Usage

Overview

Project was created by Eyoel Hailemariam, Josh Bakelaar, Ali Mohamed, Mohammed Al-Darwish, and Bryan Lee under the supervision of Professor Ayan Sadhu and Professor Paul Mensink.

This project is a simulation for Civil Engineers to practice what they do in the real world, but bring it all to a classroom setting.

The current project provides the user with a fully playable version of the simulation for both the Meta Quest 2 (and newer) headset and Windows PC. In the project, there is a tutorial level to understand the controls, a sandbox mode to let the user walk around a bridge and mark down the damage without any time pressure, and finally, a timed challenge mode where the user has a certain amount of time to find and mark all of the damage on the structure.

VR Build

Prerequisites

- Meta Quest 2 (tested) or newer
    - Needs to be running in Developer Mode
    - Storage: ~750MB
- Any sideload app installed on your PC or Mac (ADB [command line tool] or SideQuest [use the Advanced Installer])

Running the Program

Upload the APK to the headset using the sideload app. Then on the headset, find the Structural Integrity app and run it.

Controls

- Left stick: Move
- Right stick or turning head: Rotate
- ≡: Pause
- Left Controller: Aim Teleport
- Left Hand Trigger: Prepare Teleport
- Left Index Trigger: Teleport
- Y: View timer
- Right Controller: Aim Pins

- Right Index Trigger: Pin/Interact with Pins and Menus
- Rotate left arm (palm up): View menus

PC Build

Prerequisites

- Windows 10 (tested) or Mac (untested but should work)
- Minimum Specs
    - CPU: Intel i5-8400H or Ryzen 5 2500U
    - GPU: Intel UHD 630 or Radeon Vega Mobile Gfx
    - RAM: 4GB
    - Storage: ~750MB
- Recommended Specs
    - CPU: Newer i5 or Ryzen 5 than minimum specs
    - GPU: Integrated graphics or Nvidia MX250 or newer
    - RAM: 8GB
    - Storage: ~750MB

Running the Program

Run the Structural Integrity executable.

Controls

- W: Move forward
- S: Move backward
- A: Move left
- D: Move right
- Spacebar: Jump
- Sprint: Shift
- Click:

Adding New Structures

Prerequisites

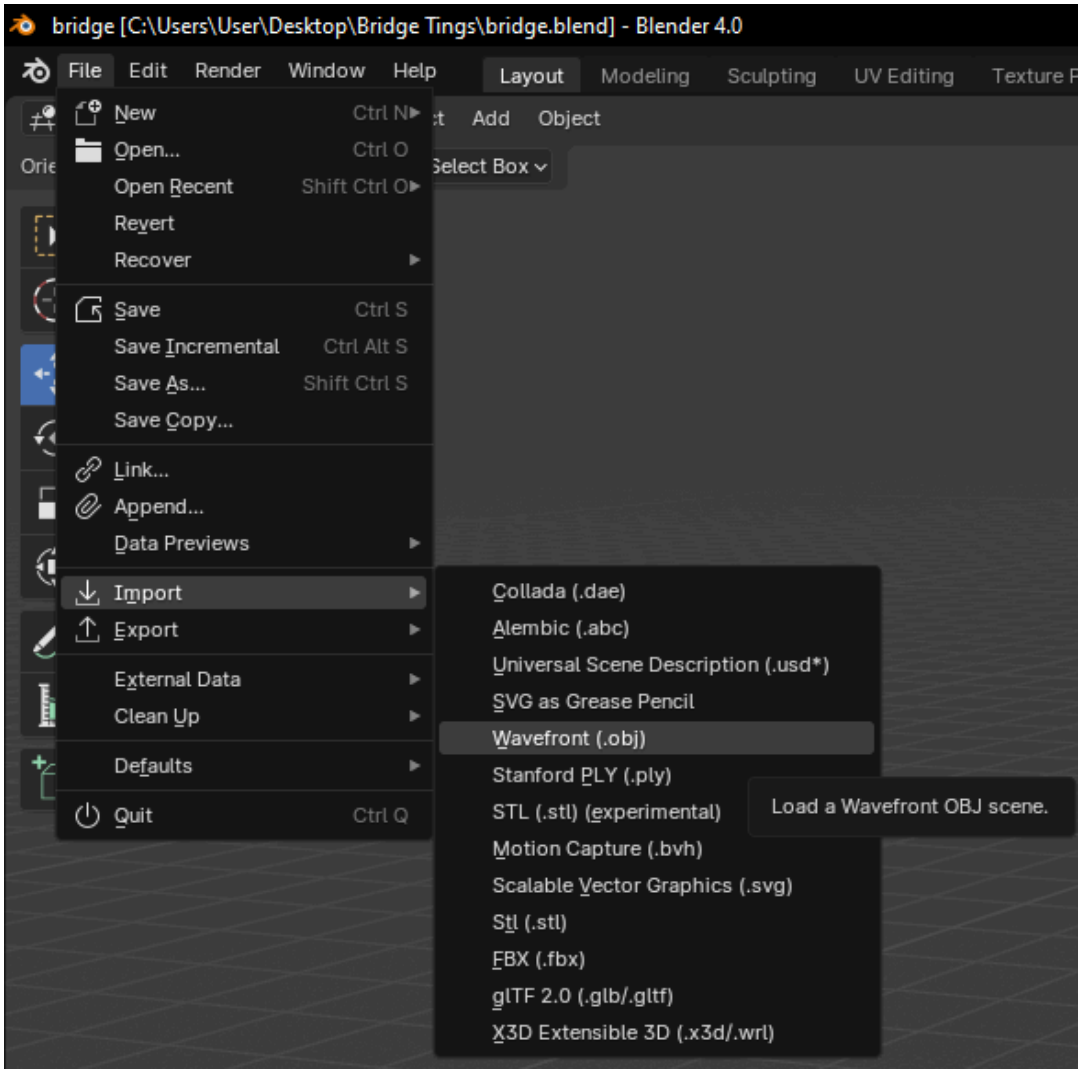
- Unity Version 2022.3.9 with Android Build Support
    - Install Unity Hub
    - Install Unity Version 2022.3.9 (make sure to select the Android Build Support Module)
        - If the Android Build Support Module wasn't installed, you can add it post-installation

- Meta XR All-in-One SDK
- Oculus Quest device in Developer Mode

Preparing the Model

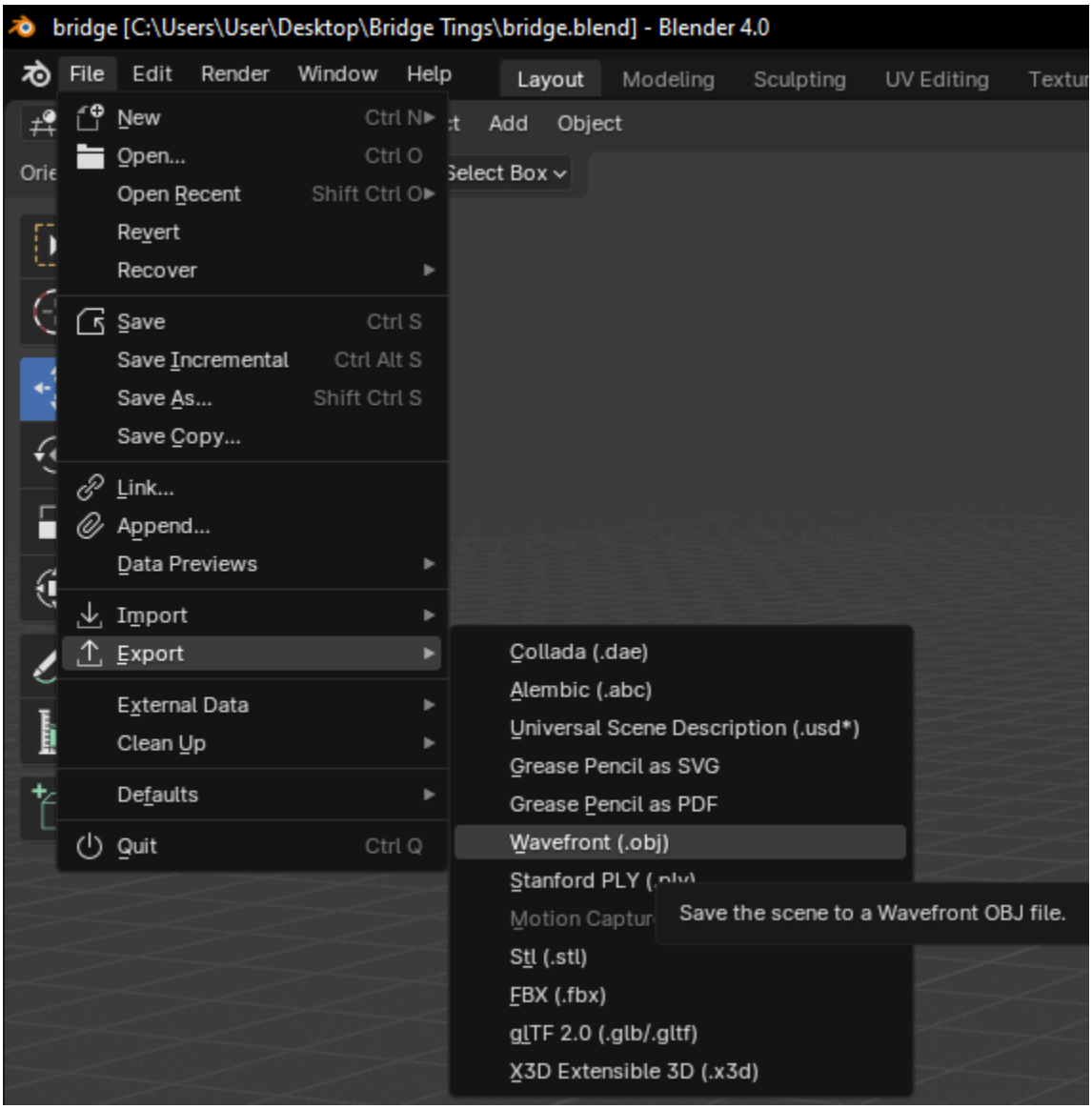
Going off the initial model we were given, the bridge scans need to be combined into one object. This can be done using Blender. See here for how to download Blender.

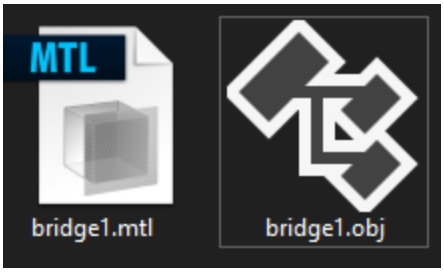
We need to import the 4 scans into a new project in Blender. Once Blender is open, create a new project and follow these steps to import the .obj files: File->Import->Wavefront (.obj)



Do this for all parts of the bridge scan and you should see the model line up in the scene. Next, you would need to use the **bisect** tool to cut up the bridge's unnecessary surroundings like trees and such. We want to keep only the important parts of the structure for performance's sake. Here is a tutorial on how to use the bisect tool to cut up the model.

When finished, you can head over to File → Export → Wavefront (.obj) to export the cut-up model into one single .obj file and its metadata (material).





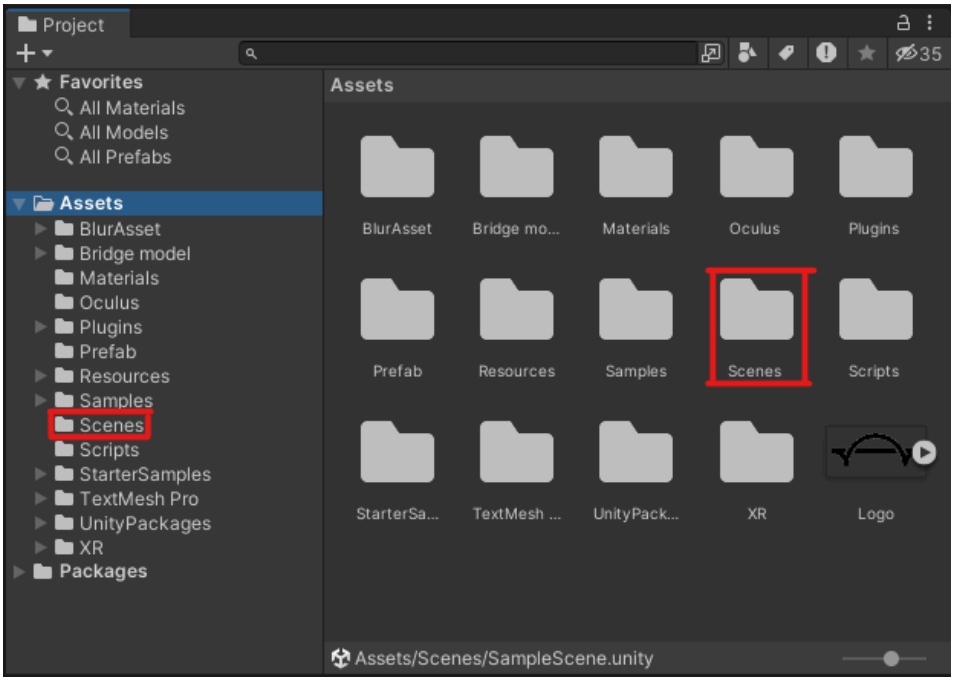
Now take these exported files (2 in this case) and drag them to the Assets folder of the unity project and into the relevant folder within,(i.e. Structures)

Now you are ready to set up the scene.

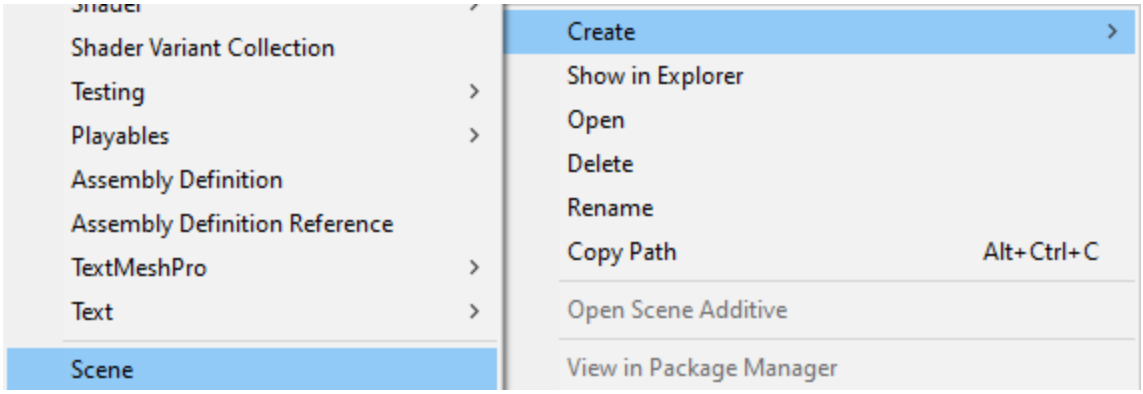
Creating the Scene

Adding a New Scene

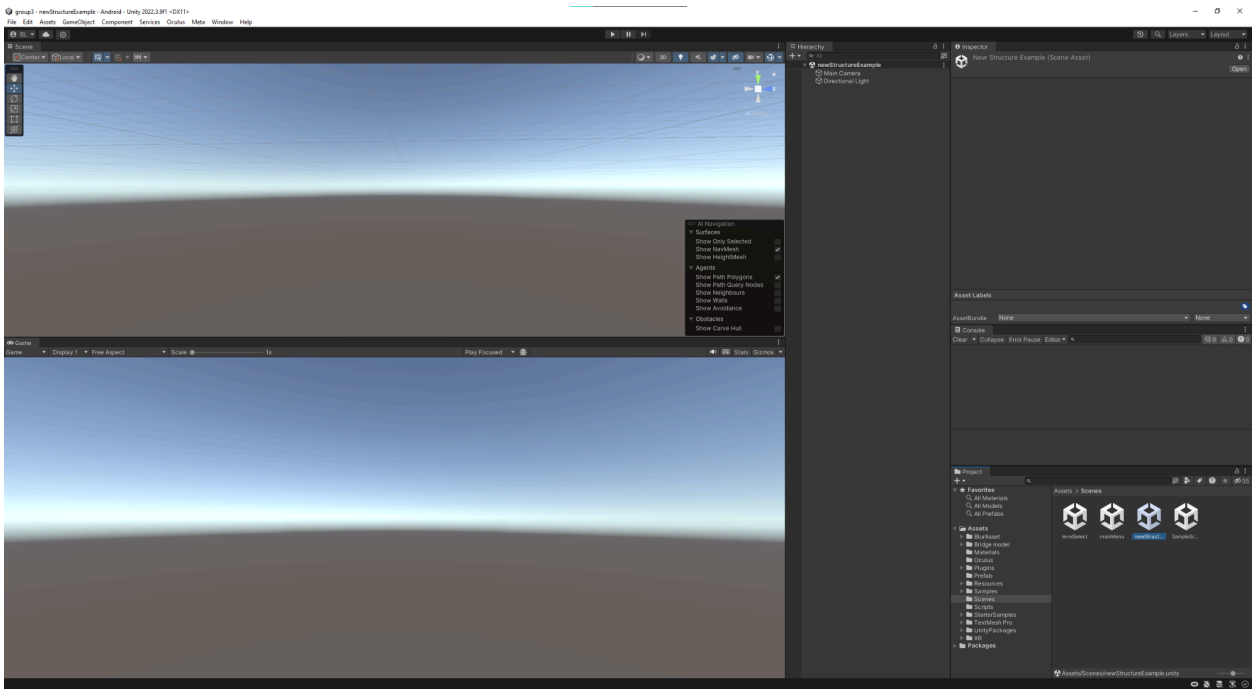
In the Unity editor, find the Project window and go to the Scenes folder (under Assets). If you can't find the Project window, go to the top menu and click on Window → General → Project



