Warmerise | Red vs Blue

CUSTOM MAP TUTORIAL

Introduction

Custom map system allows players to create their own maps in Unity and upload them to the game.

This tutorial features a step-by-step guide on how to setup a new map, so it can be used in the game.

Requirements:

- Unity Game Engine (version equal or above 2020.3) (https://unity3d.com/)
- Understanding Of Unity 3D And Level Design

The following topics will be covered:

- Installing The Export Package
- Placing Special Objects In The Map
- Adding Ladders
- Adding Doors
- Adding Destructible Objects
- Adding Water Plane
- Map Lighting
- Optimization Tips
- Supported Shaders
- Exporting The Map
- Testing The Map
- Uploading The Map
- Publishing The Map
- Updating The Map

Installing The Export Package

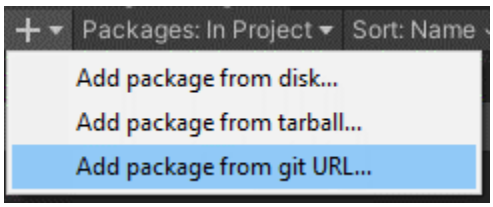
Export package is a combination of necessary scripts and files required to prepare and export the map.

The package is hosted on GitHub.

Below are the steps to install it using Unity's Package Manager:

- Install Git client
- In your Unity project go to Window -> Package Manager

- In the top left corner click (+) symbol -> Add package from git URL…



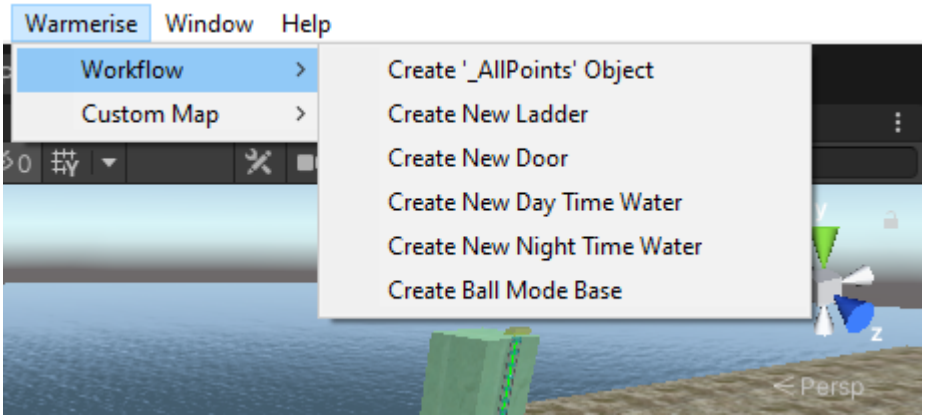
- Paste the URL below and click Add

  https://github.com/nsdesigngames/WarmeriseMapExport.git

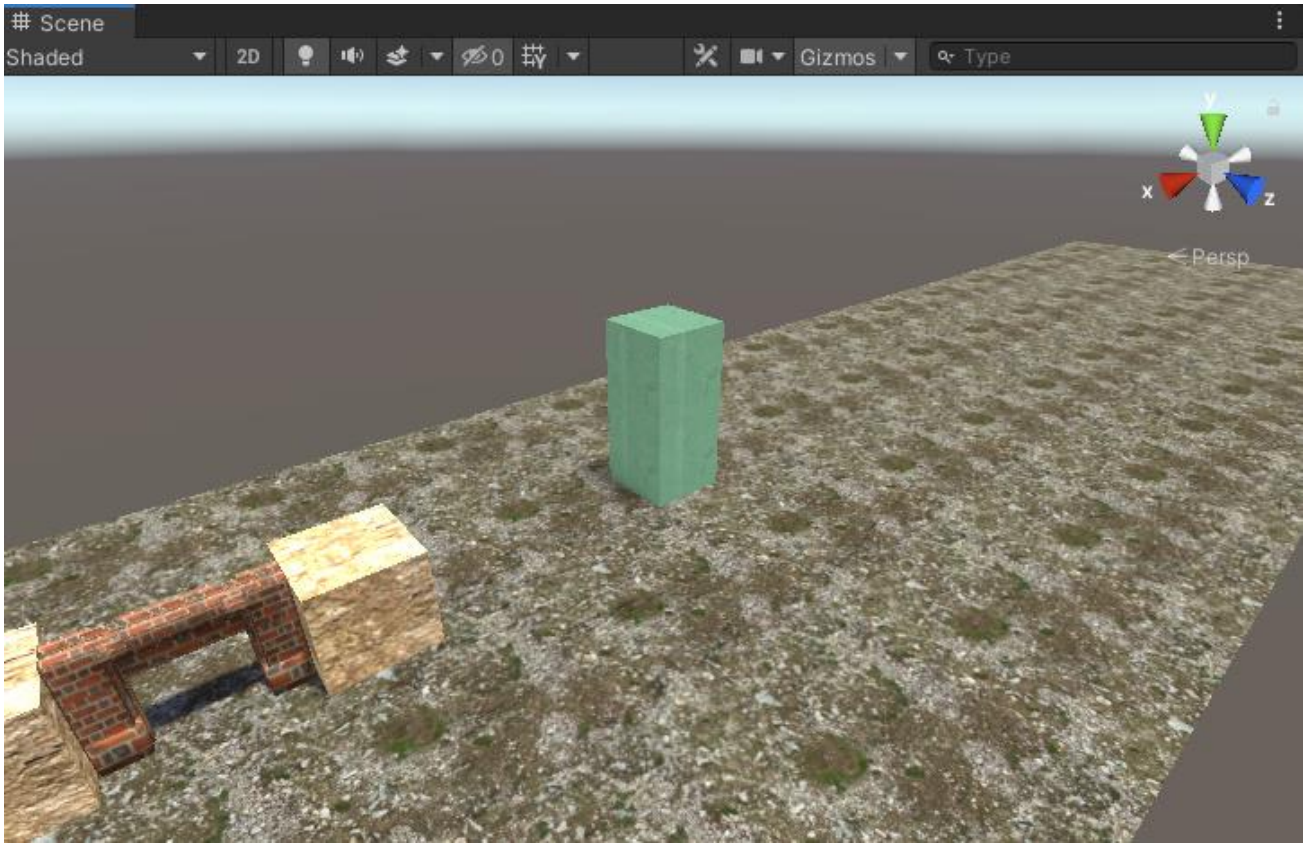
  If there are no errors in the console, the package should now be installed in your project.

Special Objects

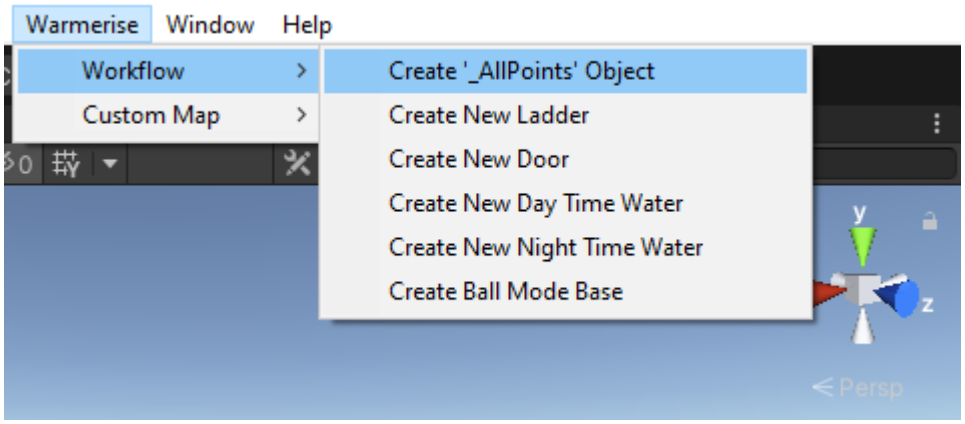
After installing the export package, the new menu item with the name ‘Warmerise’ should appear, which contains all the options needed to prepare and export the map:

