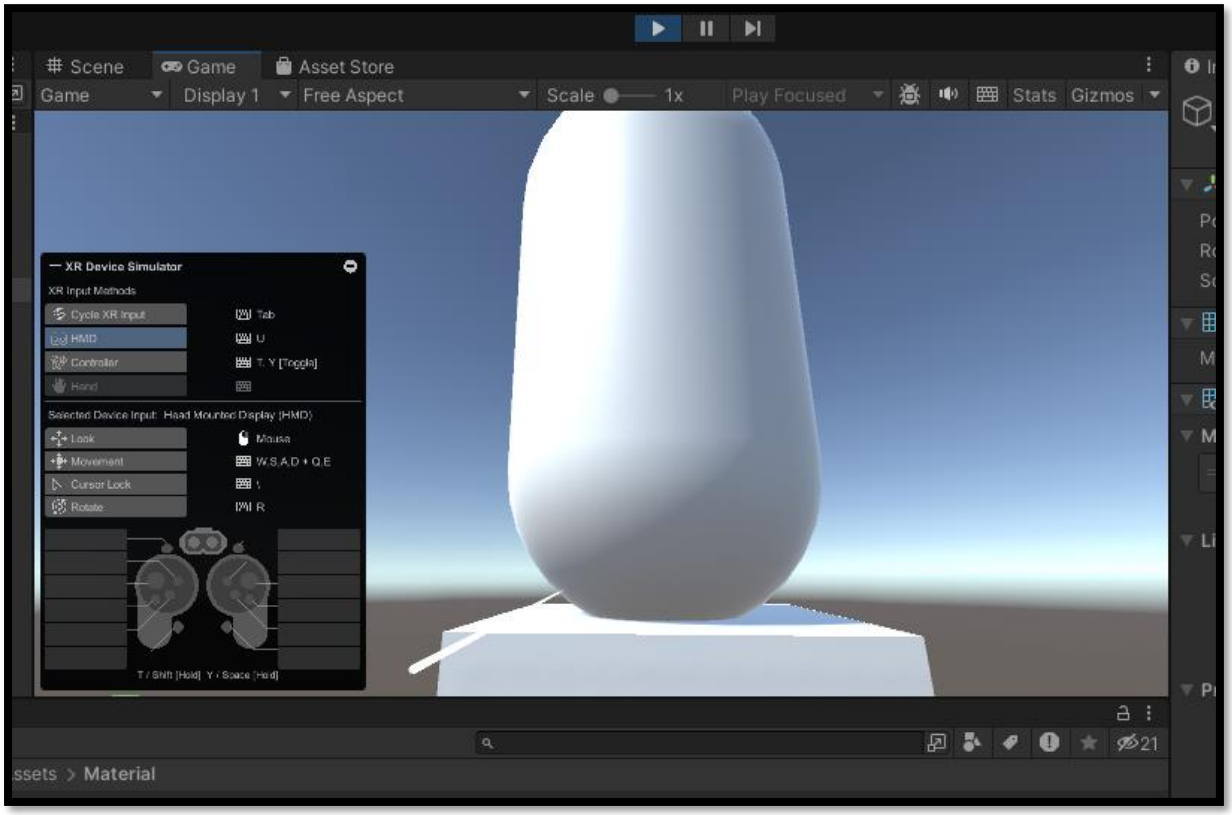
And it works, but as our VR rays get away of the buttoncube gameObject it still remains Red, so to bring it back to its original state lets see the event to be used

Step 36: We will go for Hover Exited event ➞ click on ✚ icon ➞ drag and drop buttoncube gameObject ➞ select MeshRenderer from Functionality ➞ click on Material material ➞ then add material White.