In the Scenes folder, right-click and click on Create → Scene

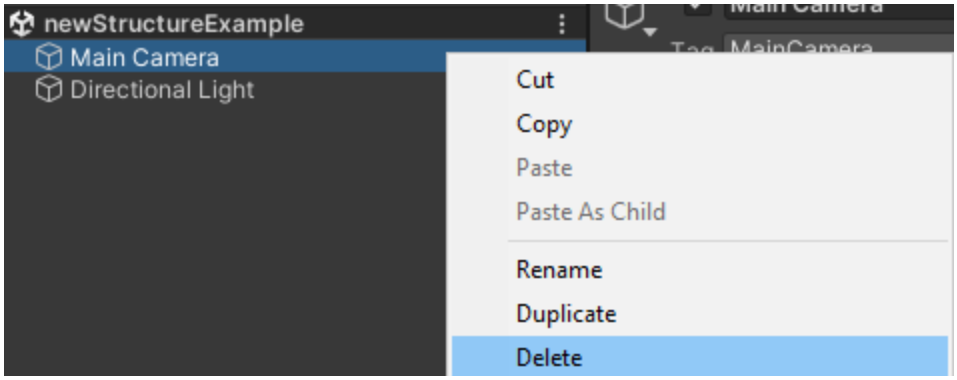


Give the Scene a name and then double-click on the new Scene you created to load it. This will bring you to a blank scene.



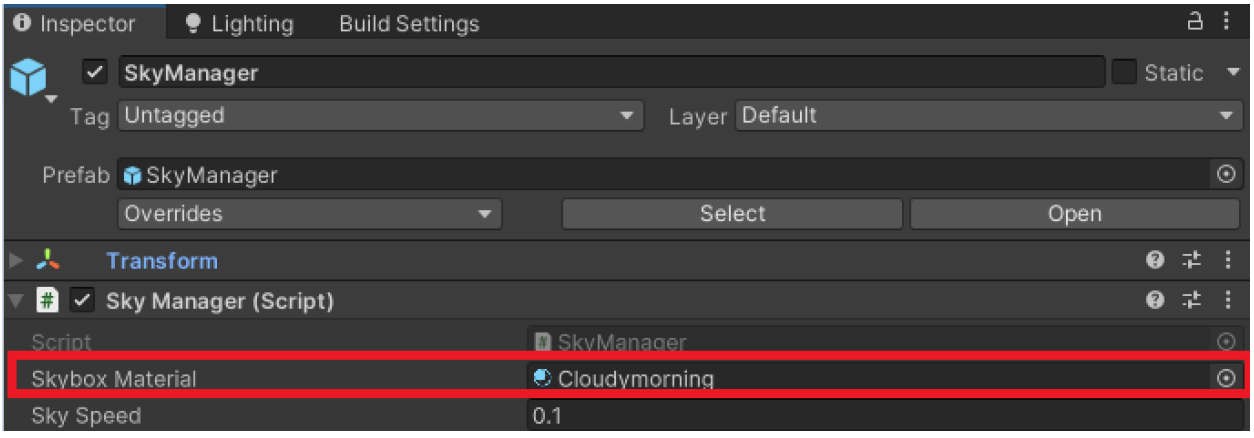
Preparing the New Scene

In the Hierarchy window, remove the Main Camera by right-clicking on it and selecting Delete or left-clicking on it and pressing the Delete key on your keyboard. If you can't find the Hierarchy window, go to the top menu and click on Window → General → Hierarchy

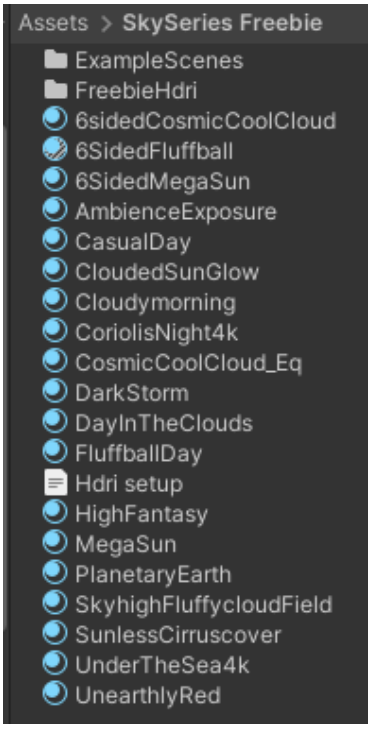


Before dragging in the bridge model, first, let's set up the playable area. Head to the Prefabs folder and drag in the **Playable Area** prefab. This will give you a bounded area that is ready for the player to move in. This area can be modified as needed (smaller or bigger).

You can also drag in the **SkyManager** prefab to apply a default sunny day skybox to the scene. The Skybox can be changed by going to the SkyManager object in the hierarchy and changing the Skybox material within.



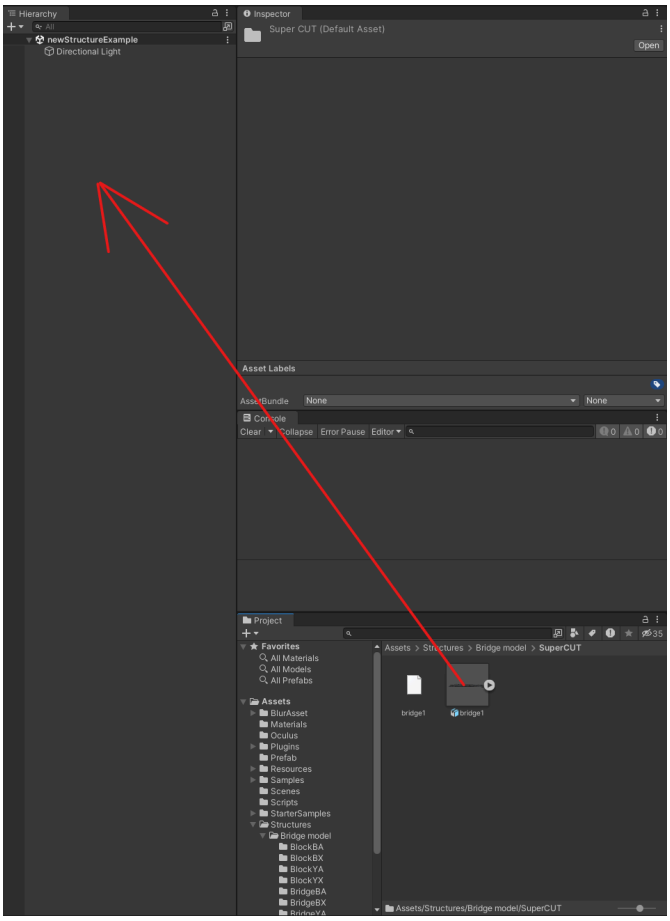
The different sky box materials can be found in the “Assets/SkySeries Freebie” folder:



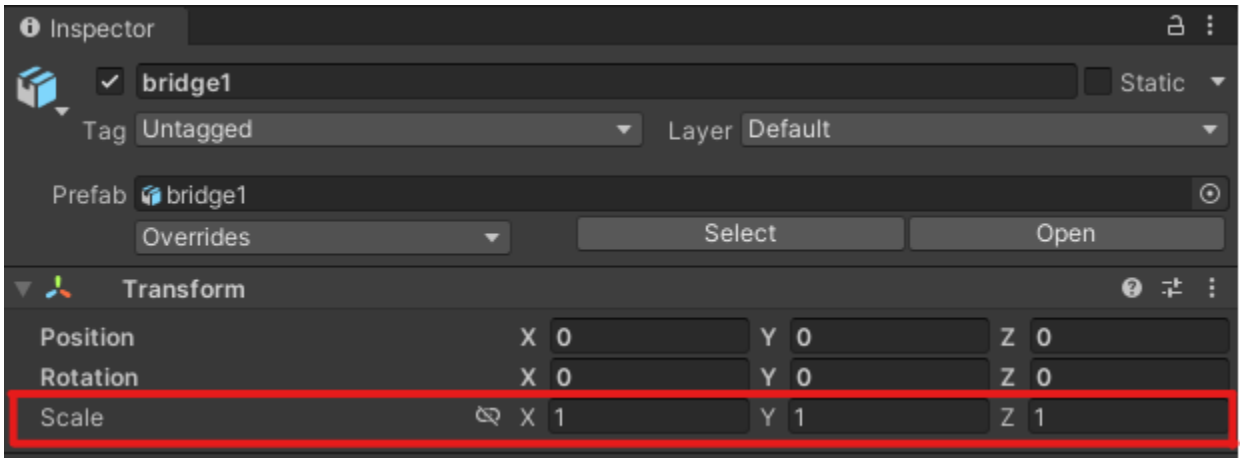
You can drag and drop the one you want into the skybox material slot or simply click on the slot and type the name into the search field.

Importing the Structure

In the Project window, go to the Structures folder and drag the 3D model of the new Structure into the Hierarchy window.



If the size of the model needs to be adjusted, you can change the scale in the Inspector window. If you can't find the Inspector window, go to the top menu and click on Window → General → Inspector.

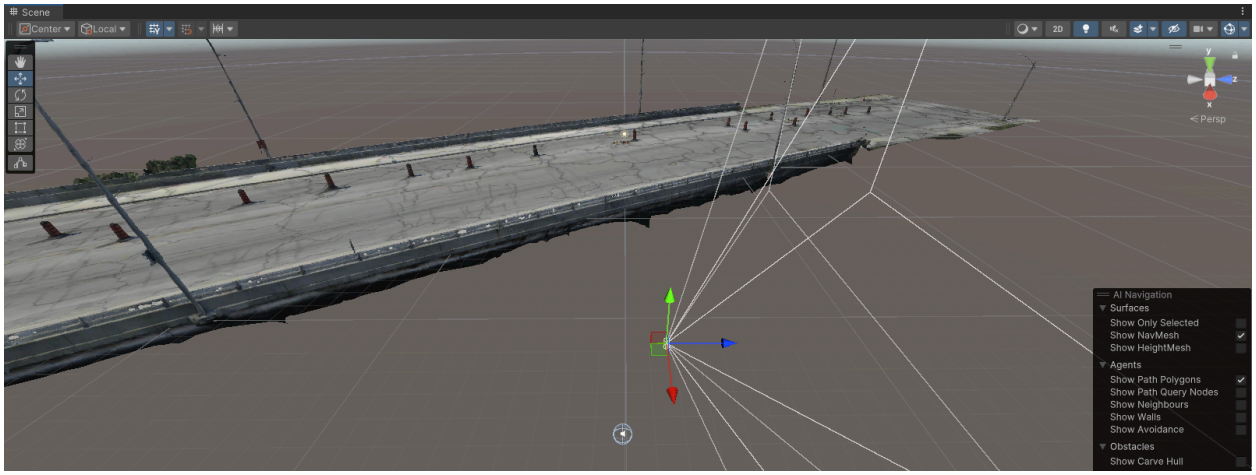


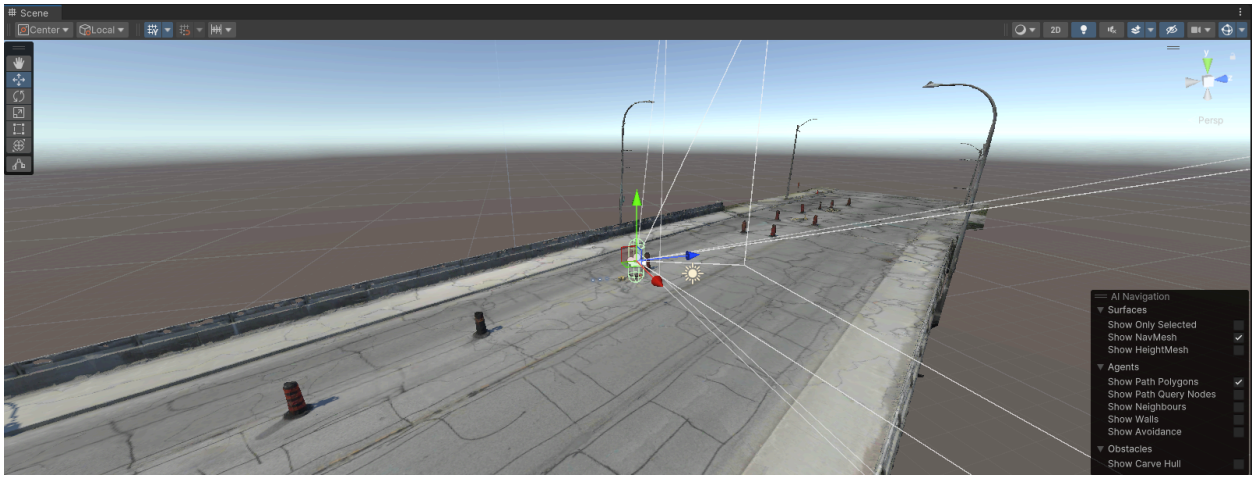
In the Project window, go to the Prefab folder and add the following objects:

- PlayerController
- CanvasModule

- Select Damage Canvas
- Remove Damage Canvas
- See Damage Canvas
- PopulateCurvedCanvas
- DebugPanel
- Pause Menu Variant
- GameController
- Timer (down) or Timer - up

If the position of the PlayerController needs to be adjusted, this can be done in the Inspector window by changing the Position (under the transform component) or by dragging it in the scene.