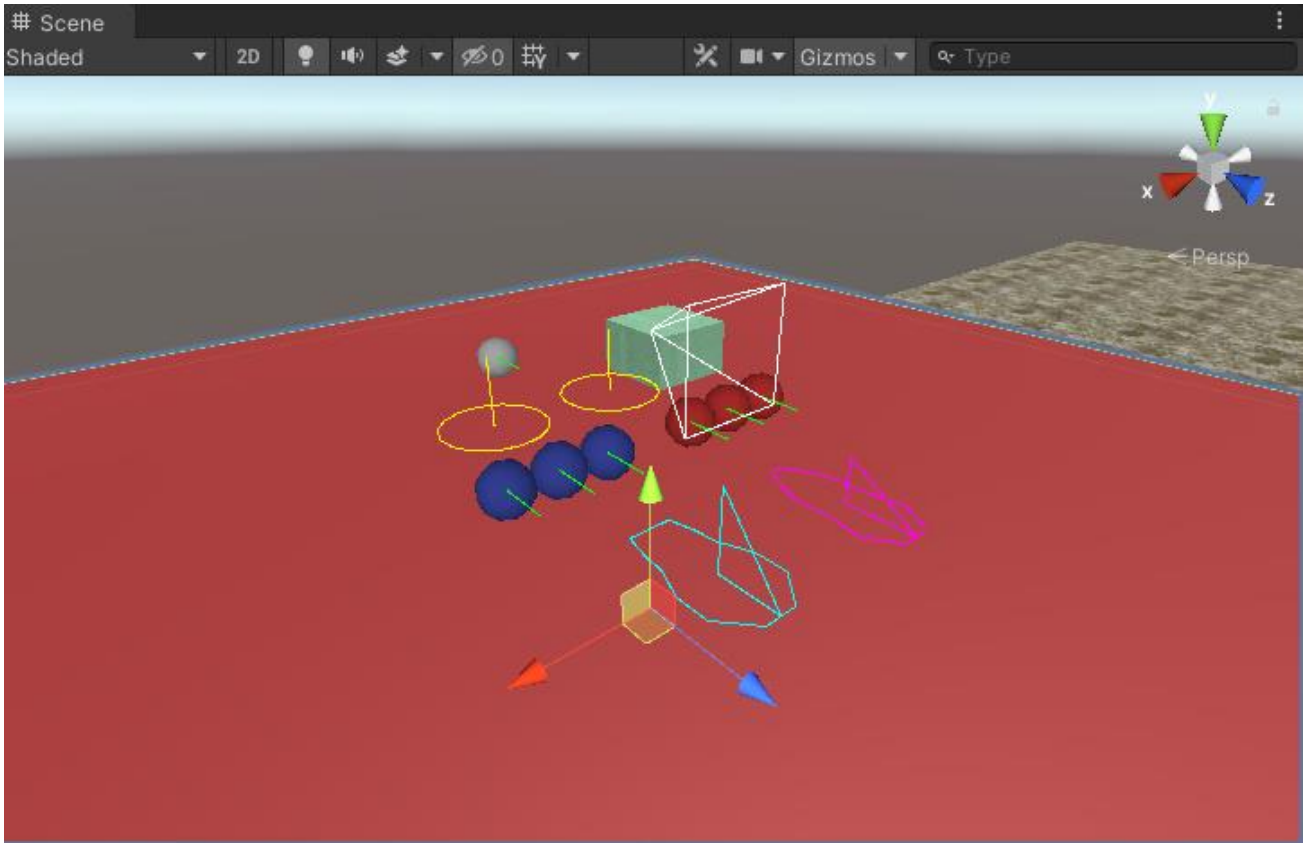


- Open the Scene with your map

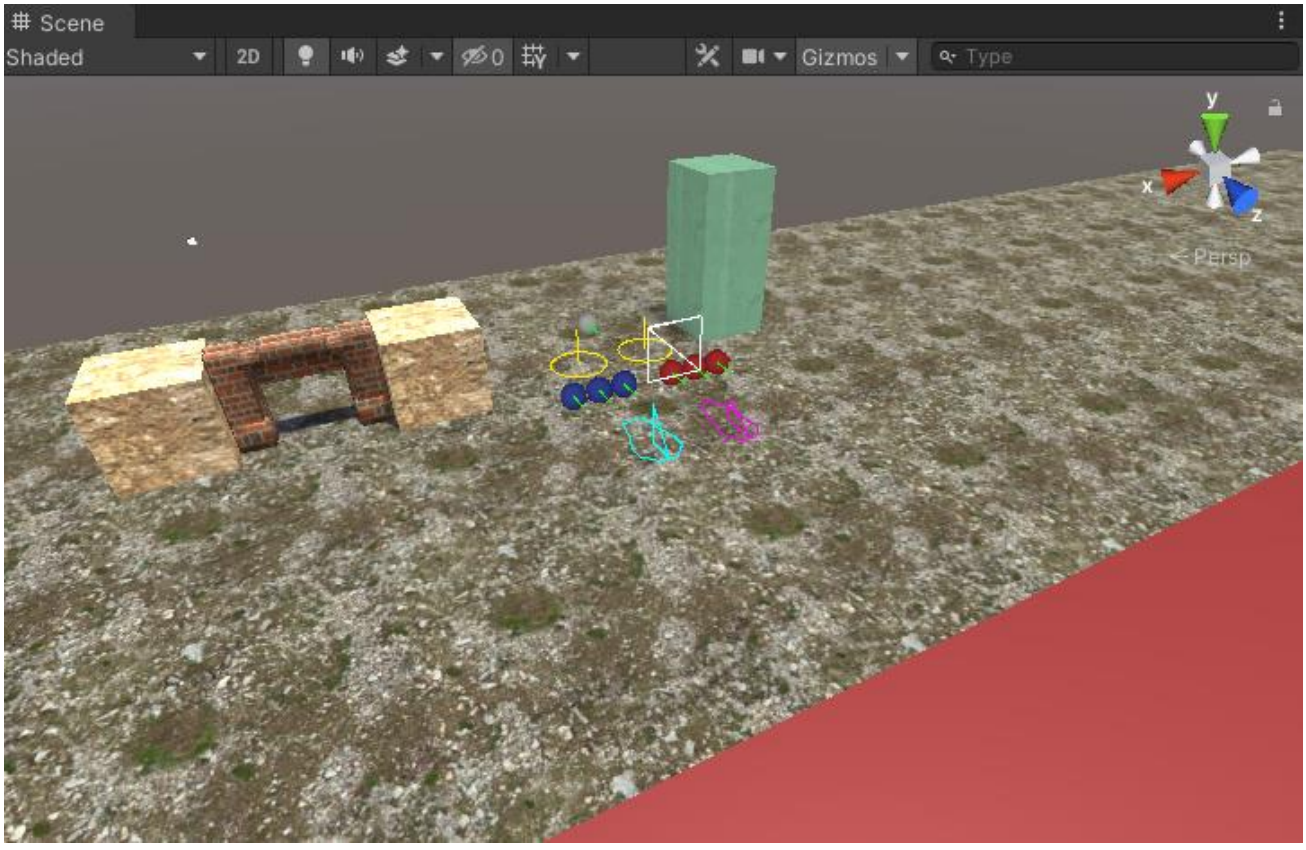


- Create the main key object by clicking Warmerise -> Workflow -> Create '_AllPoints' Object

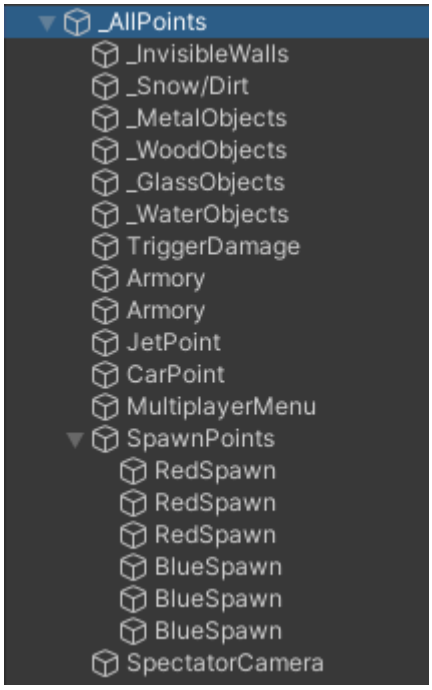




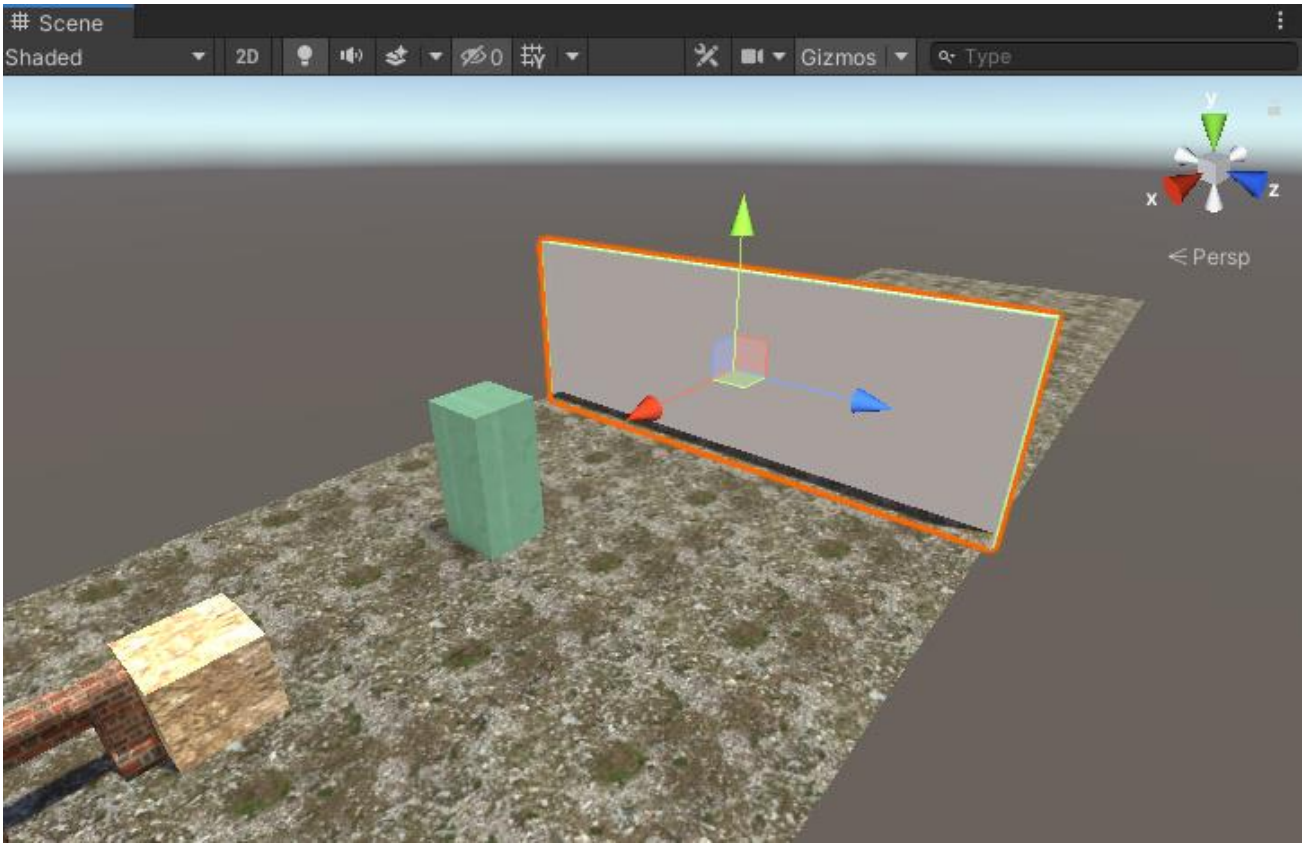
- Adjust the object position until it matches the surface of the map:



- Looking at the hierarchy view, you can see that the object '_AllPoints' has many objects inside it, with each having its own purpose:



_InvisibleWalls – this object should contain the objects with collider component, which will be used as an invisible wall to limit the boundaries of the map. [This Object Can Be Removed But Should Not Be Duplicated]



_Snow/Dirt – this object should contain map objects that need to exhibit snow/dirt surface behavior (predominately used for detecting bullet hit particles and/or footstep sounds). [This Object Can Be Removed But Should Not Be Duplicated]

_MetalObjects – this object should contain map objects that need to exhibit metal surface behavior (predominately used for detecting bullet hit particles and/or footstep sounds). [This Object Can Be Removed But Should Not Be Duplicated]

_WoodObjects – this object should contain map objects that need to exhibit wood surface behavior (predominately used for detecting bullet hit particles and/or footstep sounds). [This Object Can Be Removed But Should Not Be Duplicated]