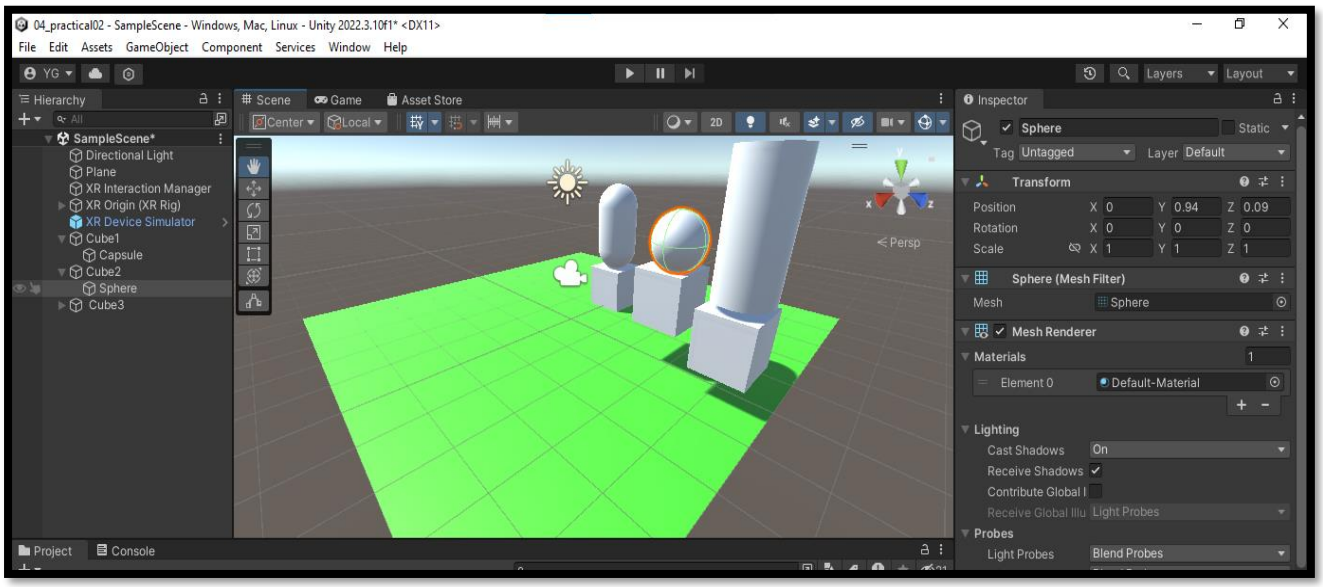
Step 37: Now once again check in game mode.



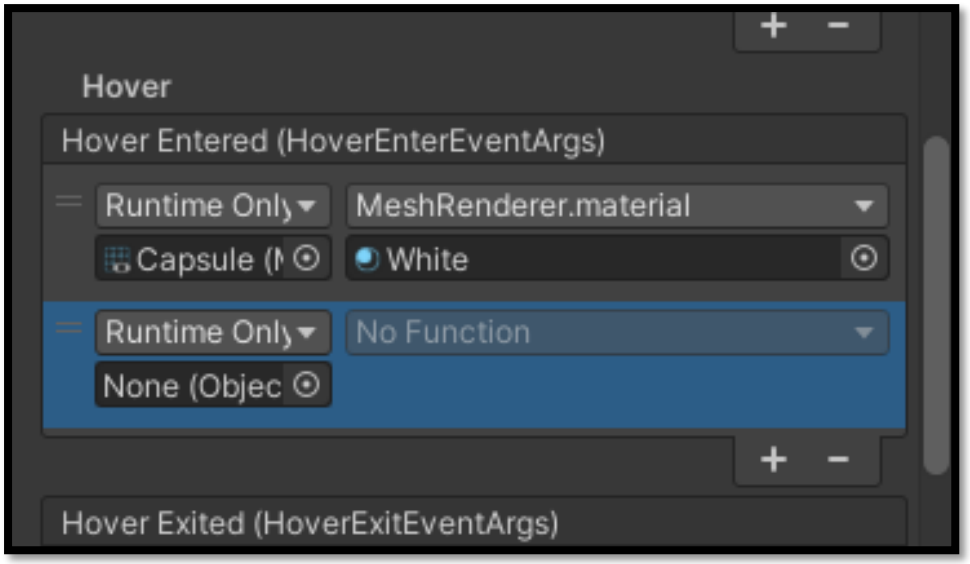
Step 38: Let us add one more interactability i.e. the sphere that you can see on the other Cube should not be visible and should be visible only when our VR rays hover on capsule.