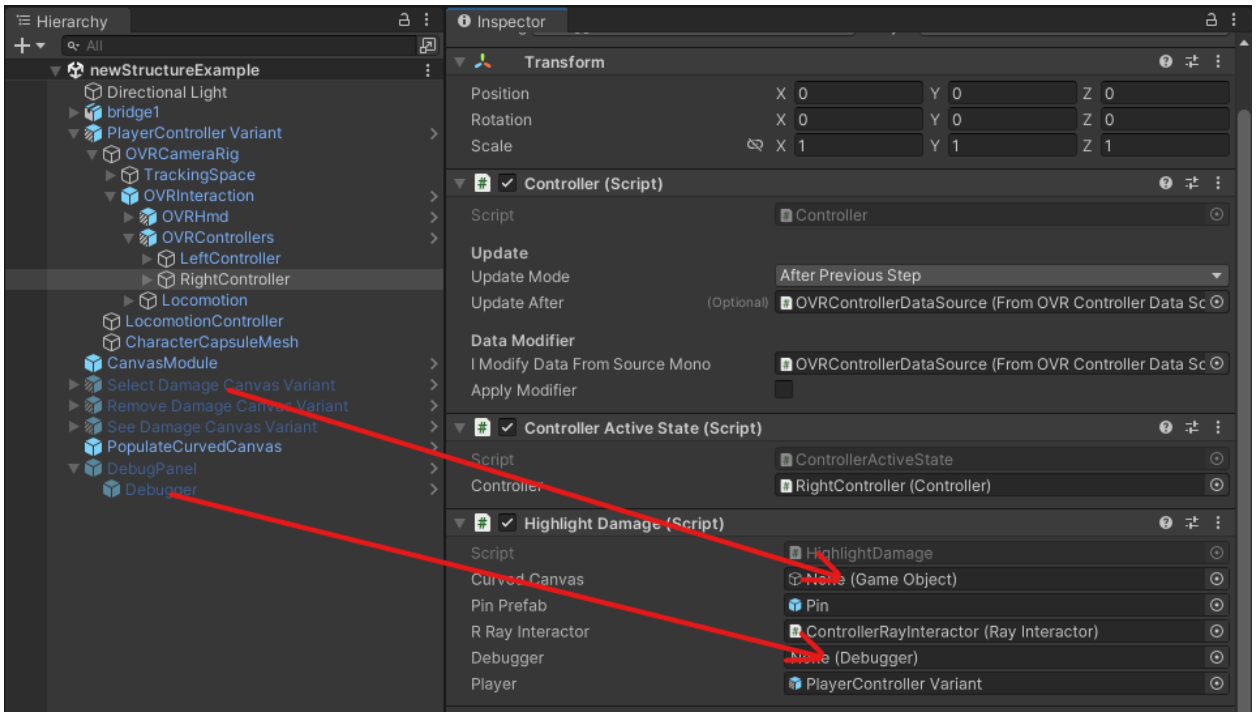


Binding the GameObjects

PlayerController

In the Hierarchy window, click on the dropdowns to traverse through PlayerController → OVRCameraRig → OVRInteraction → OVRControllers → RightController

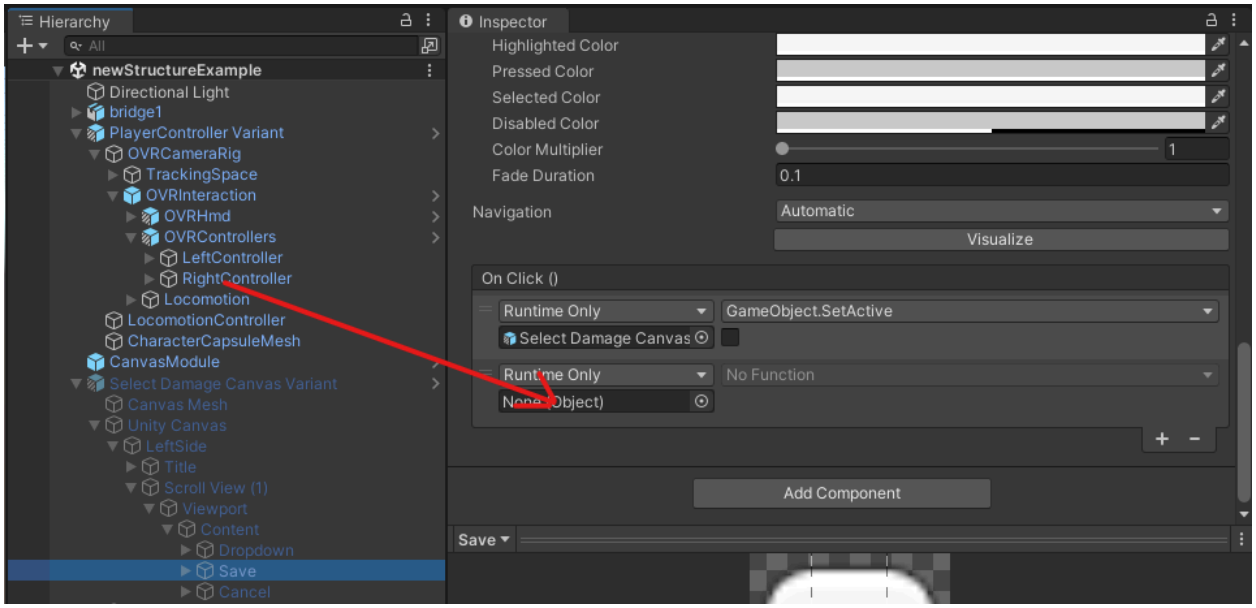
Click on RightController and find the Highlight Damage Script in the Inspector window. Drag and drop the Select Damage Canvas and Debugger (click the dropdown on Debug Panel) to the Curved Canvas and Debugger slots on the script.



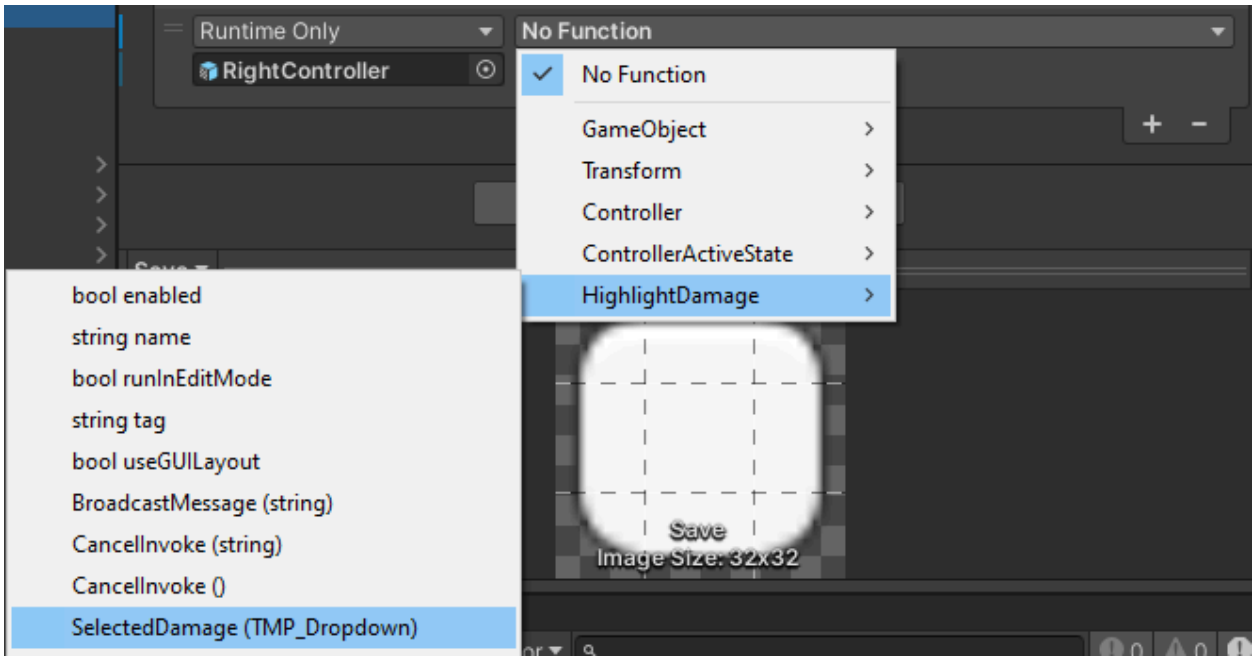
Select Damage Canvas

In the Hierarchy window, click on the dropdowns to traverse through Select Damage Canvas → Unity Canvas → LeftSide → Scroll View (1) → Viewport → Content → Save

Click on Save and find the Button component in the Inspector window. Under the On Click () section, drag and drop the RightController from the PlayerController (PlayerController → OVRCameraRig → OVRInteraction → OVRControllers → RightController).

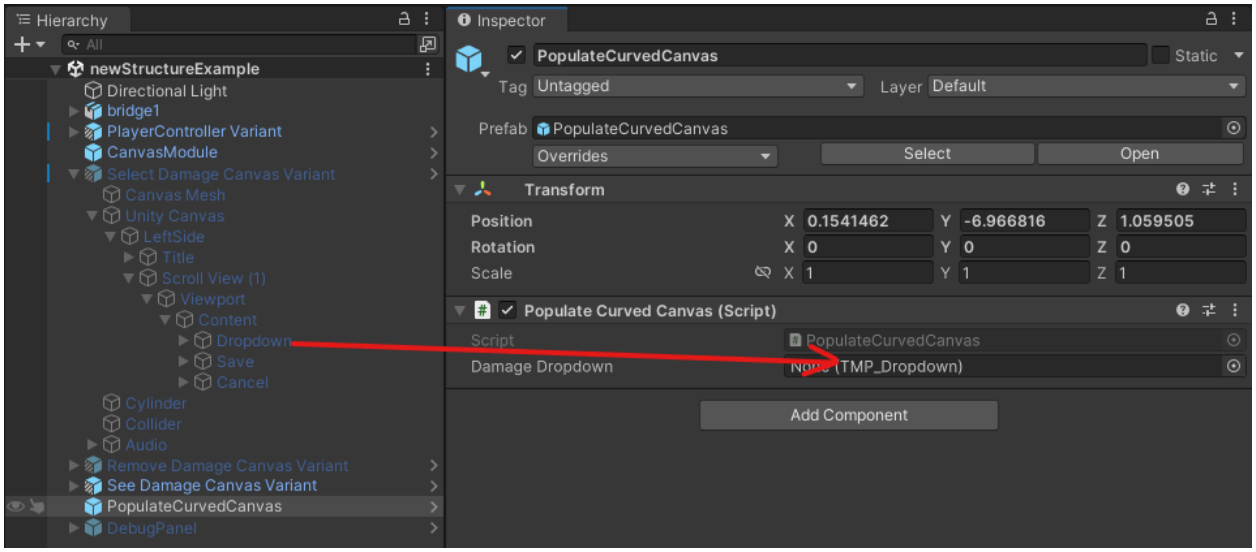


Add a function to the button by clicking on the dropdown that says “No Function” and clicking HighlightDamage → SelectedDamage (TMP_Dropdown)



PopulateCurvedCanvas

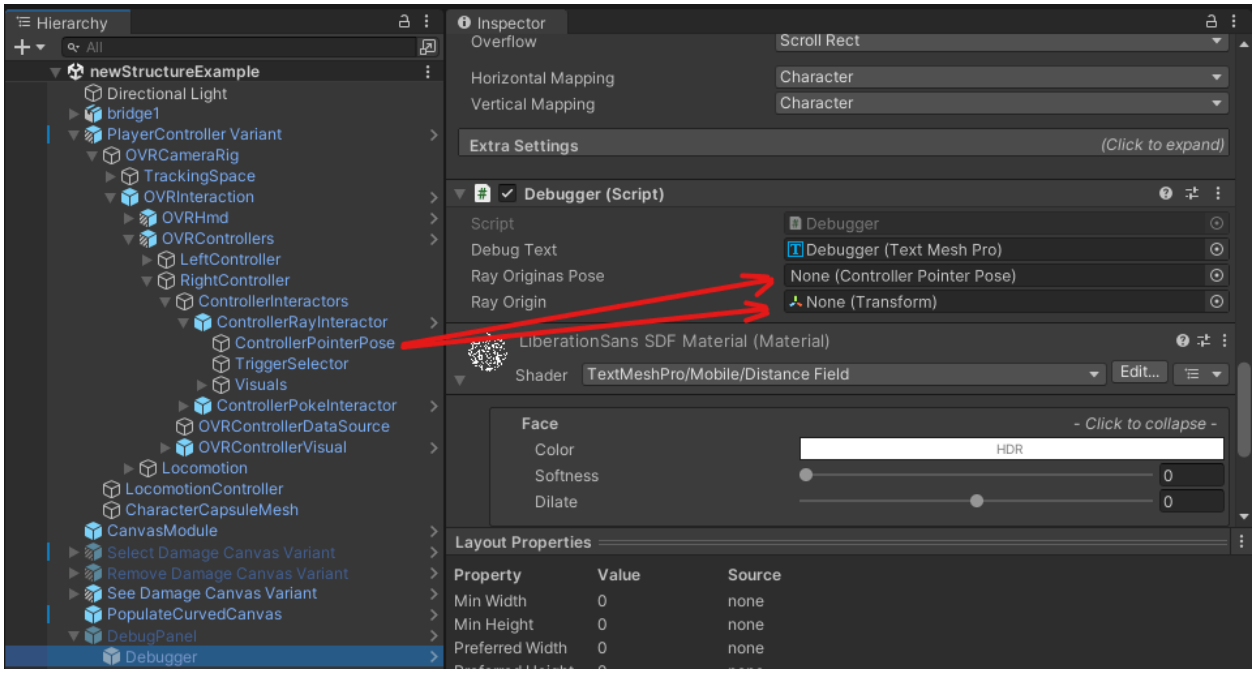
In the Hierarchy window, click on the PopulateCurvedCanvas object and find the Populate Curved Canvas script in the Inspector window. Drag and drop the Dropdown component from the Select Damage Canvas (Select Damage Canvas → Unity Canvas → LeftSide → Scroll View (1) → Viewport → Content)



DebugPanel

In the Hierarchy window, click on the dropdown to traverse through DebugPanel → Debugger

Click on the Debugger and find the Debugger script in the Inspector window. Drag and drop the ControllerPointerPose (PlayerController → OVRInteraction → OVRControllers → RightController → ControllerInteractors → ControllerRayInteractor) to the Ray Originas Pose and Ray Origin spots on the script.



GameController (for Time Mode Scenes)

In the Hierarchy window, click on the ‘GameController’.

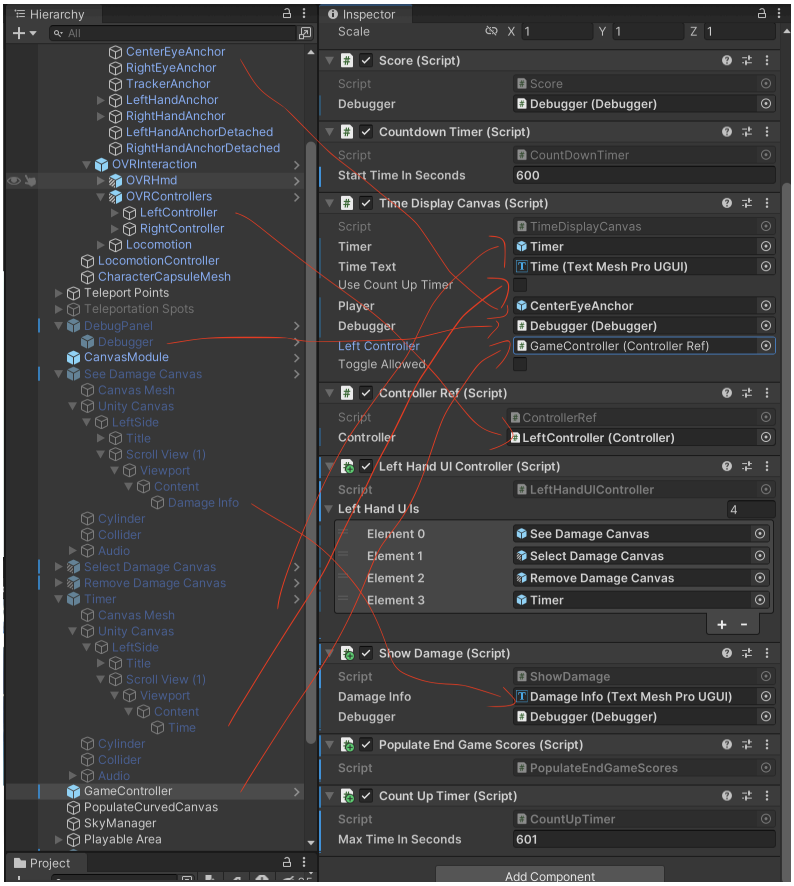
In the Inspector window, find the Score script and drag and drop the Debugger (DebugPanel → Debugger) into its Debugger field. Next, find the TimeDisplayCanvas script and drag and drop the following:

- Timer (In the Hierarchy. If not from Prefabs folder)
- Time (Timer → Unity Canvas → LeftSide → Scroll View (1) → Viewport → Content → Time)
- CenterEyeAnchor (PlayerController → OVRCameraRig → TrackingSpace → CenterEyeAnchor)
- Debugger (DebugPanel → Debugger)
- GameController (In the Hierarchy. If not from the Prefabs folder)

Then, find the Controller Ref script and drag and drop LeftController (PlayerController → OVRInteraction → OVRControllers → LeftController) into its ‘Controller’ field.