_GlassObjects – this object should contain map objects that need to exhibit glass surface behavior (predominately used for detecting bullet hit particles and/or footstep sounds). [This Object Can Be Removed But Should Not Be Duplicated]

_WaterObjects – this object should contain map objects that need to exhibit water surface behavior (predominately used for detecting bullet hit particles). [This Object Can Be Removed But Should Not Be Duplicated] *NOTE: 'Day Time Water' and 'Night Time Water' do not need to be placed inside this object, as they are assigned as a water surface automatically.*

TriggerDamage – this object inflicts damage to players on contact. One of the uses, is placing it under the map, so players who fall from the boundaries are instantly killed. [This Object Can Be Duplicated or Removed]

Armory – point where the Armory Spot will be placed in the game (good practice is placing at least one armory spot for each team, or having just one armory spot in the middle of the map) [This Object Can Be Duplicated or Removed Until 1 is Remaining]

JetPoint – position where Jet Ship will be spawned [This Object Can Be Duplicated or Removed]

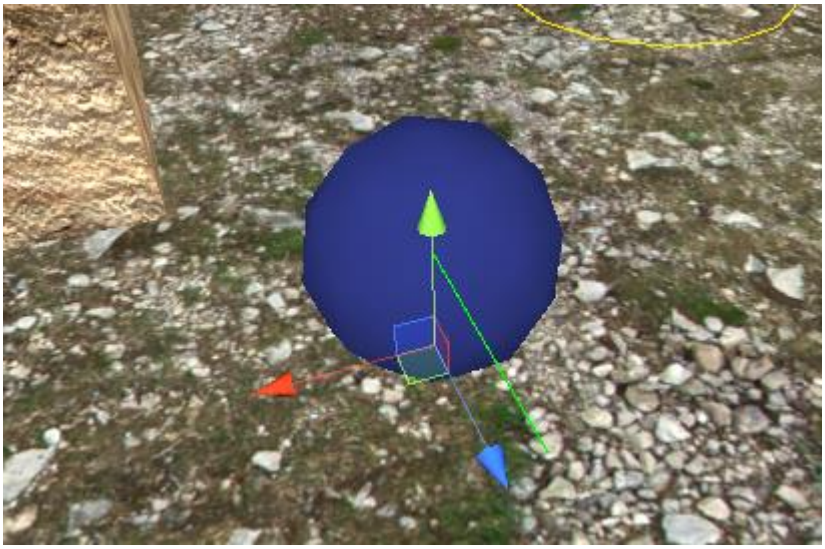
CarPoint – position where Humvee will be spawned [This Object Can Be Duplicated or Removed]

MultiplayerMenu – position where the welcome camera will be placed (for testing, move the Main Camera inside it and change its position and rotation to (0, 0, 0) so you can observe where it will point in the game). [This Object Should Not Be Duplicated or Removed]

NOTE: You might be tempted to place an image in front of the Camera, however it's better to point the Welcome Camera directly at the map, to give players a better preview.

SpawnPoints – object that contains spawn points for both teams [This Object Should Not Be Duplicated or Removed]

RedSpawn / **BlueSpawn** – spawn points for a Red Team and Blue Team respectively (the green line from the center indicates which direction the player will face during the spawn) [This Object Can Be Duplicated or Removed But At Least 3 For Each Team Must Be Placed]



NOTE: Ideally, try to place at least 8 spawn points per team and make sure they are 'spaced out' enough to prevent the crowded spawns, which could lead to base camping and reduced gaming experience.

SpectatorCamera – point where the flying spectator will be spawned (green line indicates the direction that the spectator will face during the spawn) [This Object Should Not Be Duplicated or Removed]

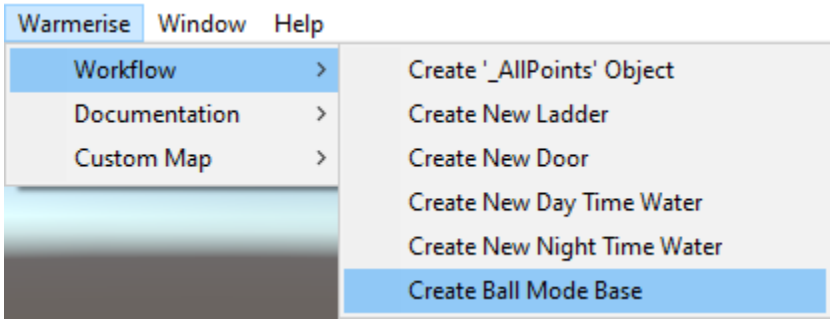
- Once the key objects are placed, the map is ready to be exported and tested in the game.

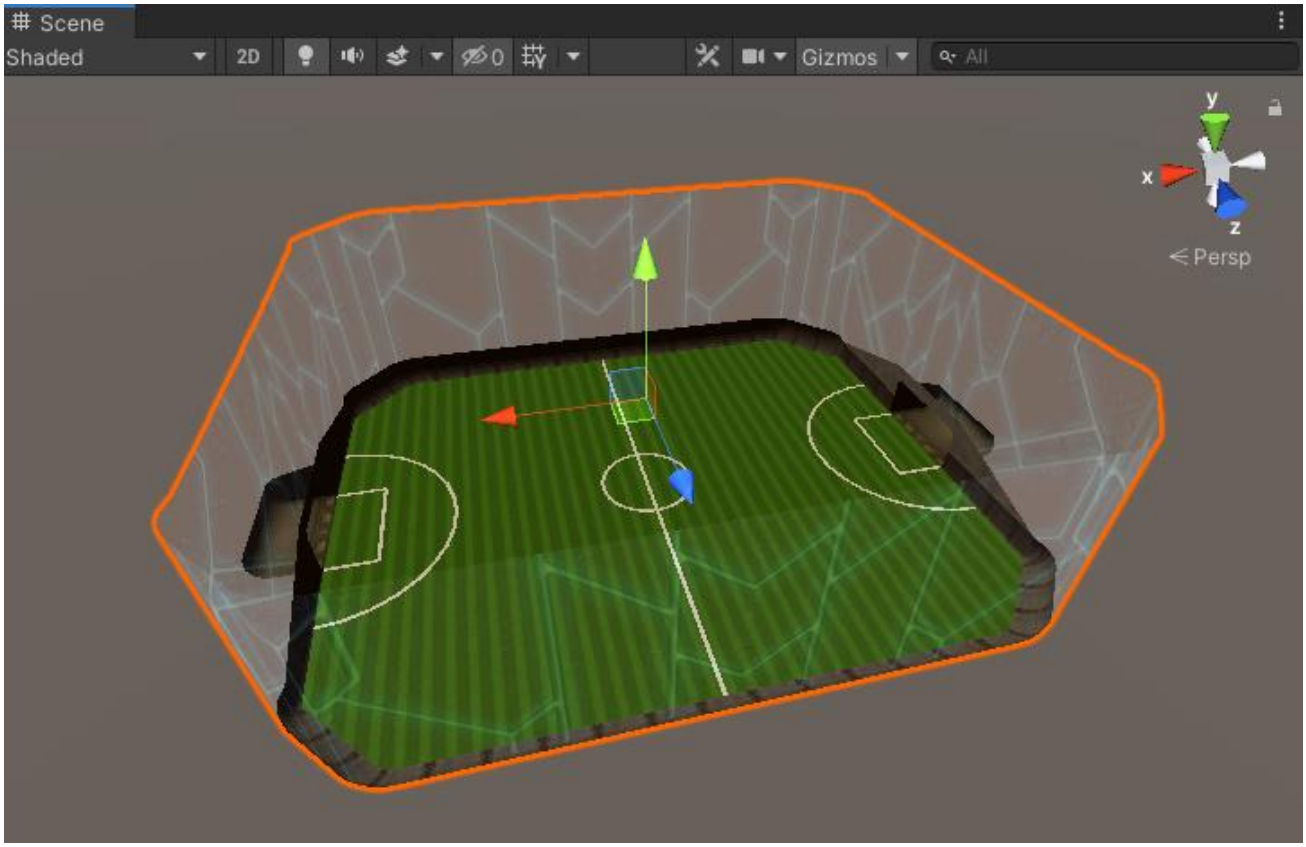
BALL MODE SPECIAL OBJECTS

Creating a map for Ball mode requires an additional key object, which is '_BallModeBaseMap'.

This object provides a layout reference, so you can see where the goals will be placed.

- Create new Scene
- Create base layout by clicking Warmerise -> Workflow -> Create Ball Mode Base