Let us work on this

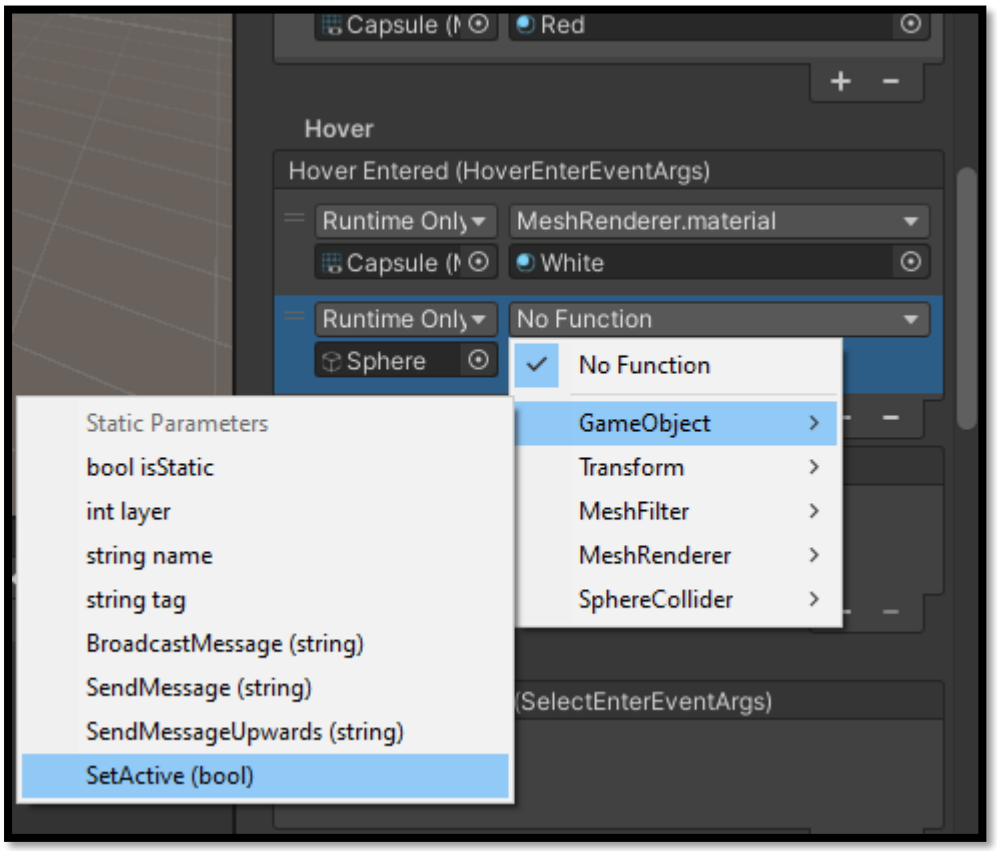
Go to Sphere gameObject ➞

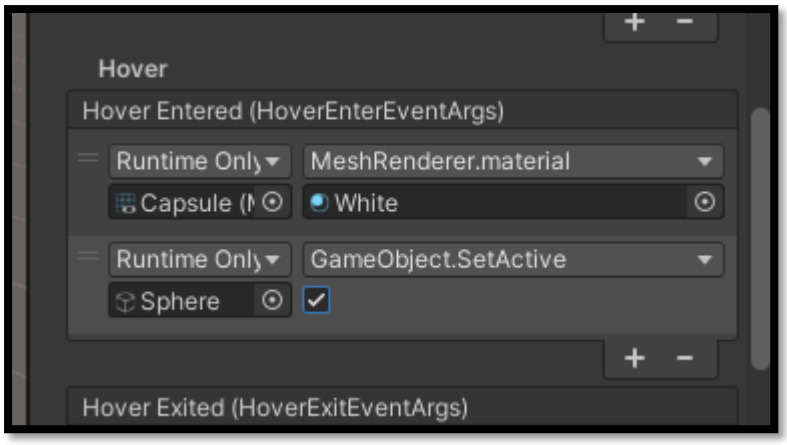


Step 39: Click on capsule➞ go to Inspector Panel ➞ On Hover Entered event we will add one more functionality.



Step 40: On above screen we will add sphere gameObject ➞ will set functionality to GameObject ➞ SetActive (bool) ➞ and then enable checkbox.





Step 41: Let us test it in game mode.