After that, find the LeftHandUIController script and using the ‘+’ button add 4 elements in the ‘Left Hand UIs’ field. Then, in each element drag and drop See Damage Canvas, Select Damage Canvas, Remove Damage Canvas, and Timer game objects. Order does not matter. All of those objects can be found in the hierarchy.

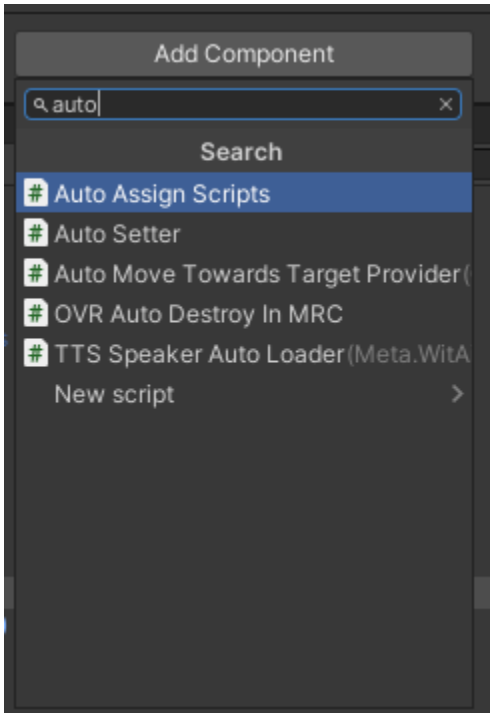
Lastly, find the Show Damage script and drag and drop Damage Info (See Damage Canvas → Unity Canvas → LeftSide → Scroll View (1) → Viewport → Content → Damage Info) and Debugger (DebugPanel → Debugger) into Damage Info and Debugger fields respectively.



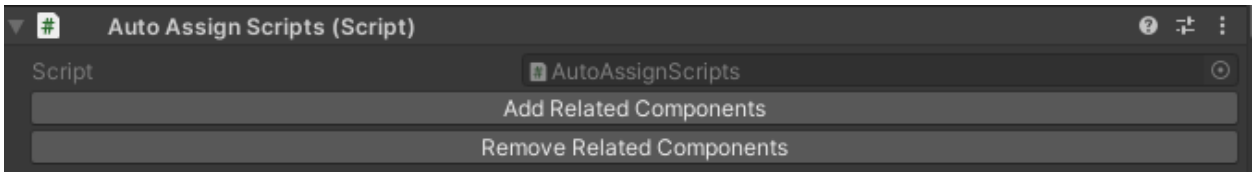
Making The Structure Interactable

To make the imported structure interactable, click on the object model from the hierarchy

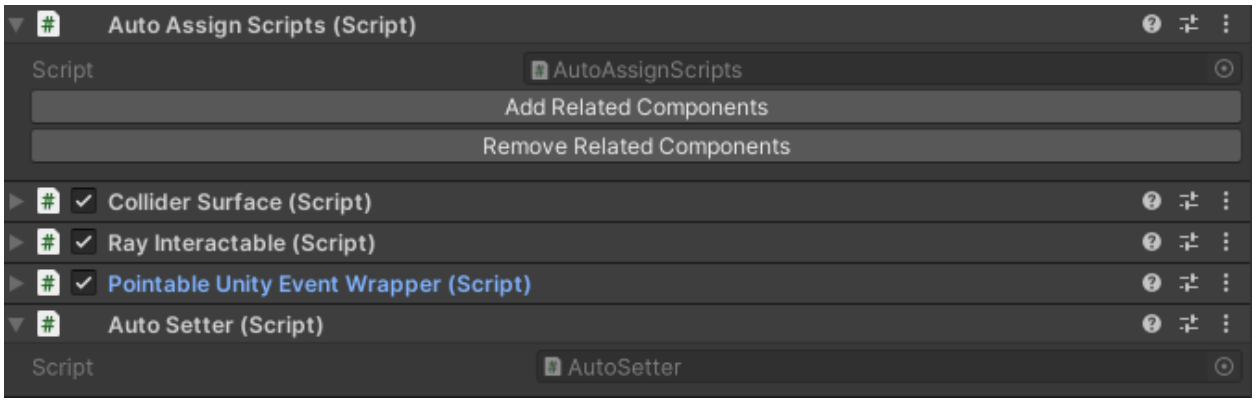
In the Inspector window scroll to the bottom till you find the add component button. In there, you want to search for “Auto Assign Scripts” and add that to the object.



Now you should see this component in the Inspector.



By clicking on “Add Related Components”, the interaction scripts will be added and now you should see these in the Inspector:

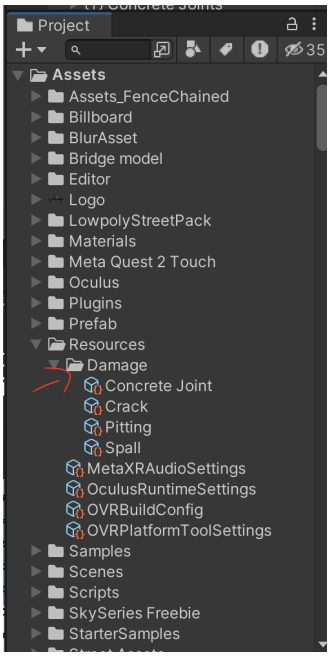


NOTE: This same method can be done to any object in the scene to make it damage interactable (i.e. Cones, Concrete Slabs, Fences, Walls, Roadblocks, etc).

Adding Damages

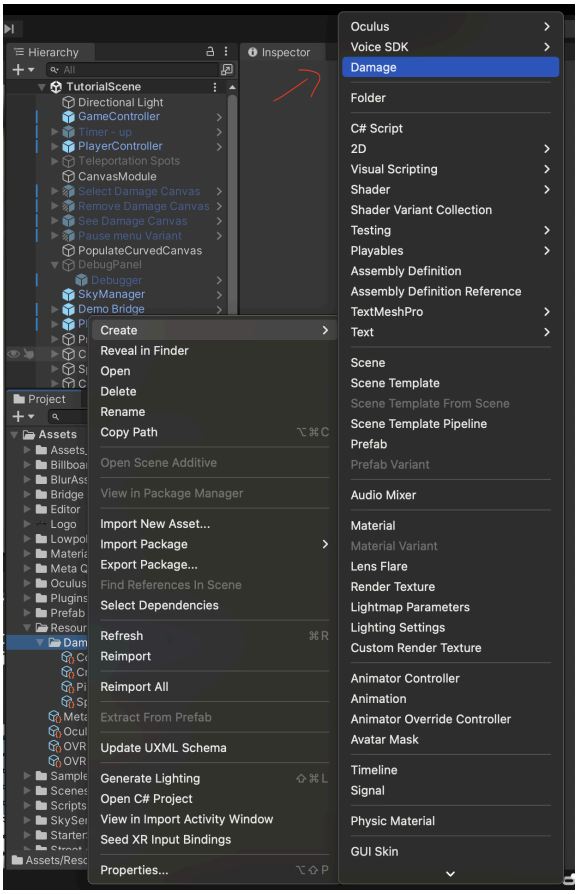
Locating the Damage Scriptable Objects

In the Project folder navigate to ‘Assets’ → ‘Resources’ → ‘Damage’. Here you can find premade Damages.

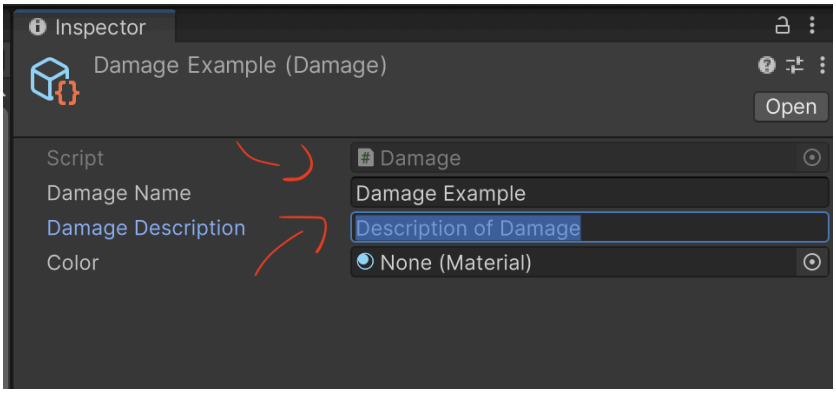


Making the Damage Scriptable Objects

If you would like to add more Damages, then right-click on the ‘Damage’ folder or inside it. Then go to ‘Create’ → ‘Damage’.



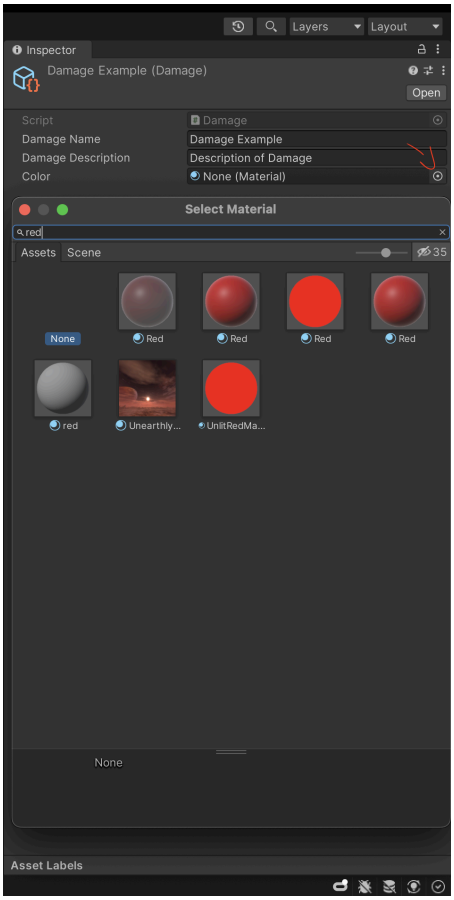
Now fill the ‘Damage Name’ and the ‘Damage Description’ fields with the appropriate information.



Search for Materials

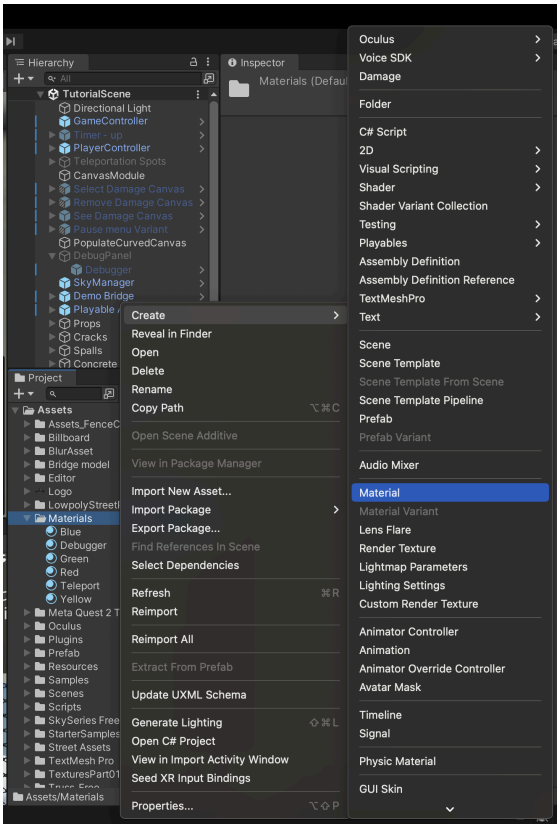
The ‘Color’ field represents the color of the pin when you choose that damage.

To add a premade color, you can press the circle button and then search for a material you like.

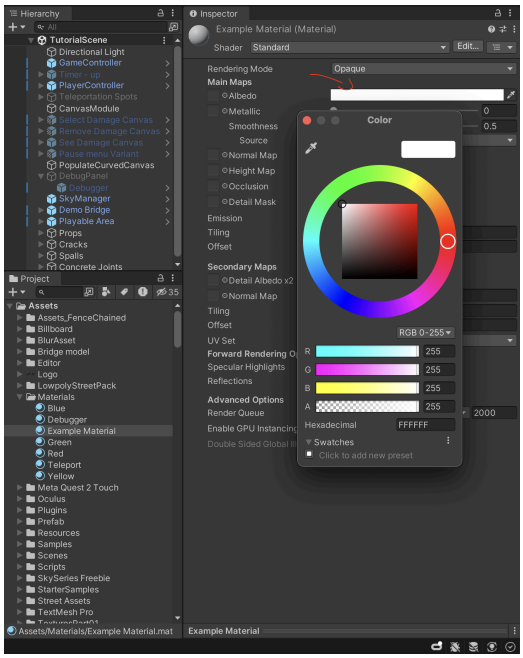


Create New Materials

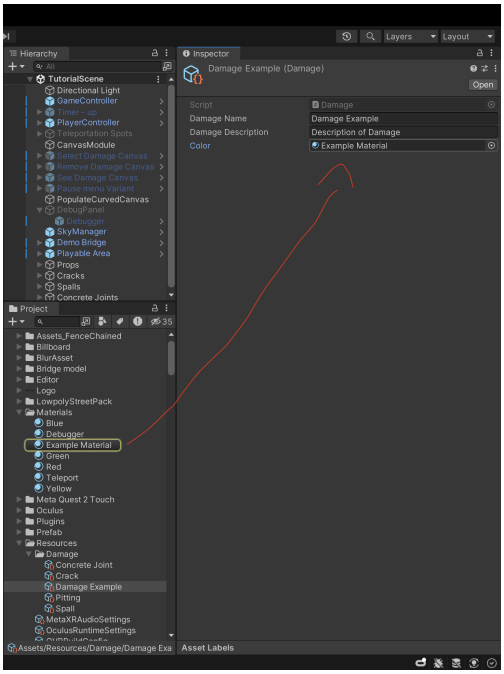
To create a new color, you can right-click on the ‘Materials’ folder and then go to ‘Create’ → ‘Material’. Give it a name (preferably the name of the color you would like to make).



Then choose a color by clicking on the white color in the ‘Albedo’ field of your newly created material.



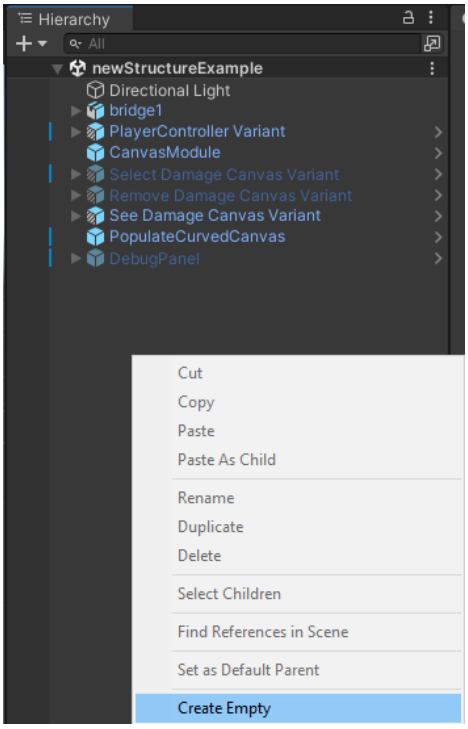
After you are satisfied with the color, drag and drop your newly created material into the ‘Color’ field of your Damage.