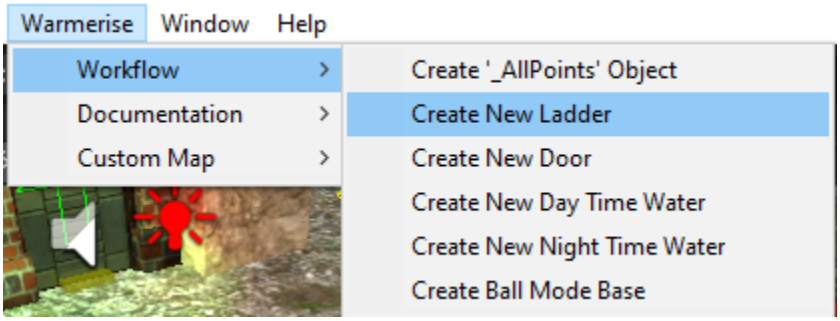


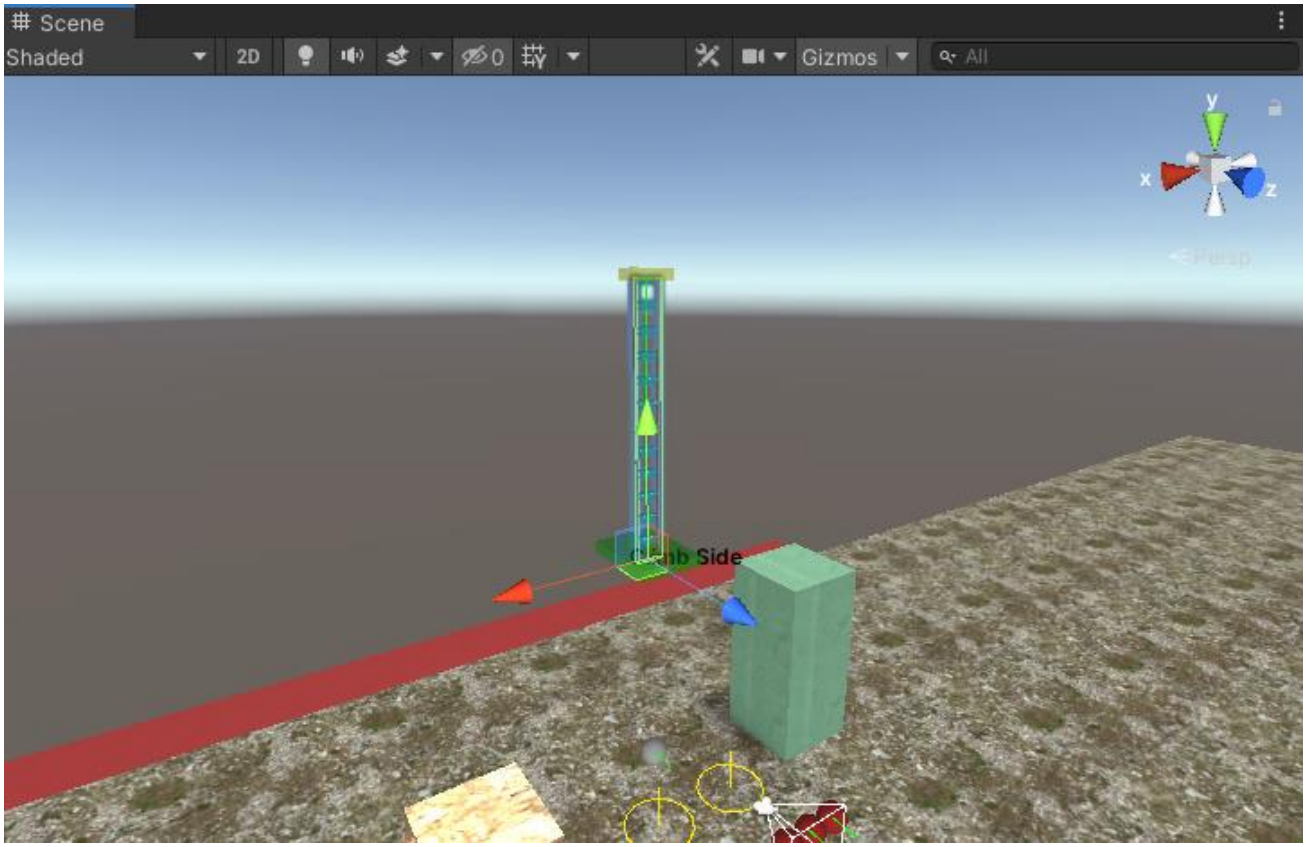
- Create '_AllPoints' object by clicking Warmerise -> Workflow -> Create '_AllPoints' Object
- Remove unnecessary object from '_AllPoints' (Armory, JetPoint, CarPoint and SpawnPoints)
- Start placing the objects around the map so it mimics the base layout, then select '_BallModeBaseMap' and disable its MeshRenderer component, so it becomes invisible.
- The Ball mode map is ready.

Adding Ladders

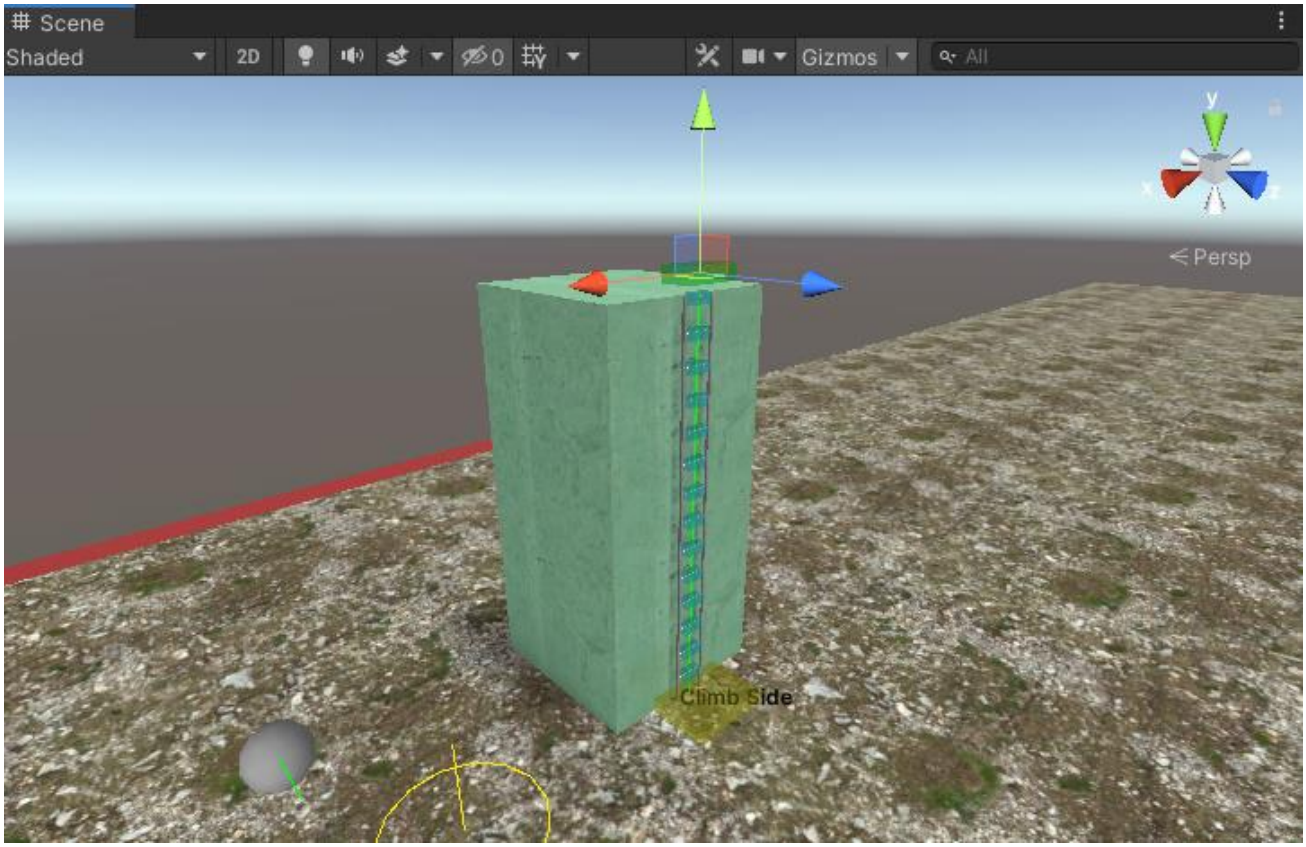
Ladders allow players to climb up and down, and are used when the map has multiple levels, floors, towers, etc.:

- Create new ladder by clicking Warmerise -> Workflow -> Create New Ladder

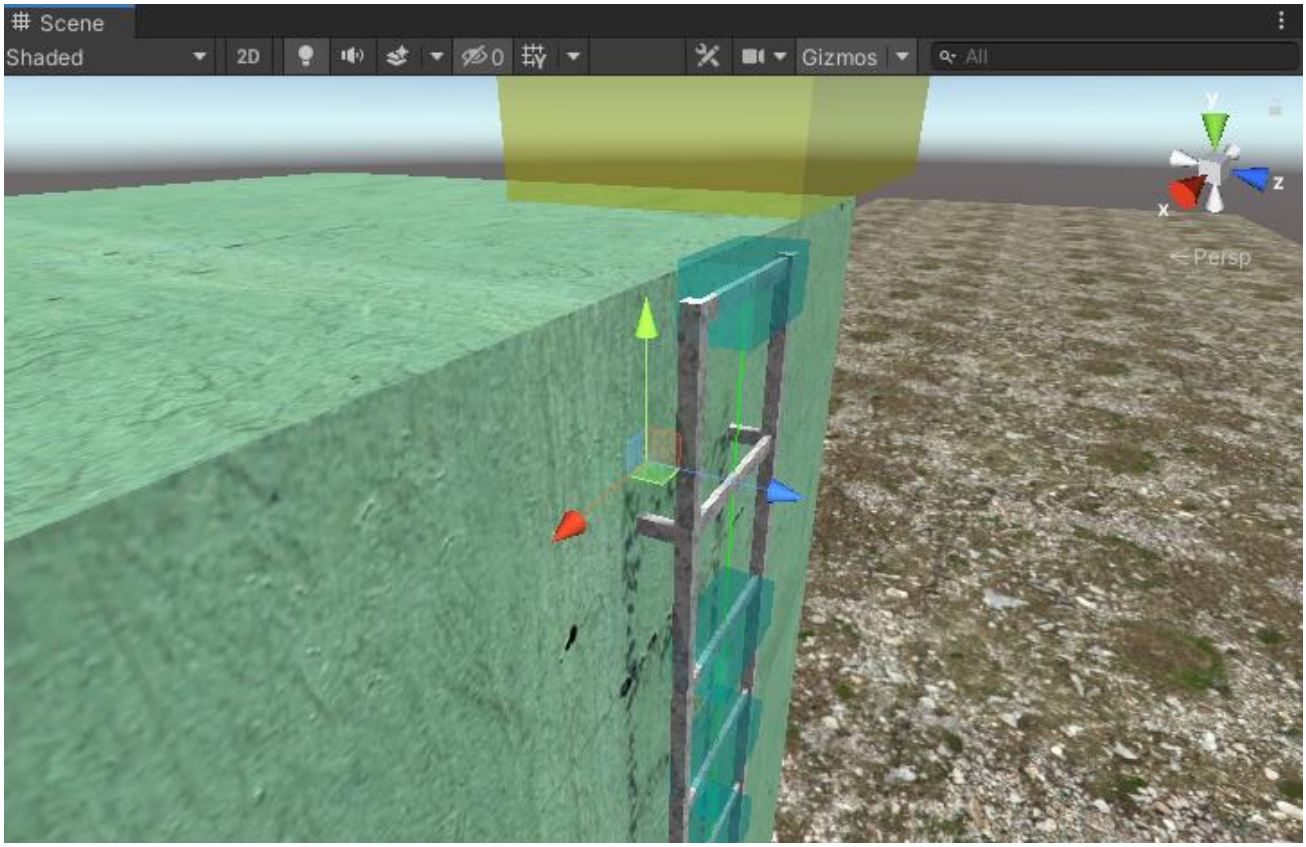




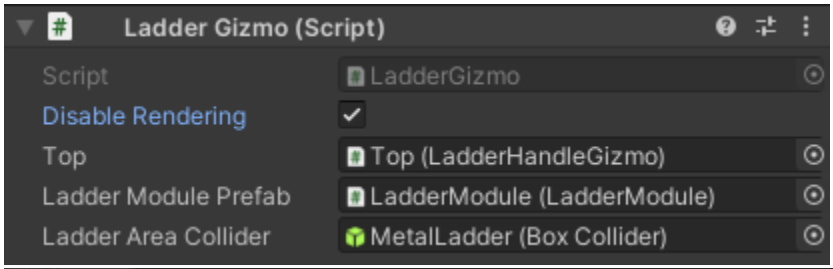
- Move the ladder to where you need it, then select the Top square and move it up or down, to adjust the height (make sure the ‘Climb Side’ is facing away from the wall):