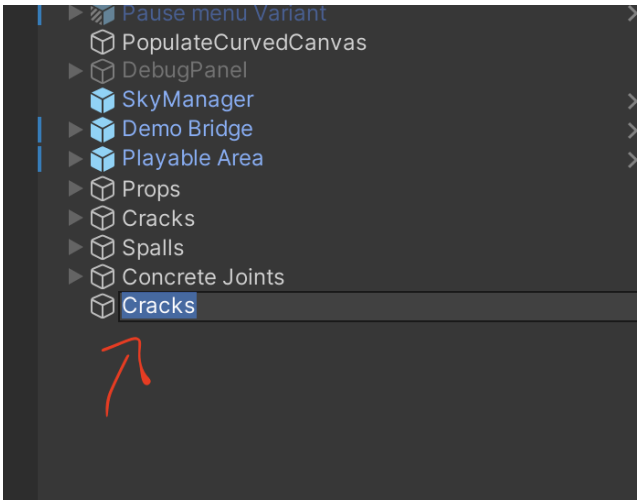
Adding Damage Detectors

Making the Parent GameObject

Make an empty GameObject in the Hierarchy window by right-clicking and selecting Create Empty.

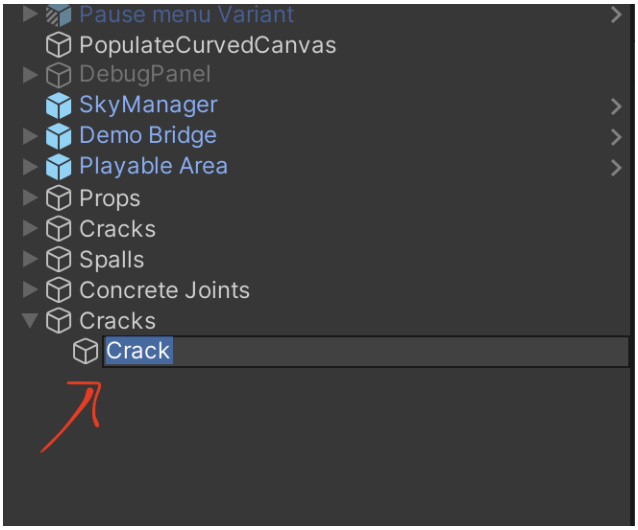


Rename the GameObject to the damage you would like to detect and add ‘s’ to it to make it plural.

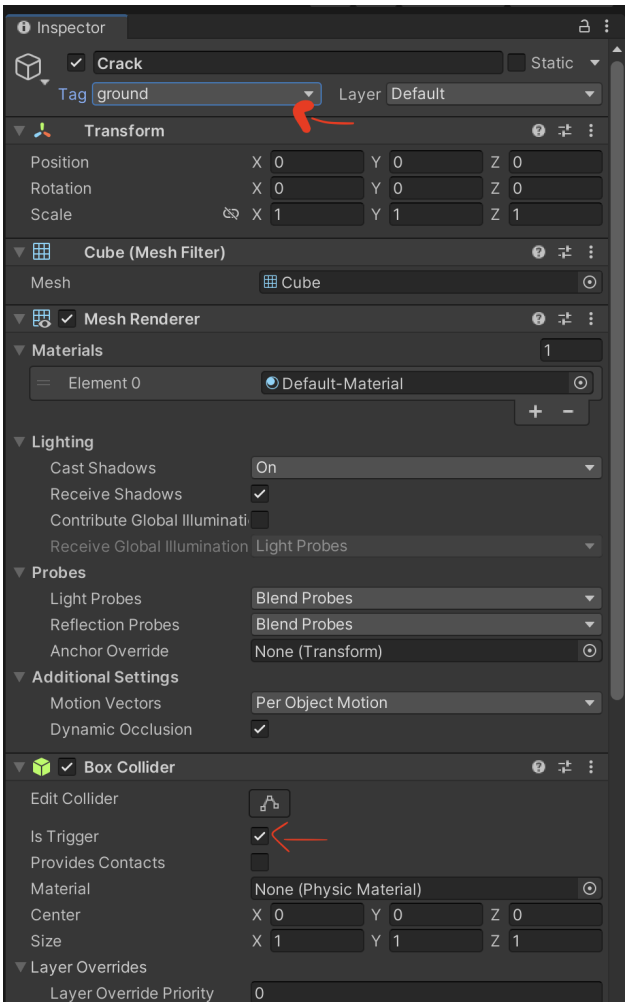


Making the Damage Detector

Then create a 3D cube and make it its child by right-clicking on the GameObject → ‘3D Object’ → ‘Cube’. Now rename the Cube into the appropriate damage, this time without the ‘s’ so it is single.

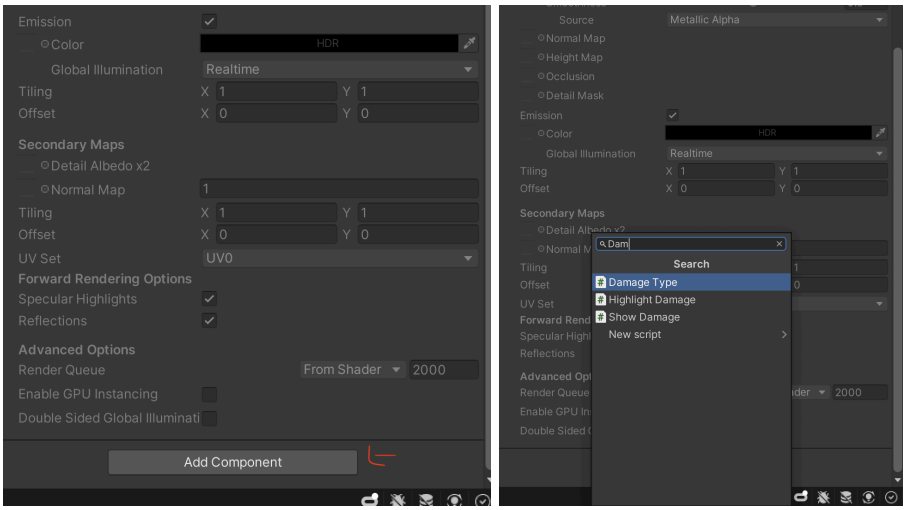


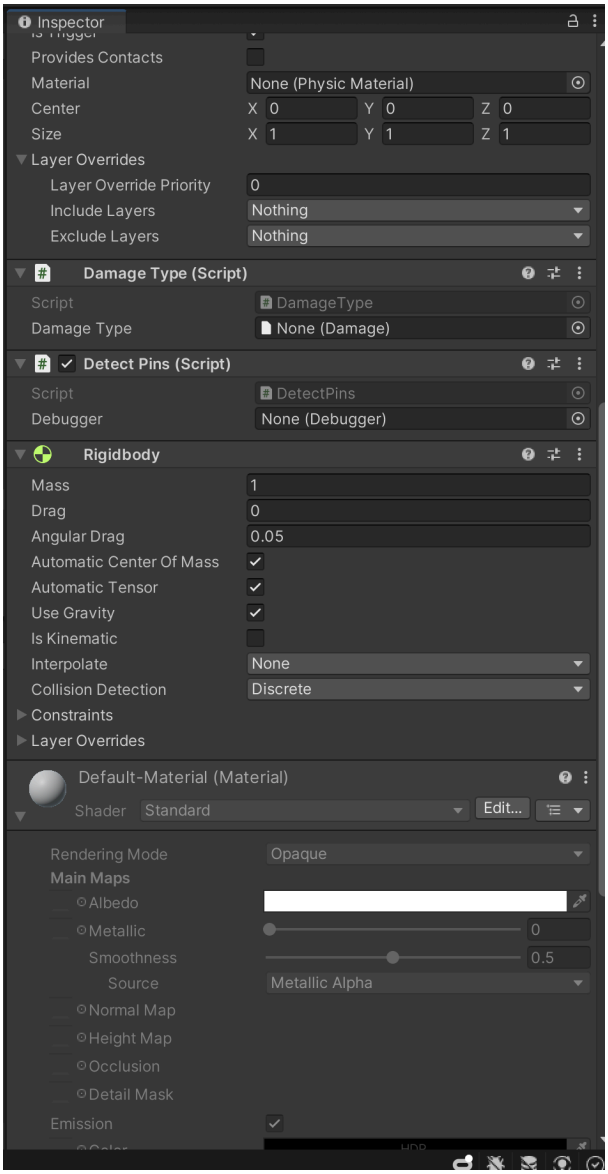
Then, click on the Cube and in the inspector change the Tag of this Cube to ‘ground’. Then under ‘Box Collider’, check the ‘Is Trigger’ toggle.



If you would like to add a color/material to the mesh renderer, refer to ‘Search For Materials’ and ‘Create New Materials’ in the previous ‘Adding Damages’ section to obtain the material you like.

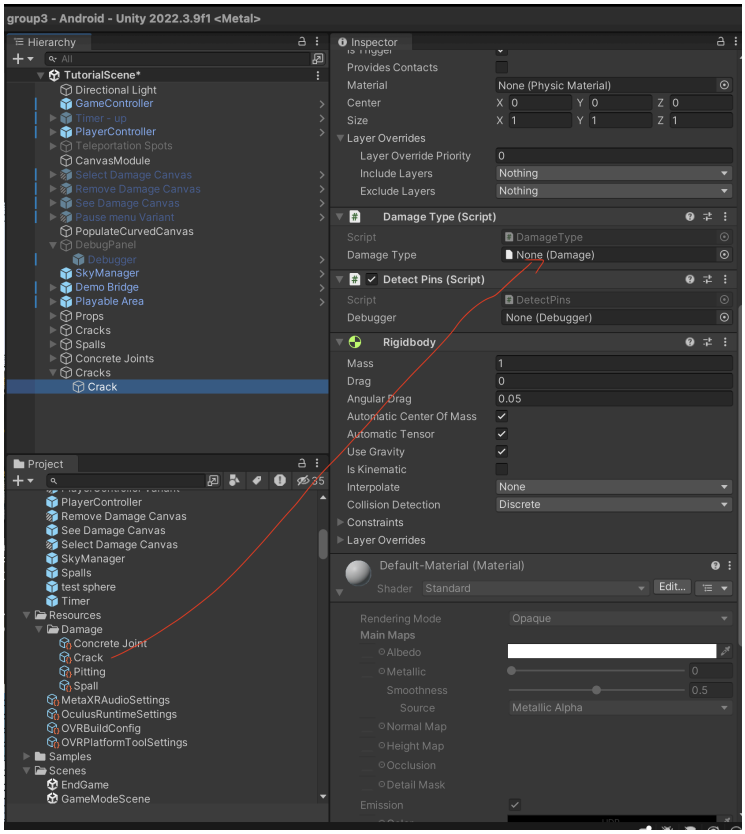
Now add ‘Damage Type’ and ‘Detect Pins’ scripts and ‘Rigidbody’ component by pressing on the ‘Add Component’ button and then searching for their name in the search bar.



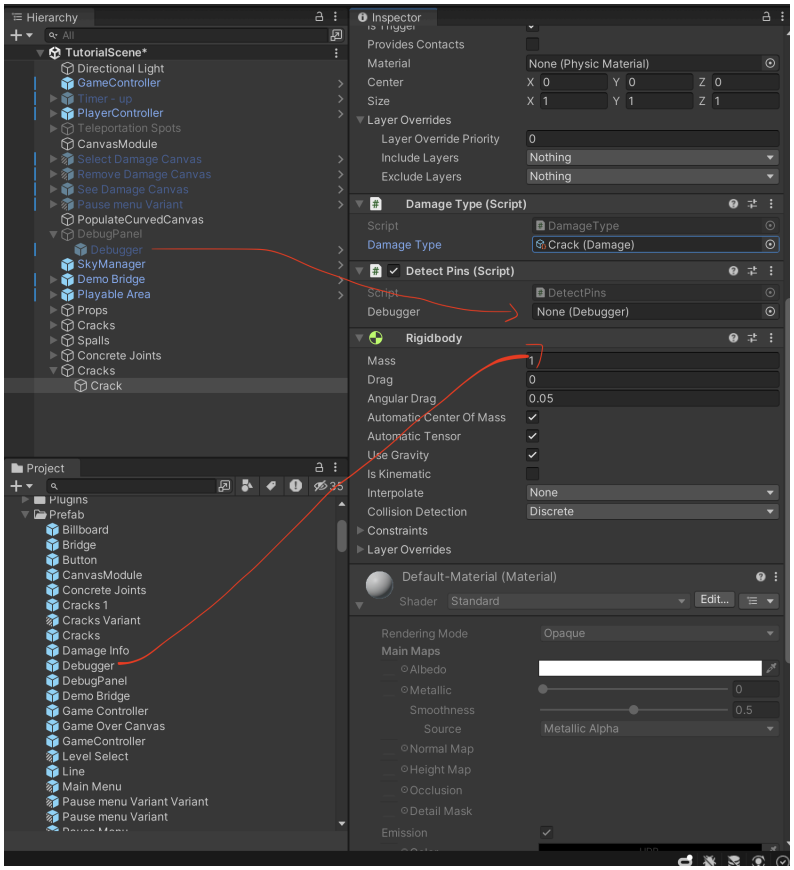


Initializing the Components

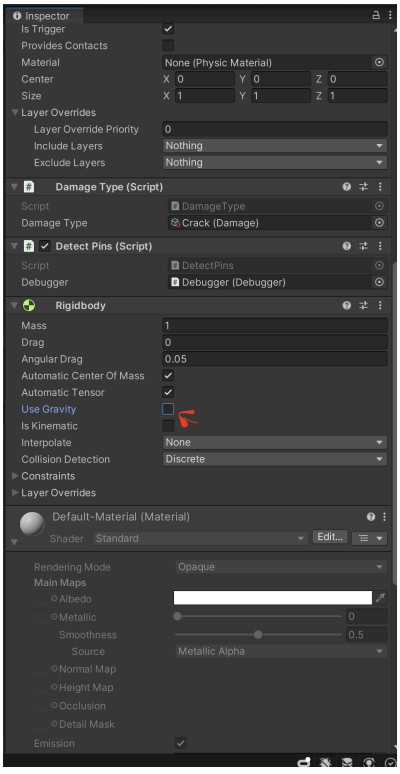
In the ‘Damage Type’ script add the damage you like which can be found in the ‘Resources’ → ‘Damage’ folder.



In the ‘Detect Pins’ script you can add the Debugger GameObject in the ‘Debugger’ section. The Debugger GameObject can be found either in the Hierarchy as the child of ‘DebugPanel’ GameObject or it can also be found in the ‘Prefab’ folder and is called ‘Debugger’.

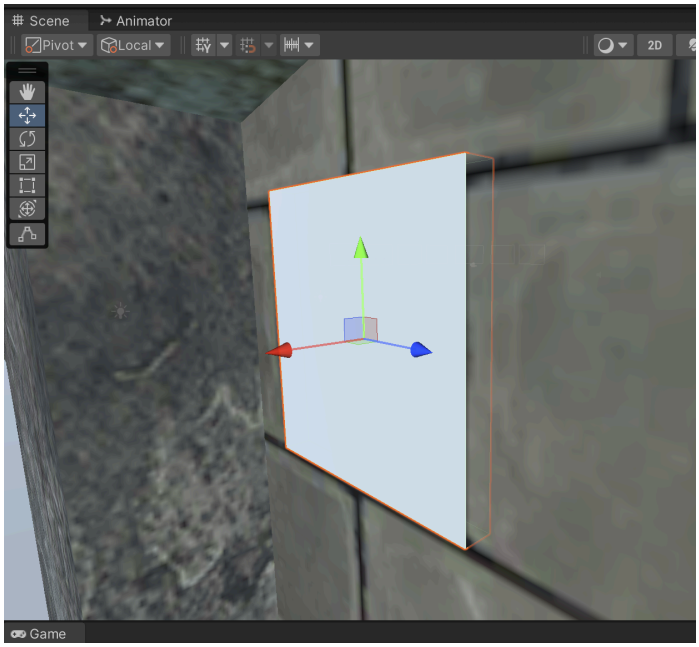


In the ‘Rigidbody’ component uncheck the ‘Use Gravity’ toggle.

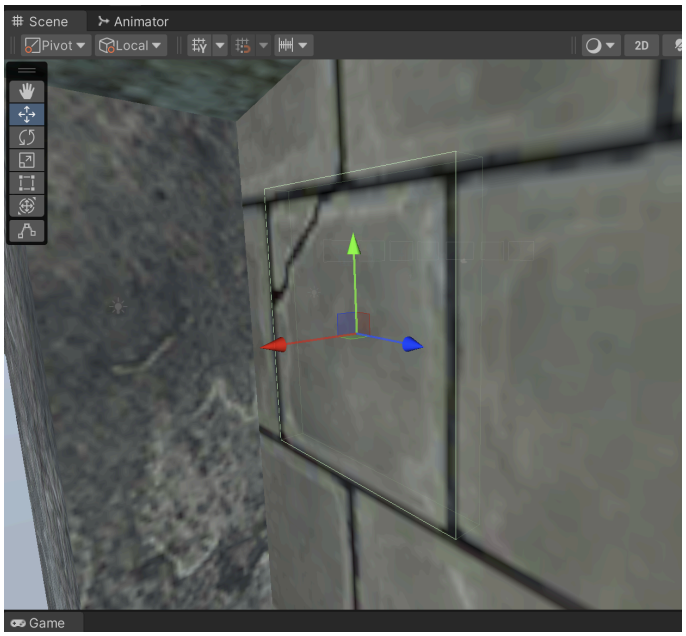


Positioning the Damage Detector

Now position the newly made Cube so that it is slightly above the damage on the bridge you would like to detect and scale it however necessary.



Then disable the ‘Mesh Renderer’ component of this Cube so that it can be enabled when the correct pin is put on it.

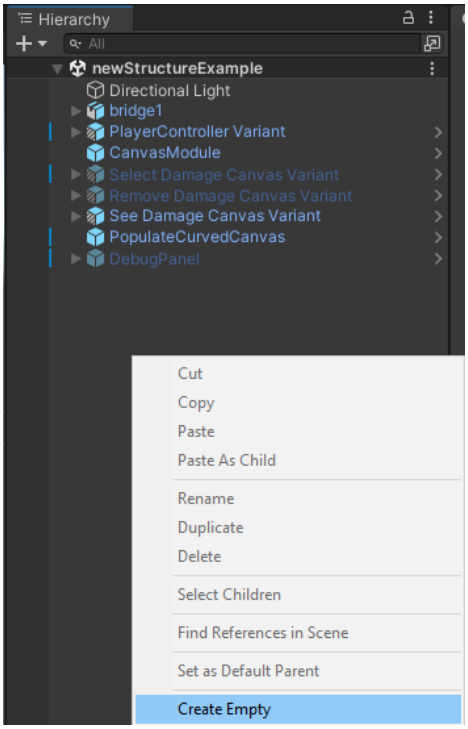


If you like to add more damage detectors for the same damage type, repeat from step ‘Making theDamage Detector’ up to and including the previous step ‘Positioning the Damage Detector’.

Adding Teleport Waypoints

Empty GameObject for Organization (Optional)

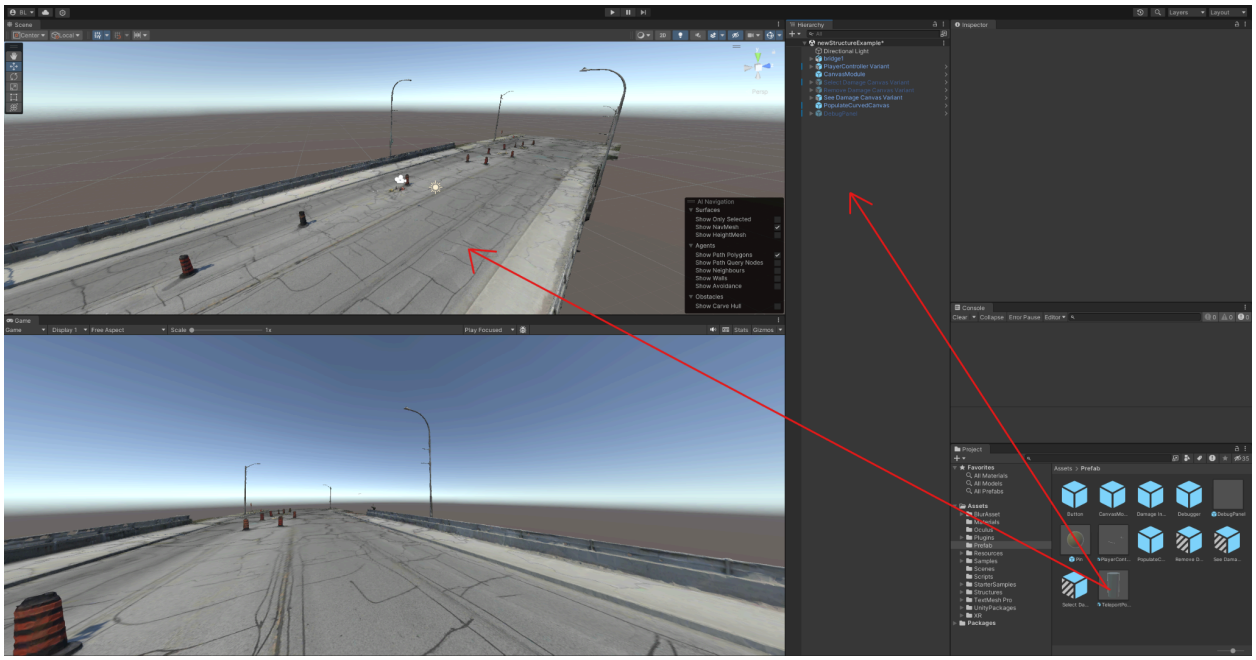
Make an empty GameObject in the Hierarchy window by right-clicking and selecting Create Empty.



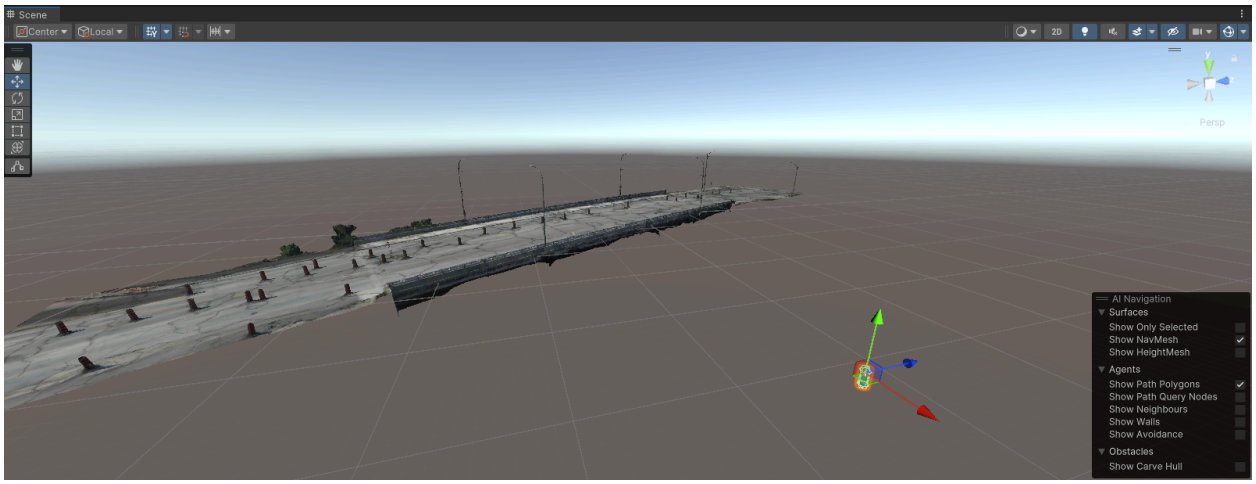
Rename the GameObject if you want to something more descriptive (like Teleport Waypoints). Then when adding waypoints, drag and drop them into this GameObject to make them child objects.

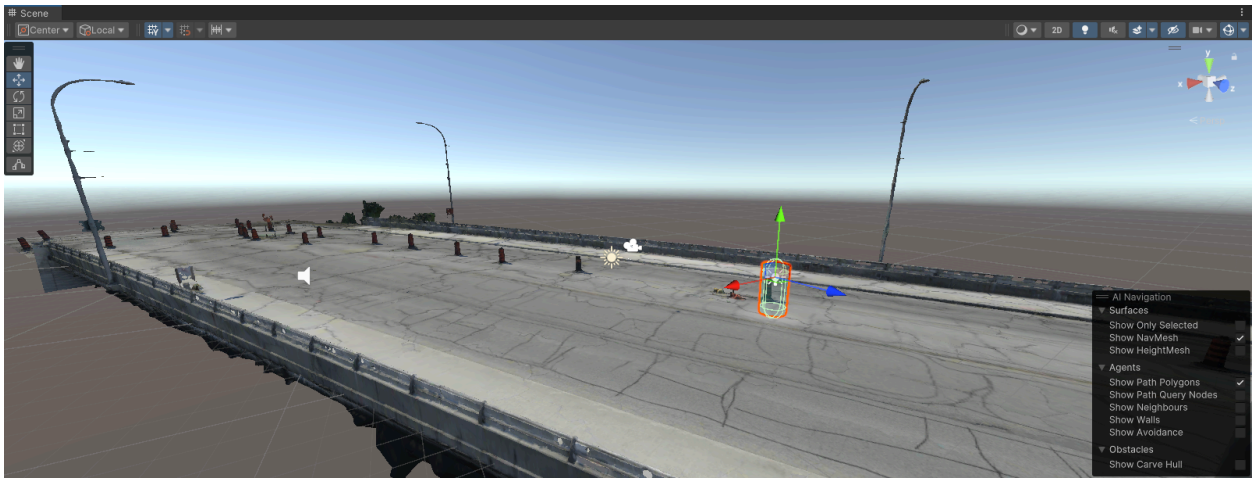
Adding TeleportPoint Objects

In the Project window, go to the Prefab folder and find the TeleportPoint Variant object. Drag and drop as many of these as you want into the Scene or Hierarchy window.



The position of the TeleportPoint GameObjects can be adjusted in the Inspector window by changing the Position (under the transform component) or by dragging it in the scene.

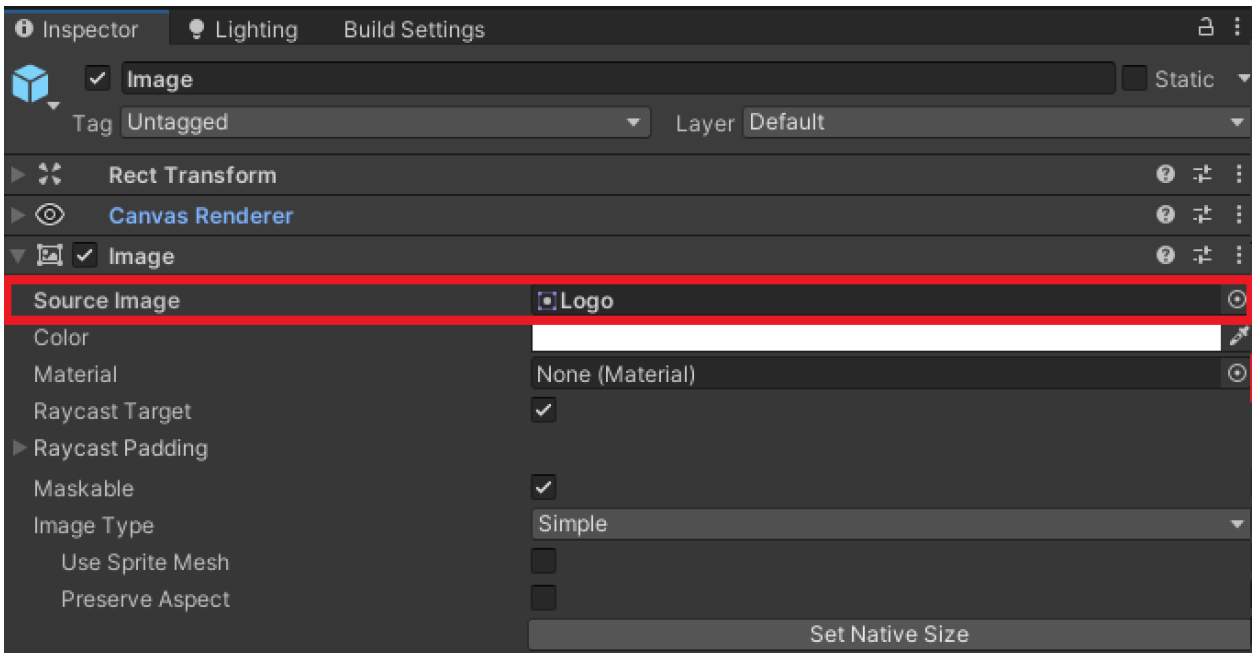




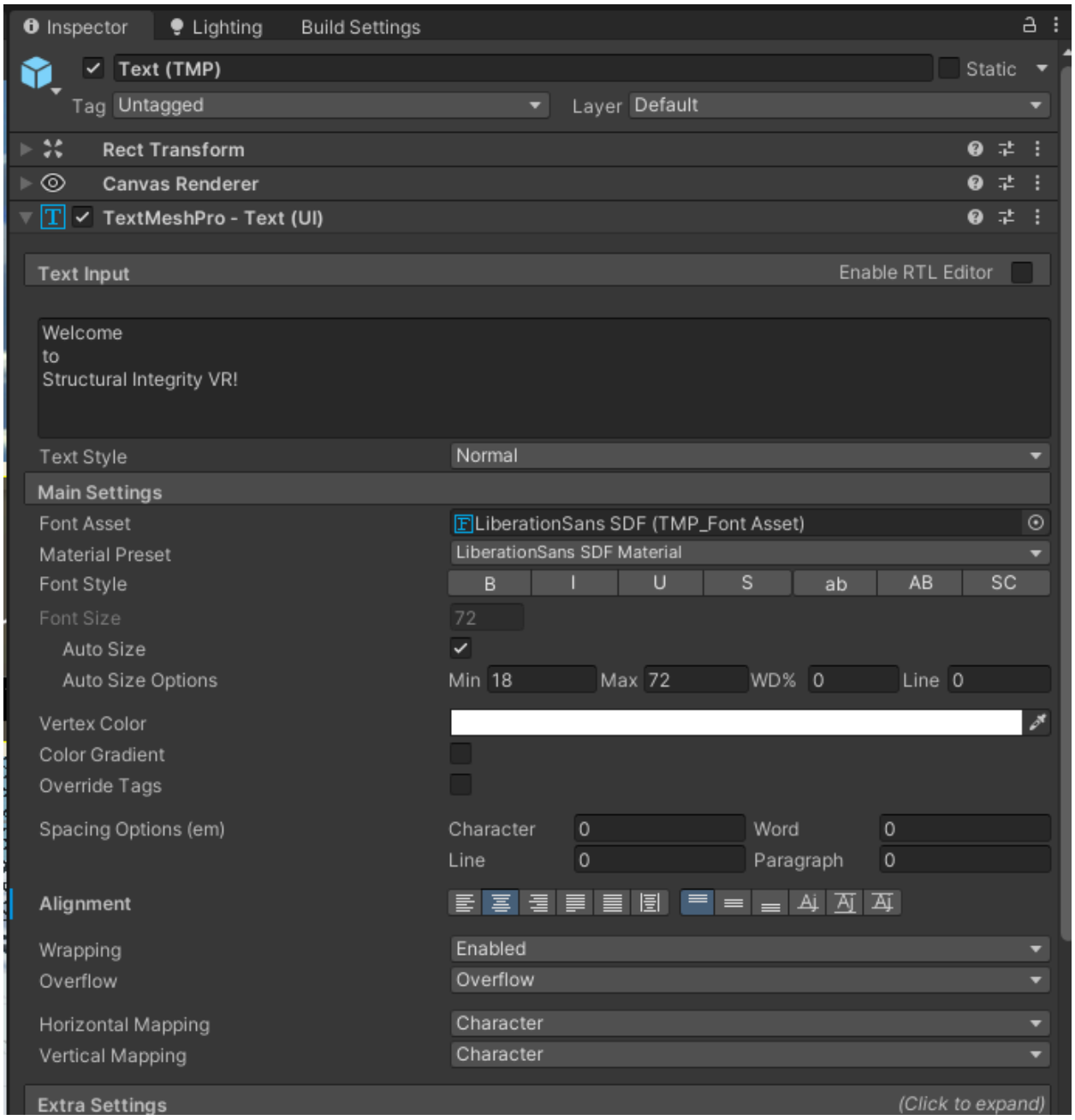
Adding Props To The Scene

In the Prefab folder, there is a pre-made **Billboard** prop that can be used to display information as a text or image in the game environment. Simply drag the prefab into the scene and it can then be modified from the hierarchy.