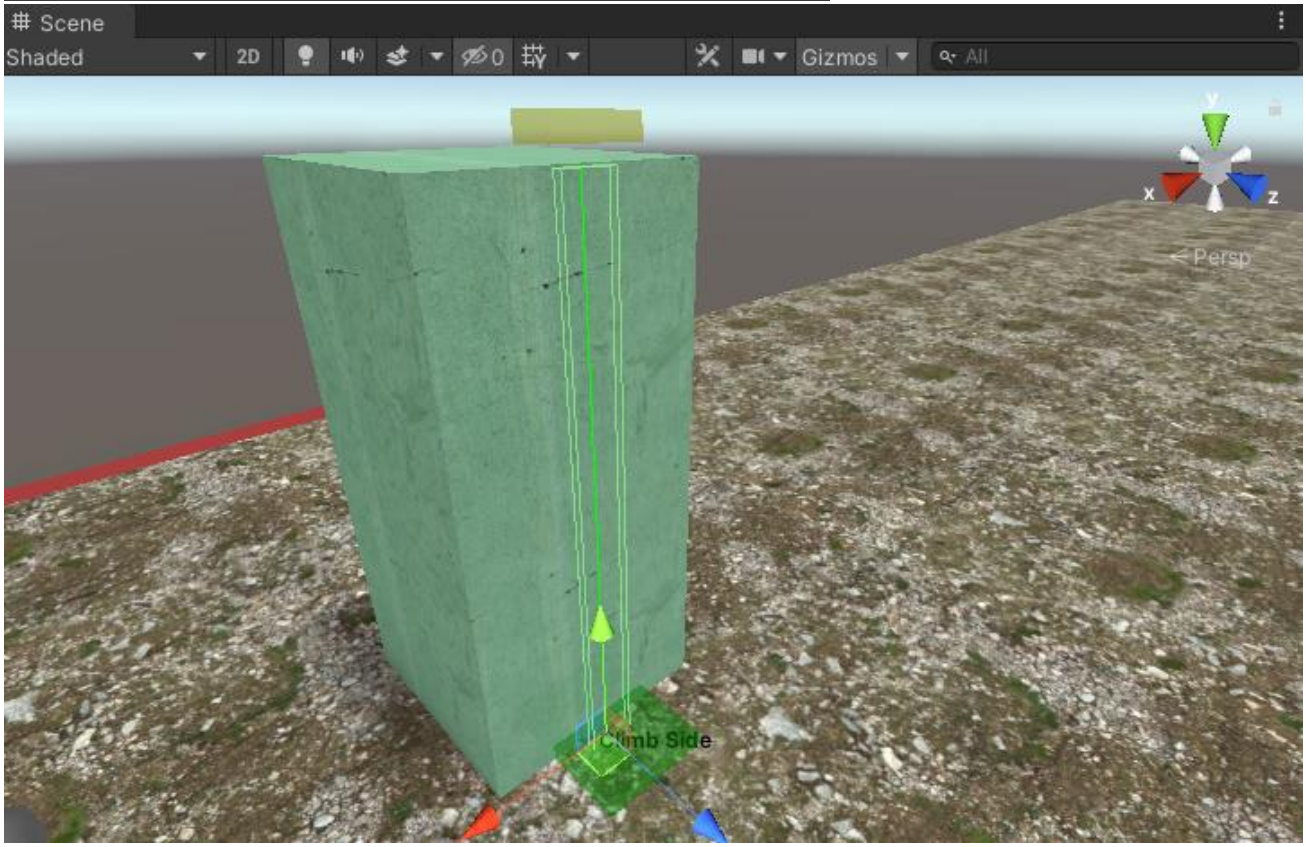


- You can tweak the support rails, by dragging the smaller squares out, towards the wall:



- If you have your own ladder model, then select the bottom square and in Inspector view select ‘Disable Rendering’ option:

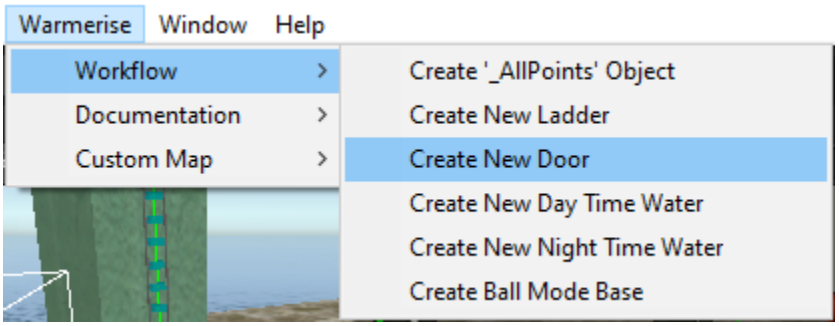


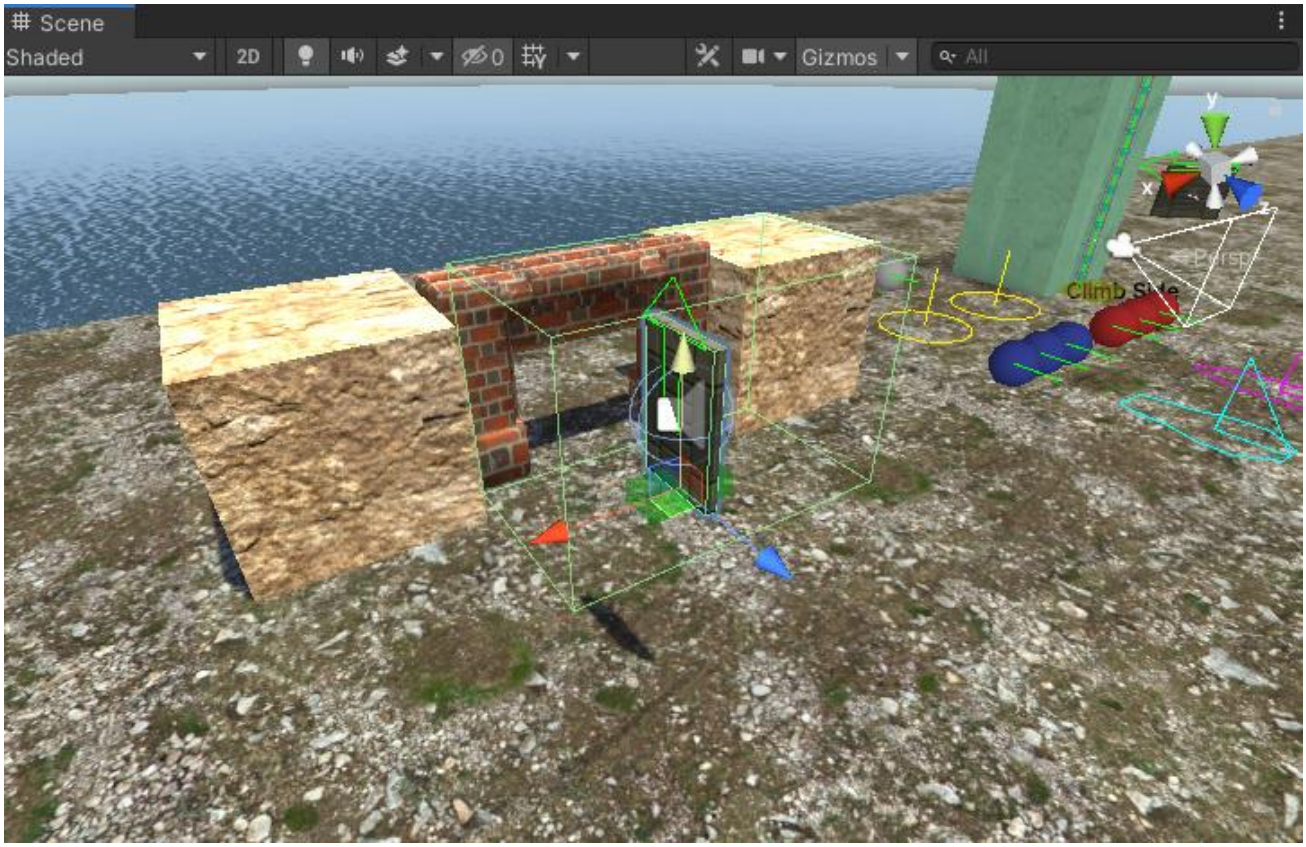


Adding Doors

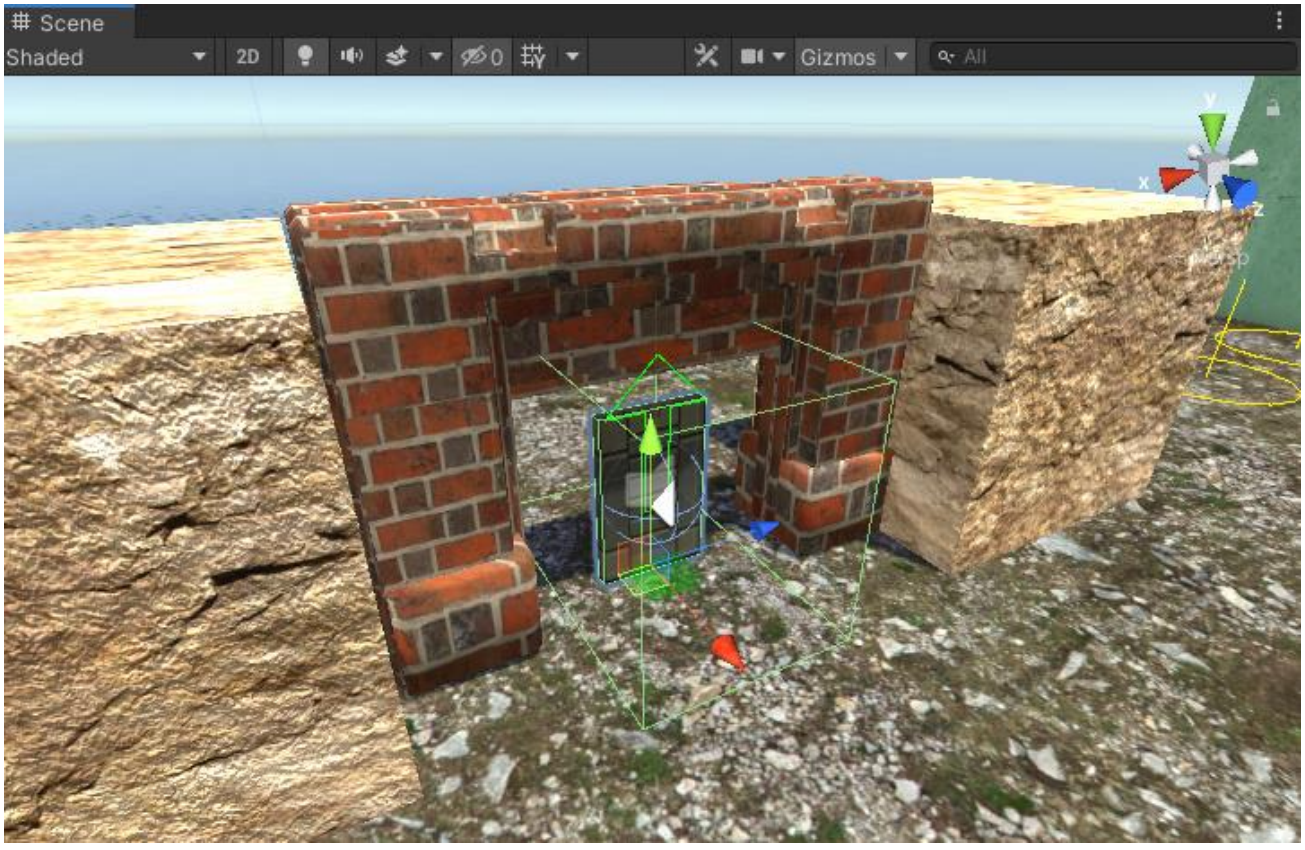
Sliding doors are suitable for maps with rooms or indoor/outdoor spaces:

- Create new door by clicking Warmerise -> Workflow -> Create New Door

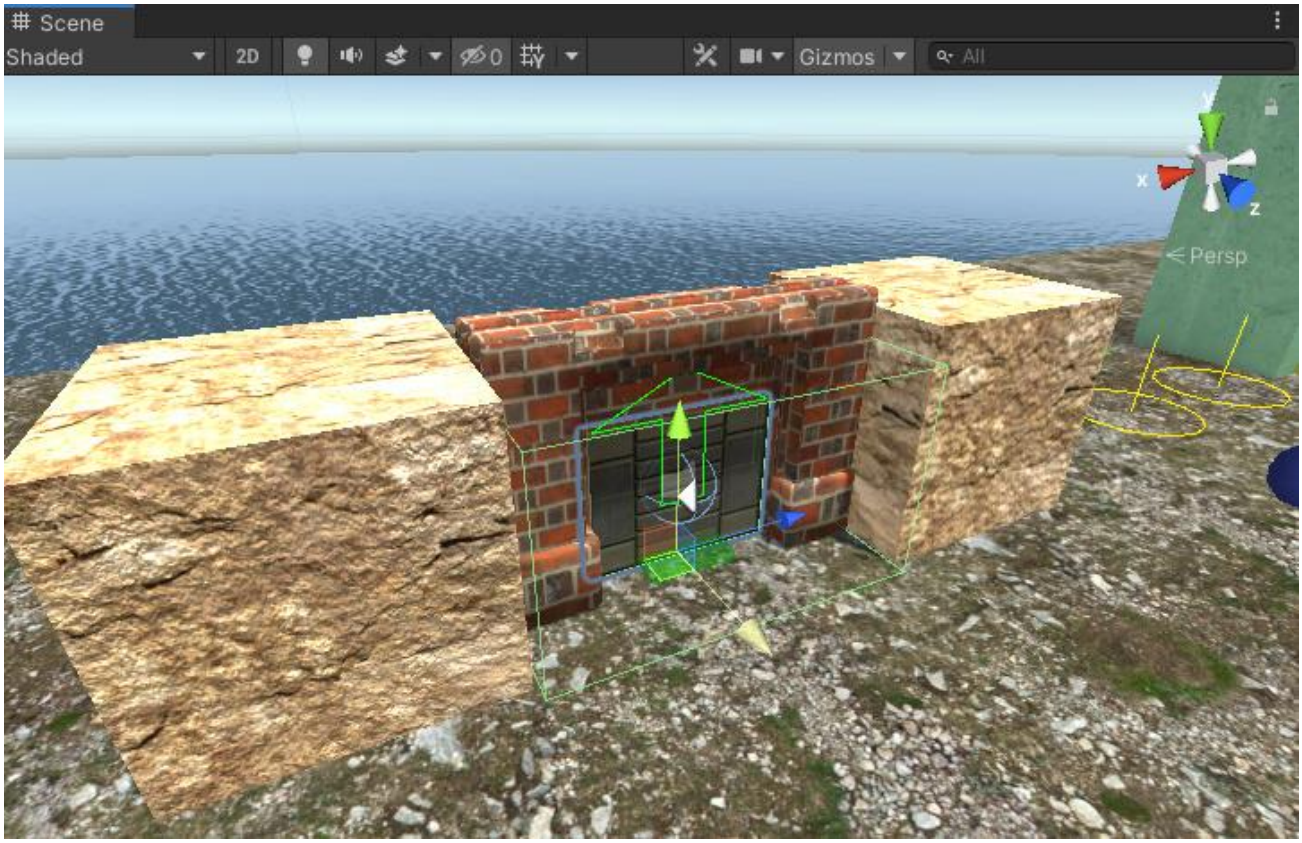




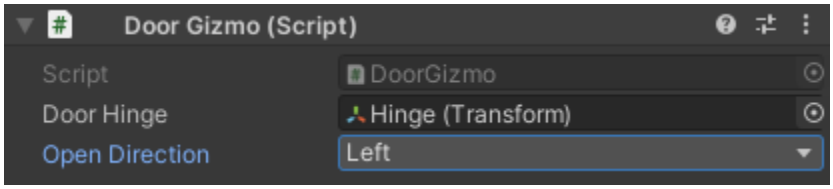
- Move the door in place



- Scale the door until it matches the passage:



- The small green box represents a handle gizmo (used for selecting the door in Editor)
- The big green outline represents the trigger Box Collider area, entering / leaving which, will open / close the door
- The green arrow represents the open direction of the door, to change it, select the door root object and change ‘Open Direction’ value in Door Gizmo:

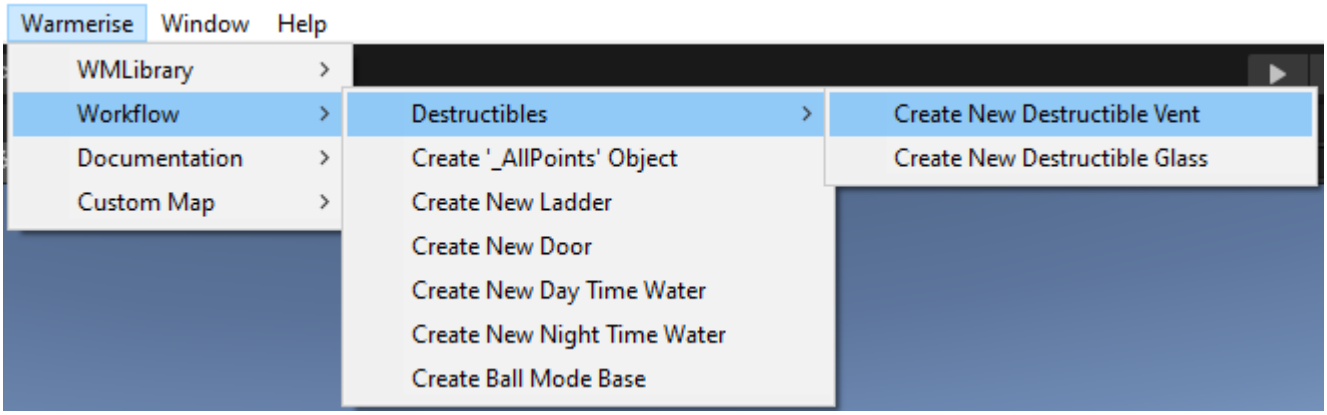


- To change door model, replace the object inside ‘Body’ object (Make sure the new door model has a collider component)
- To change the door open/close sounds assign AudioClip in AudioSources attached to the ‘Open’ and ‘Close’ objects inside the ‘Audios’ object.

Adding Destructible Objects

Destructible objects are the type of objects that can be damaged by the players in the game and is a great way to add gameplay variety to your map.

The export package comes with a few presets for destructible objects that can be created by going to Warmerise -> Workflow -> Destructibles and selecting any of the available presets.



All destructible objects must reside in _AllPoints/Destructibles object and follow a special structure.

To setup a new destructible object from scratch, follow the steps below:

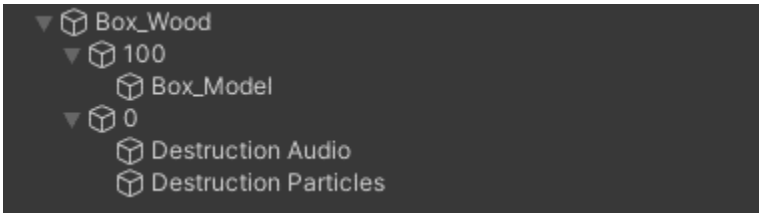
1. Create a new GameObject inside '_AllPoints' and call it ‘Destructibles’ (If it wasn’t created yet)

2. Create a new GameObject inside ‘Destructibles’ and give it any name (NOTE: You can include surface keywords for different hit particles and footstep sounds (Wood, Metal, Glass), for example: Naming it 'Box' will treat it as a default surface, and naming it 'Box_Wood', will treat it as wood surface, ‘Box_Metal’ as metal, ‘Box_Glass’ as glass, etc. For this tutorial, I will use the name ‘Box_Wood’.