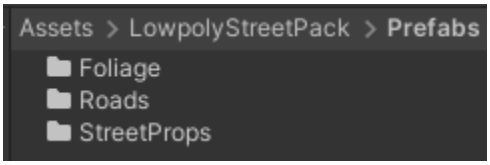
To modify the contents of the Billboard, simply go into the Image child and add an image through the inspector or disable it if not needed:



As for text, simply go into the Text (TMP) child and you can directly modify the text from the Inspector:

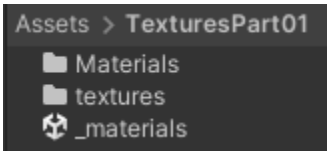


Street assets are also included under Assets/LowpolyStreetPack/Prefabs



Simply drag and drop the desired prefab into the scene.

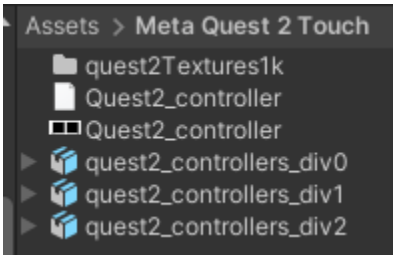
Some basic brick, asphalt, grass, and ground textures are also included under Assets/TexturesPart01



The fence prefabs are found under Assets/Assets_FenceChained/prefabs

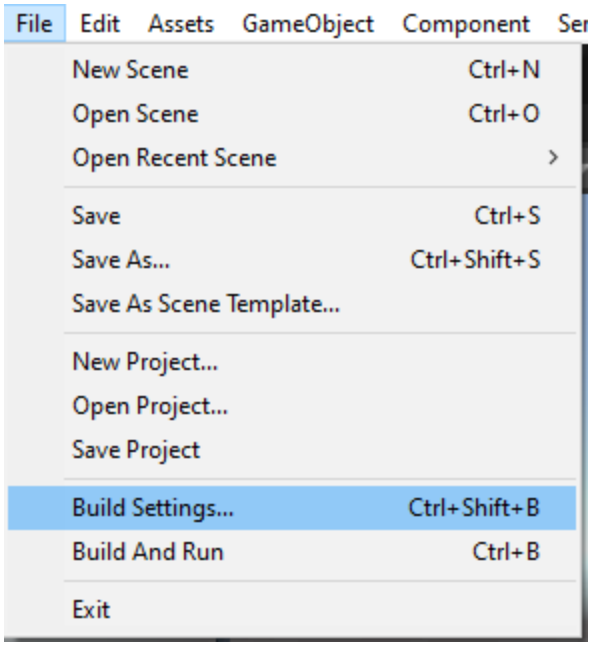
Simply drag and drop the desired prefab into the scene as needed.

Additionally, Quest 2 controller models can be found under Assets/”Meta Quest 2 Touch”

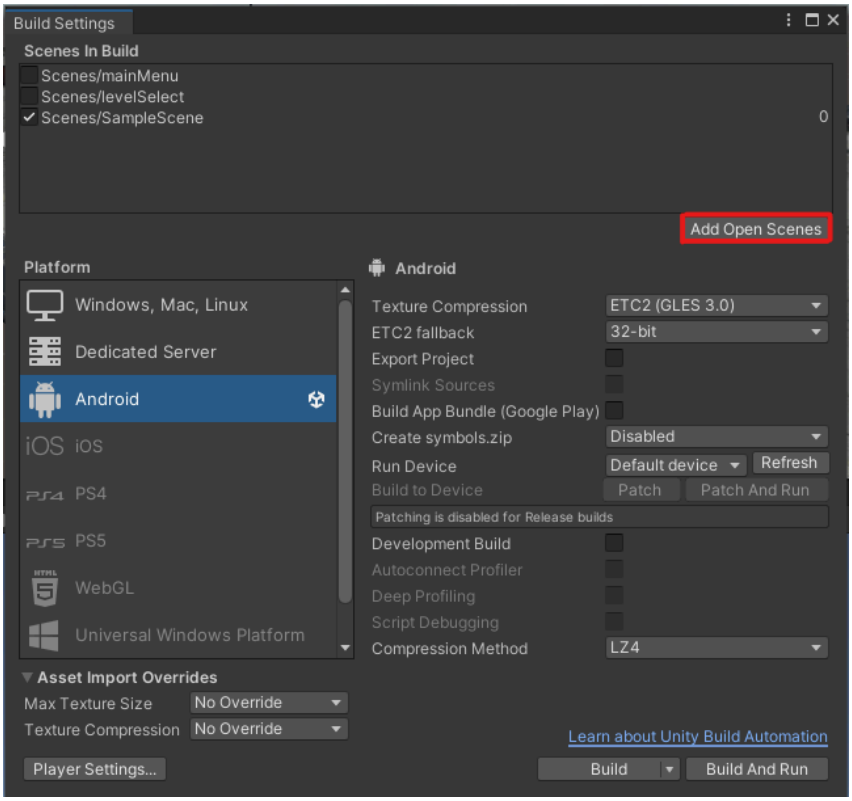


Adding New Scene to Build

In the top menu, click on File → Build Settings



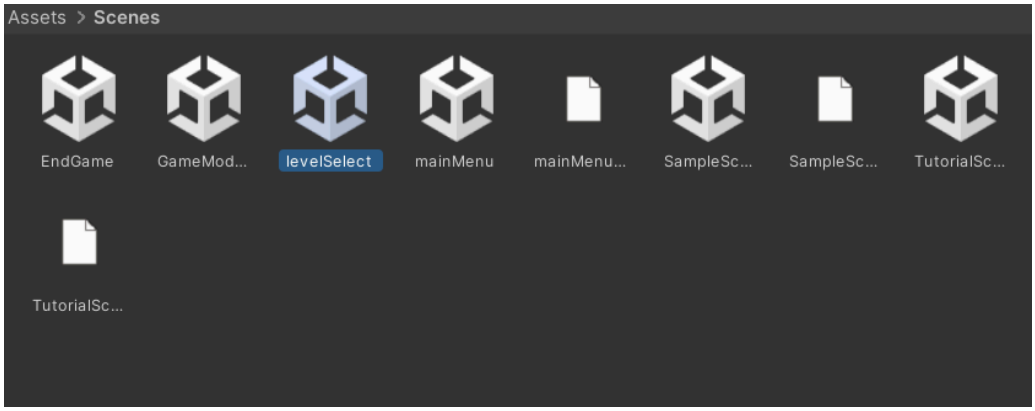
Click on Add Open Scenes.



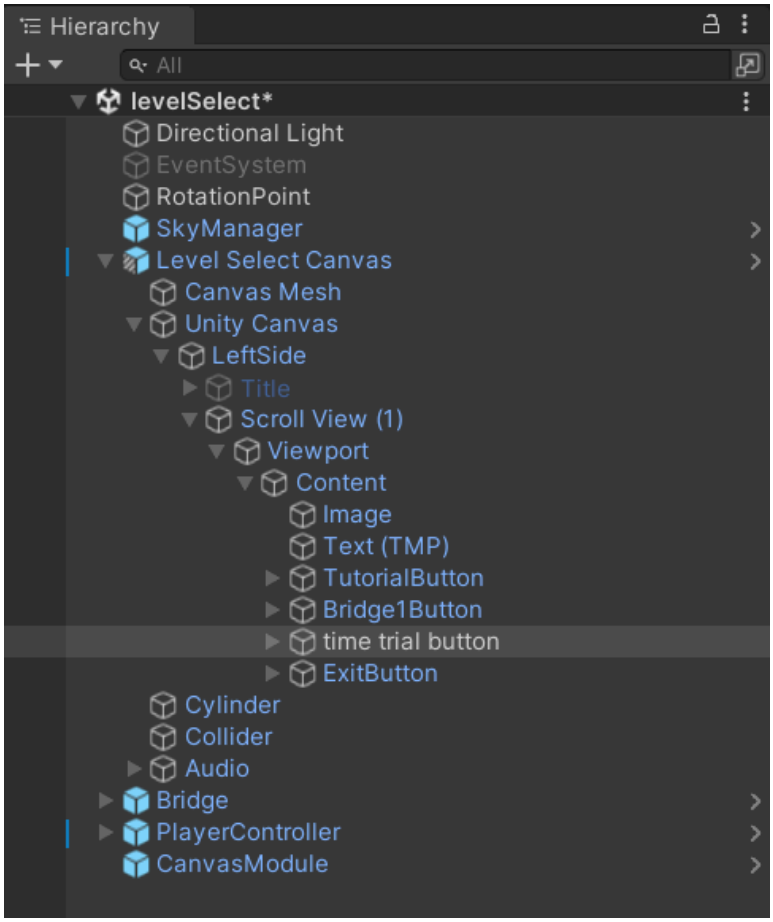
Ensure all Scenes have a checkmark next to them and Scenes/mainMenu is the first selection (0-indexed).

Adding to Level Selection

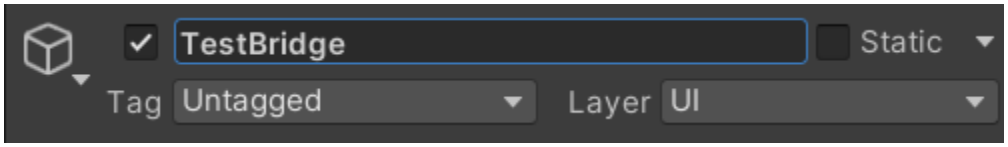
In the scene folder switch to the scene titled “levelSelect”



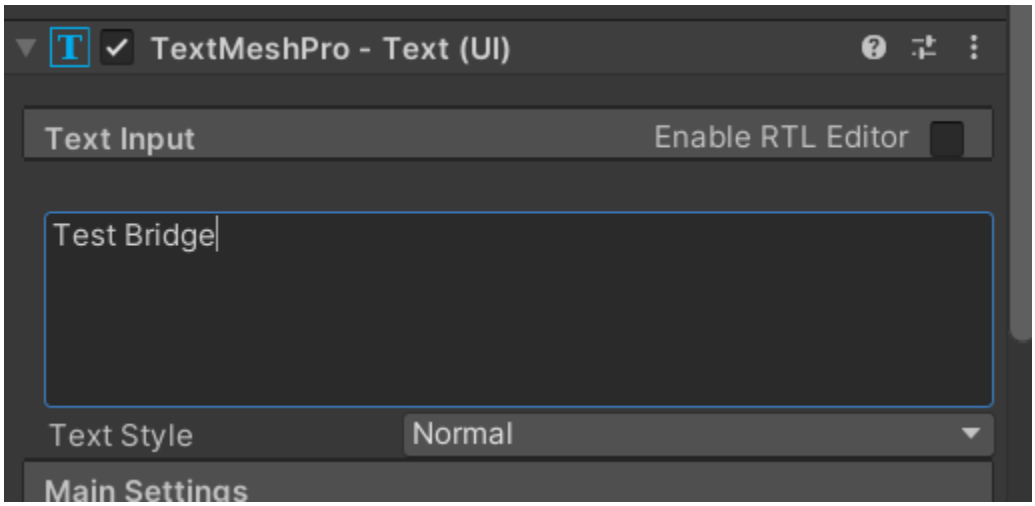
In the hierarchy expand the Main Menu → Unity Canvas → LeftSide → Scroll View (1) → Viewport → Content


Hierarchy
All
levelSelect*
Directional Light
EventSystem
RotationPoint
SkyManager
Level Select Canvas
Canvas Mesh
Unity Canvas
LeftSide
Title
Scroll View (1)
Viewport
Content
Image
Text (TMP)
TutorialButton
Bridge1Button
time trial button
ExitButton
Cylinder
Collider
Audio
Bridge
PlayerController
CanvasModule

Duplicate the Bridge1Button and change its name to the correct Bridge name.



Expand the new button and change the text.