3. Create a new GameObject inside ‘Box_Wood’ and name it any number between 0 and 1000 (this is the damage that will be required for destroying this variant). The order of the objects inside ‘Box_Wood’ is important, meaning the first object will be activated at the start, this is a version for the undamaged object, the object after that is damaged, and the last object is fully destroyed. You can have between 2 and infinite stages of destruction.

4. Place the models inside each variant and make sure the models have colliders, otherwise, no damage will be done to them. You can also place empty objects with the Audio Source components per each variant, so the audio is played when the variant is activated, and an object with the Particle System component so the particles are played when the variant is activated. Make sure the position of the model of each variant matches, so the destruction looks seamless.

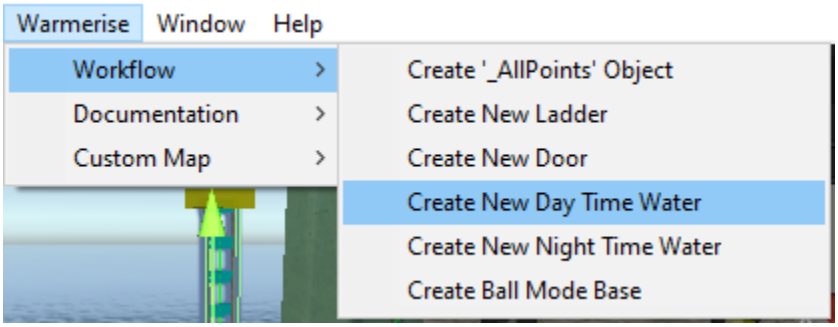
The final object structure:



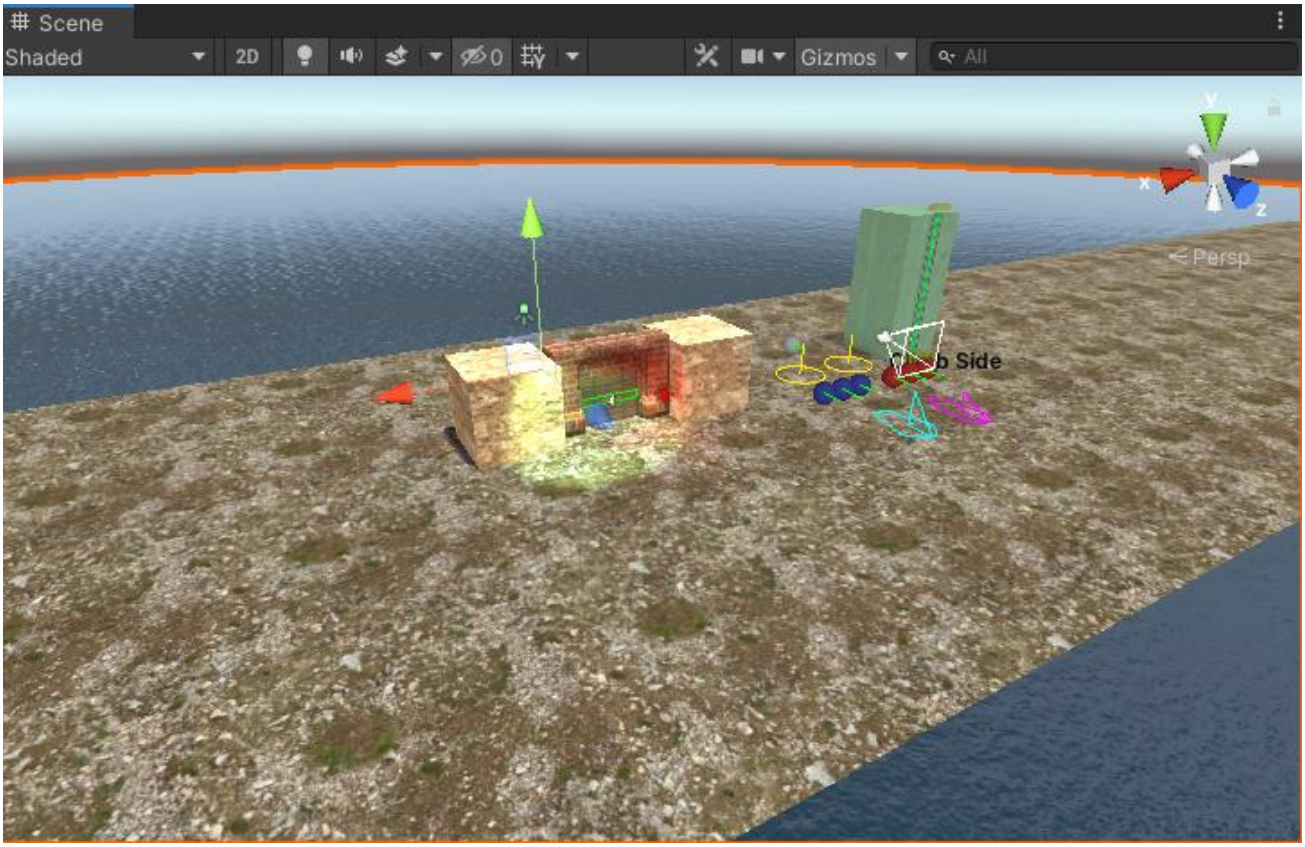
Adding Water Plane

Optionally, you can add a reflective water, suitable for island-type maps, pounds, pools etc.

- To create water click Warmerise -> Workflow -> (Create Day Time Water or Create Night Time Water)



Day Time Water is suitable for maps with "daytime" lighting and Night Time Water is suitable for dark "night time" lighting.



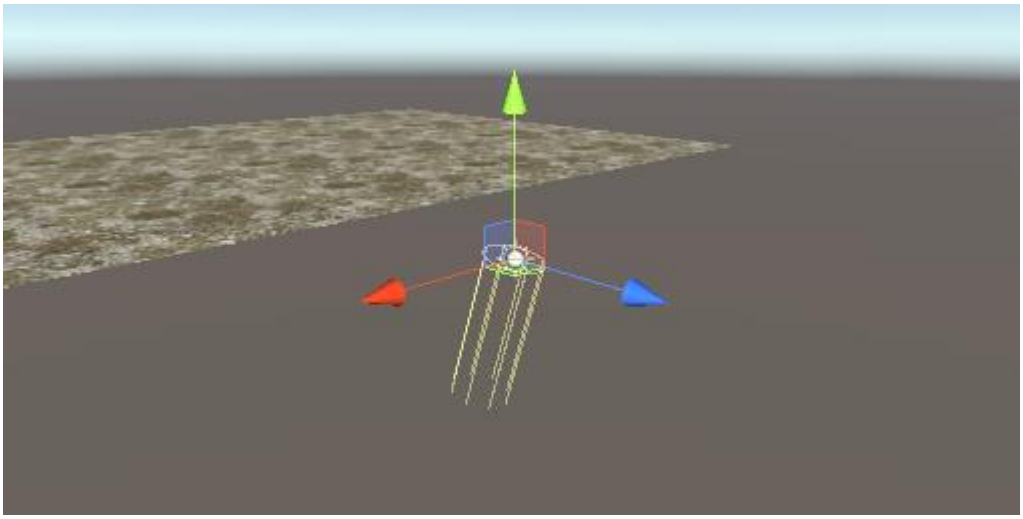
Map Lighting

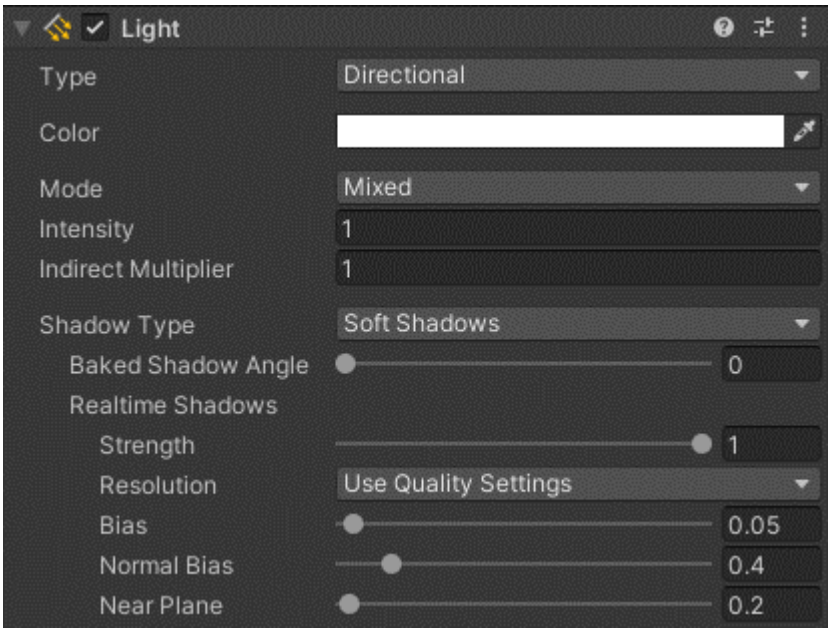
Lighting plays a big role, when it comes to making an enjoyable map.

In Unity the lighting comes from multiple sources.