Open the Script titled “ButtonController”

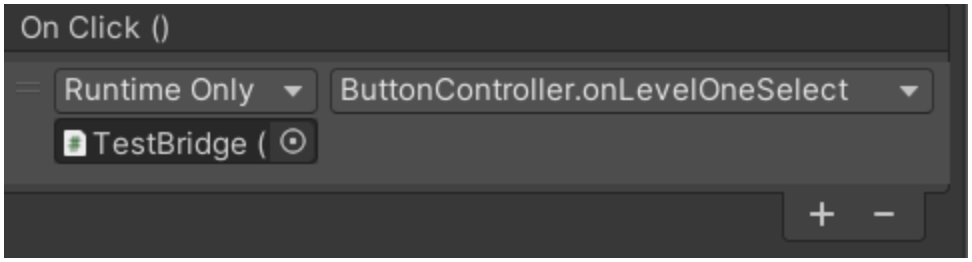
Create a new function called onLevelxSelect, where x is the bridge number you're adding (see below)

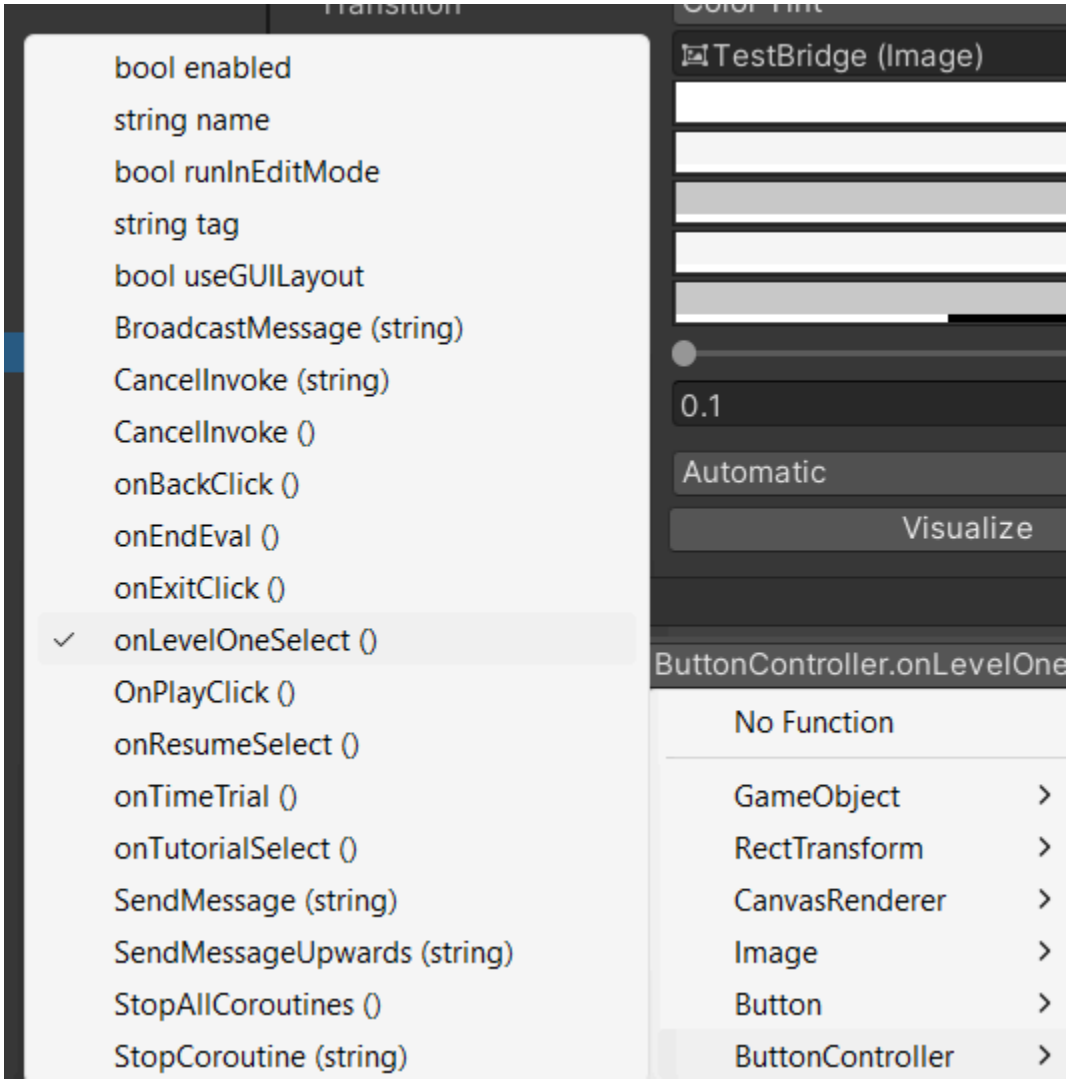
```
public void onLevelxSelect()
{
        SceneManager.LoadScene(“SceneName”);
}
```

Change SceneName to be the name of the scene which you created earlier. for example Time Trail looks like this.

```
 public void onLevelOneSelect()
 {
        SceneManager.LoadScene(“GameModeScene”);
 }
```

Finally back in the button object, that you duplicated change the onclick function to reflect the name of the function you just created.





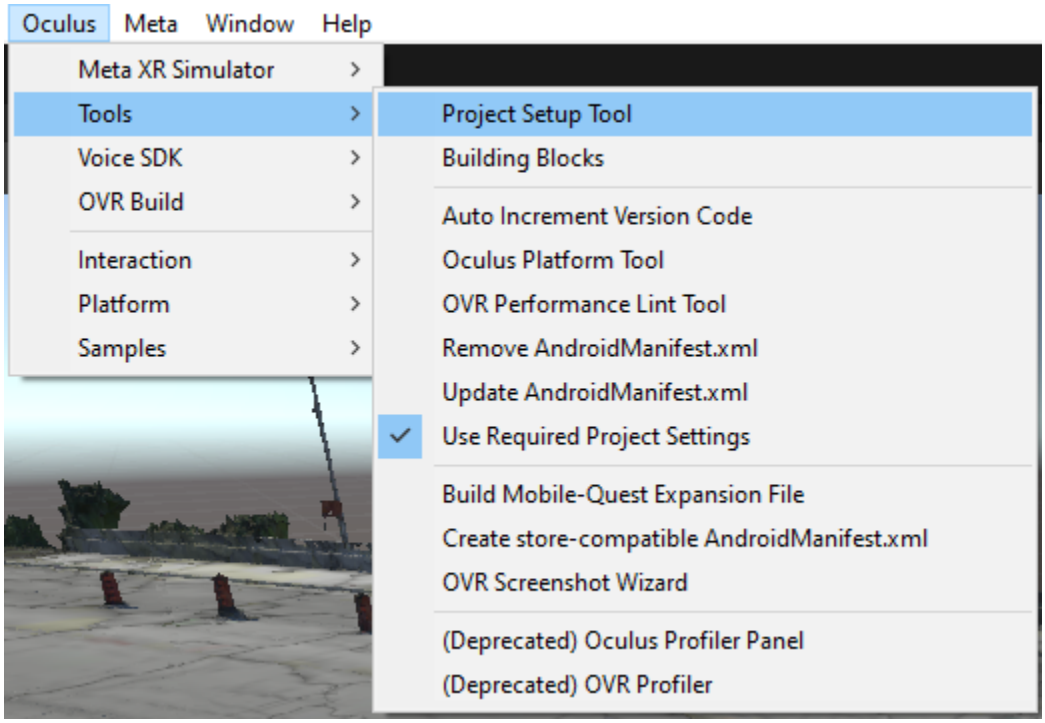
Now you should be able to load your scene from the Level Select Scene.

Building the Project

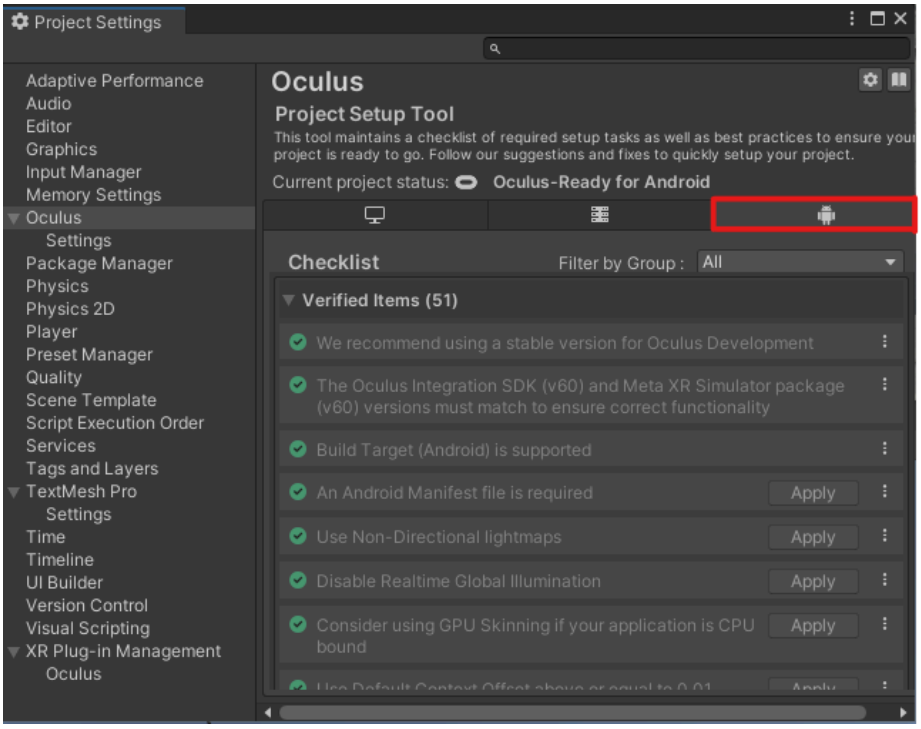
VR Build

Running the Oculus Project Setup Tool

In the top menu, click Oculus → Tools → Project Setup Tool

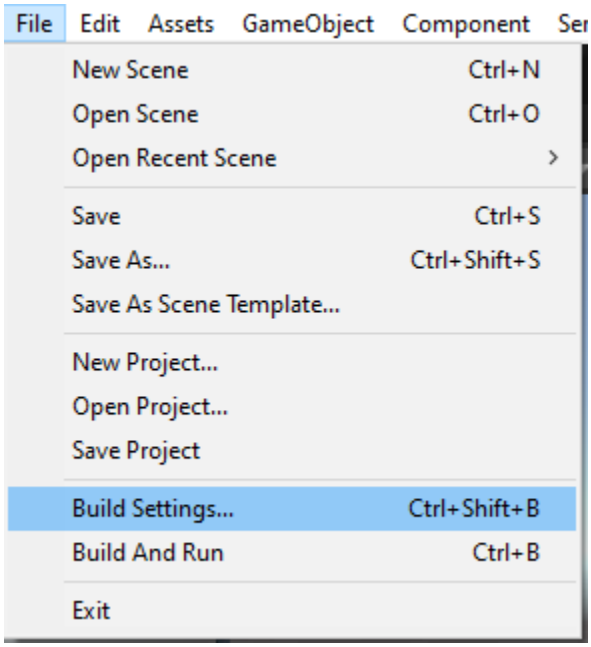


Ensure the tool is modifying the Project Settings to optimize for Android and apply all of the items the tool recommends.

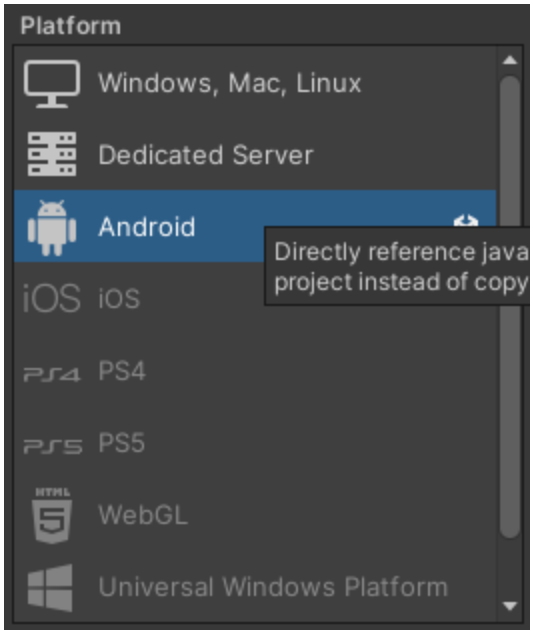


Making a Build

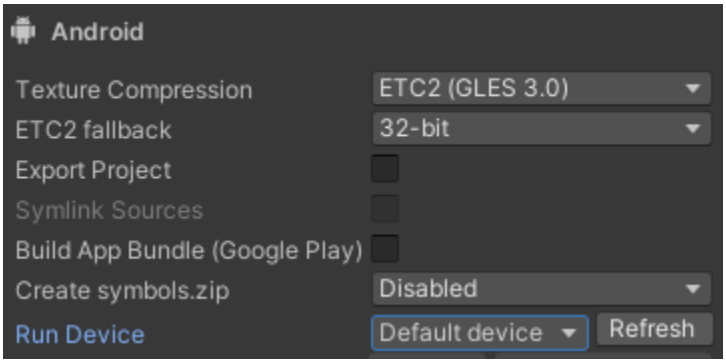
In the top menu, click on File → Build Settings



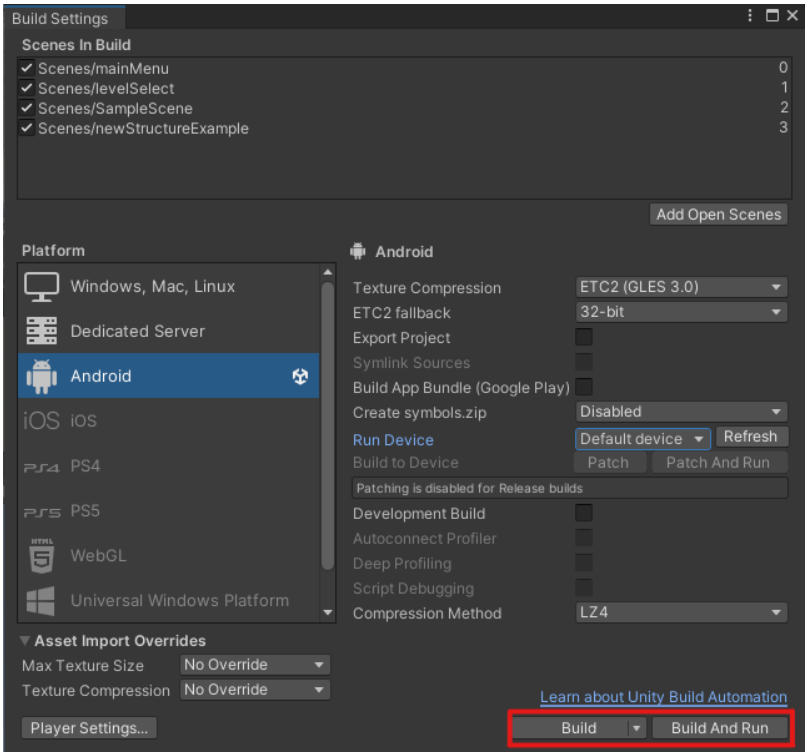
Ensure the current Platform is Android.



Ensure the Quest headset is in Developer Mode and it is plugged into the computer. Under the Run Device dropdown, select the Quest Headset (if it doesn't appear, try pressing the Refresh button).

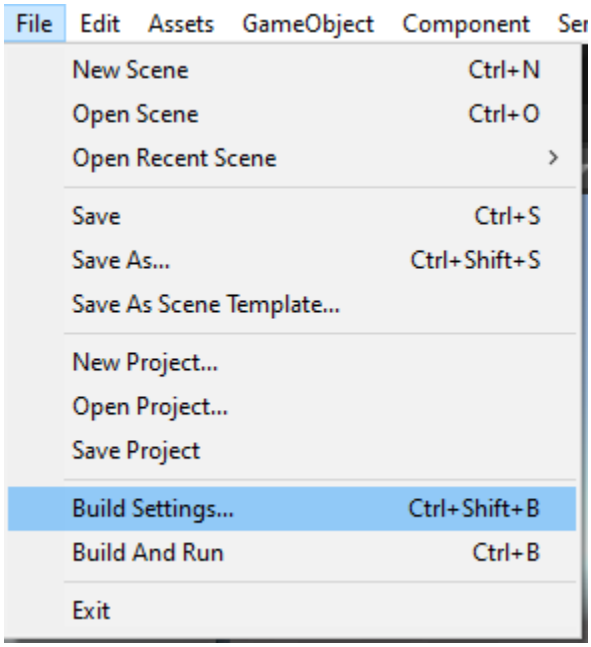


Press the Build or Build and Run button.

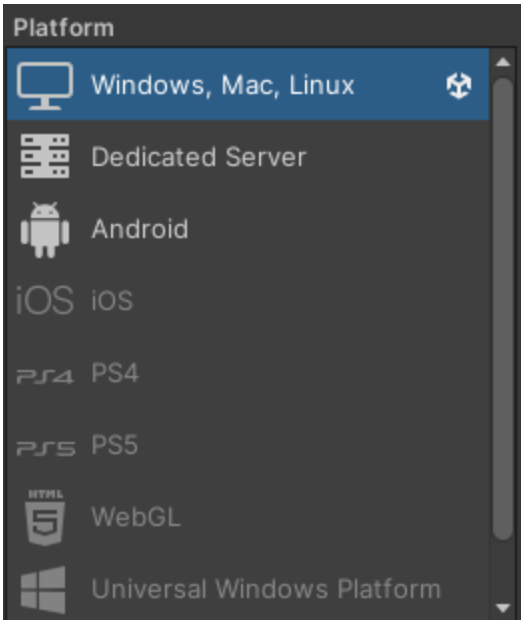


PC Build

In the top menu, click on File → Build Settings



Ensure the current Platform is Windows, Mac, Linux.



Press the Build or Build and Run button (Note: Ours is grayed out since the scenes all have the VR Components).