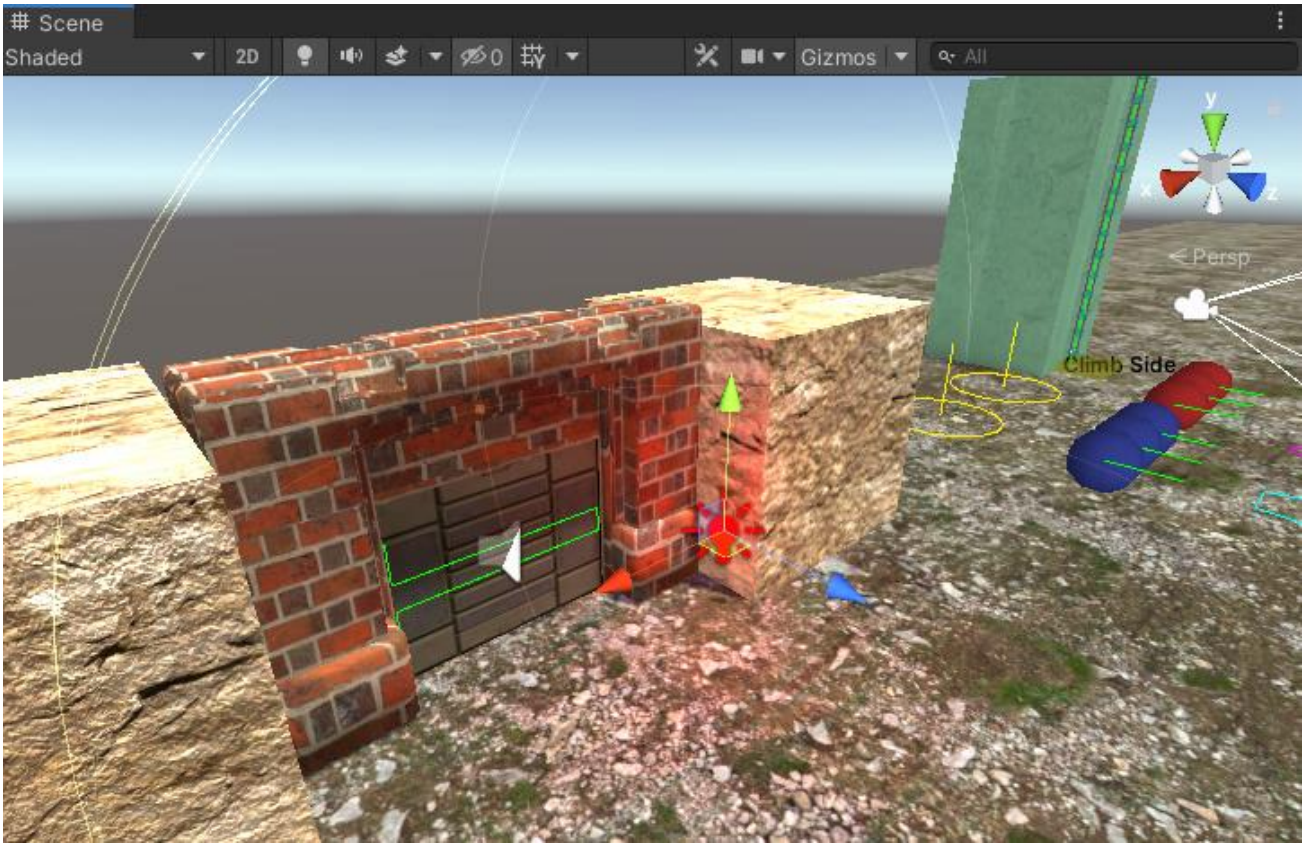
Scene Real-Time Lighting

Directional Light – when you first create a Scene, there Directional Light is added right way. You can select it and tweak the values such as Color, Intensity, and Shadow Type, and it's rotation as well. This type of light projects on every object in the Scene and should be the main light source.

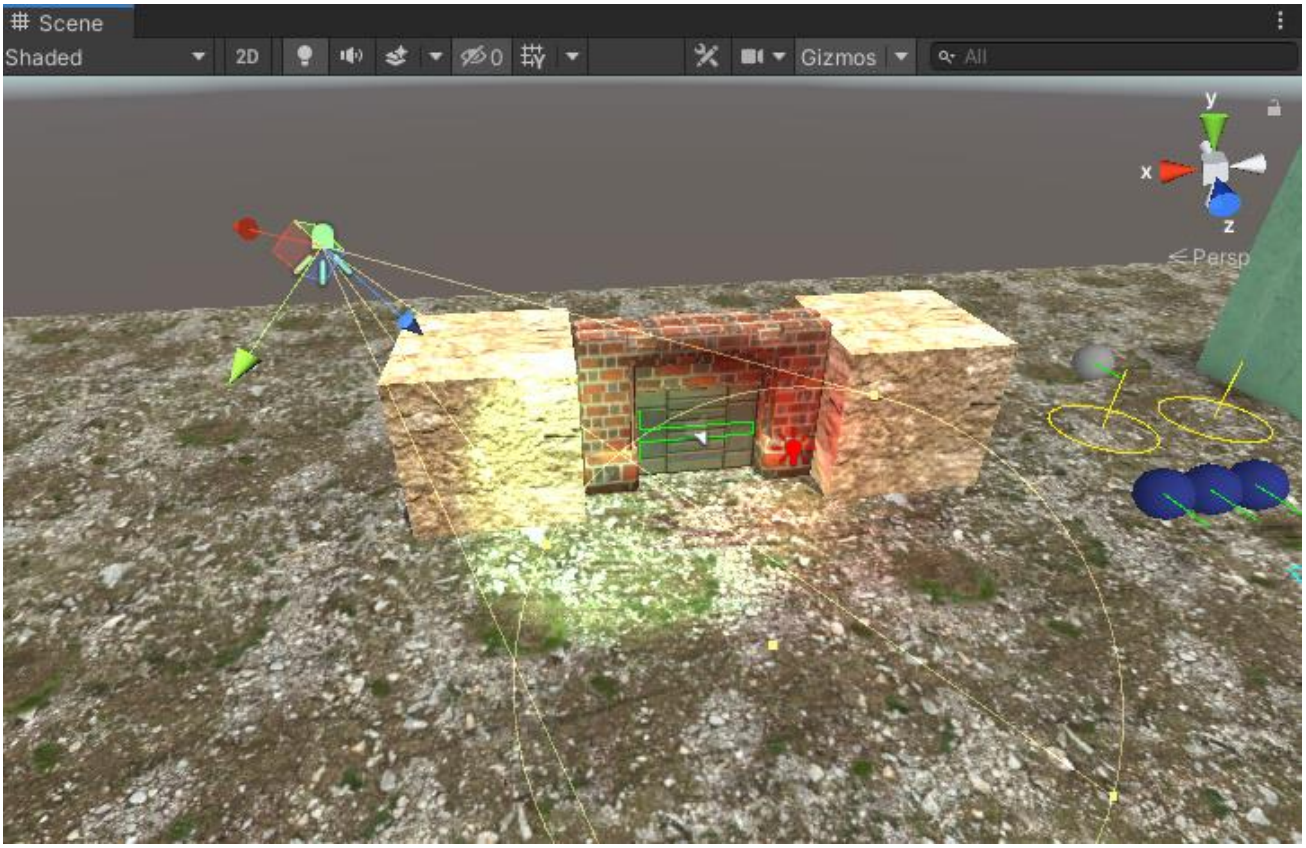




Point Light – (Game Object -> Light -> Point Light) this light illumines the area close-by (Tip: avoid having too many Point Lights, as it may negatively impact performance, unless you plan to bake the Lightmap, in which case the Lights will be translated into texture which will not affect performance).

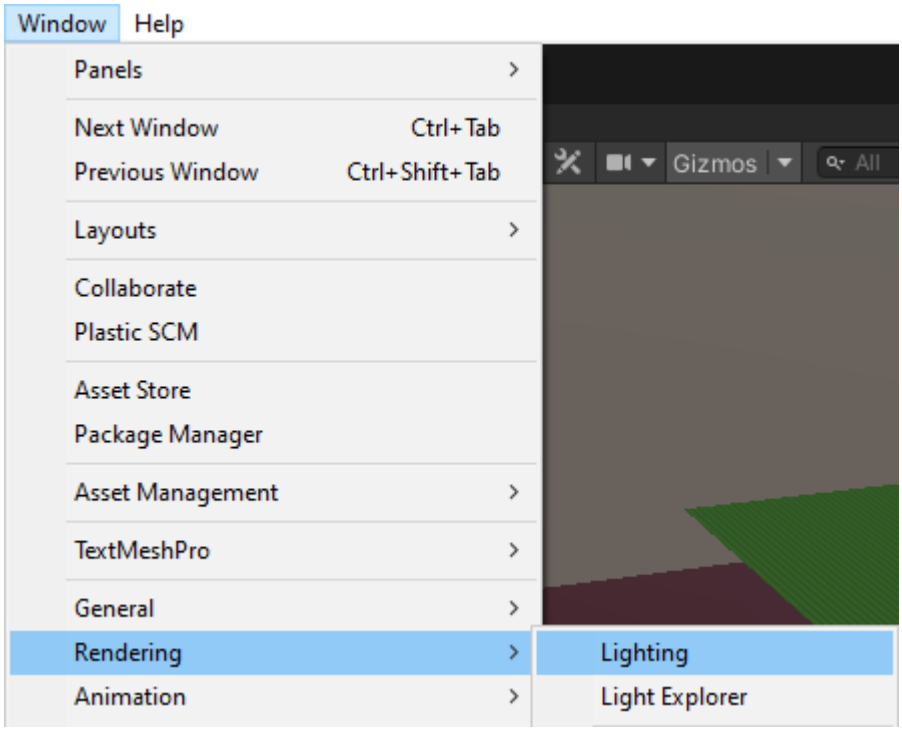


Spotlight – (Game Object -> Light -> Spotlight) just like a Point Light, it illuminates the near-by area, but only in one direction (same as Point Light, avoid having too many Spotlights, unless you plan to bake the Lightmap).

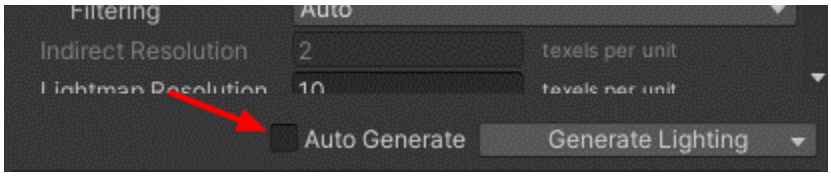


Environment Lighting

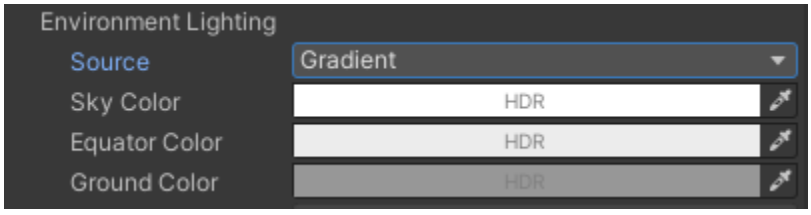
- To tweak the environment lighting, you need to open the lighting window by clicking Window -> Rendering -> Lighting



TIP: To prevent inconsistent lighting between Unity and the game, disable Auto Generate lighting then reload the Scene.



- Under 'Scenes' tab, the default Environment Lighting Source is set to Skybox, which is Ok, but sometimes can be a bit dark. To improve the lighting, change the Source to 'Gradient' and tweak the colors, until you happy with the result:

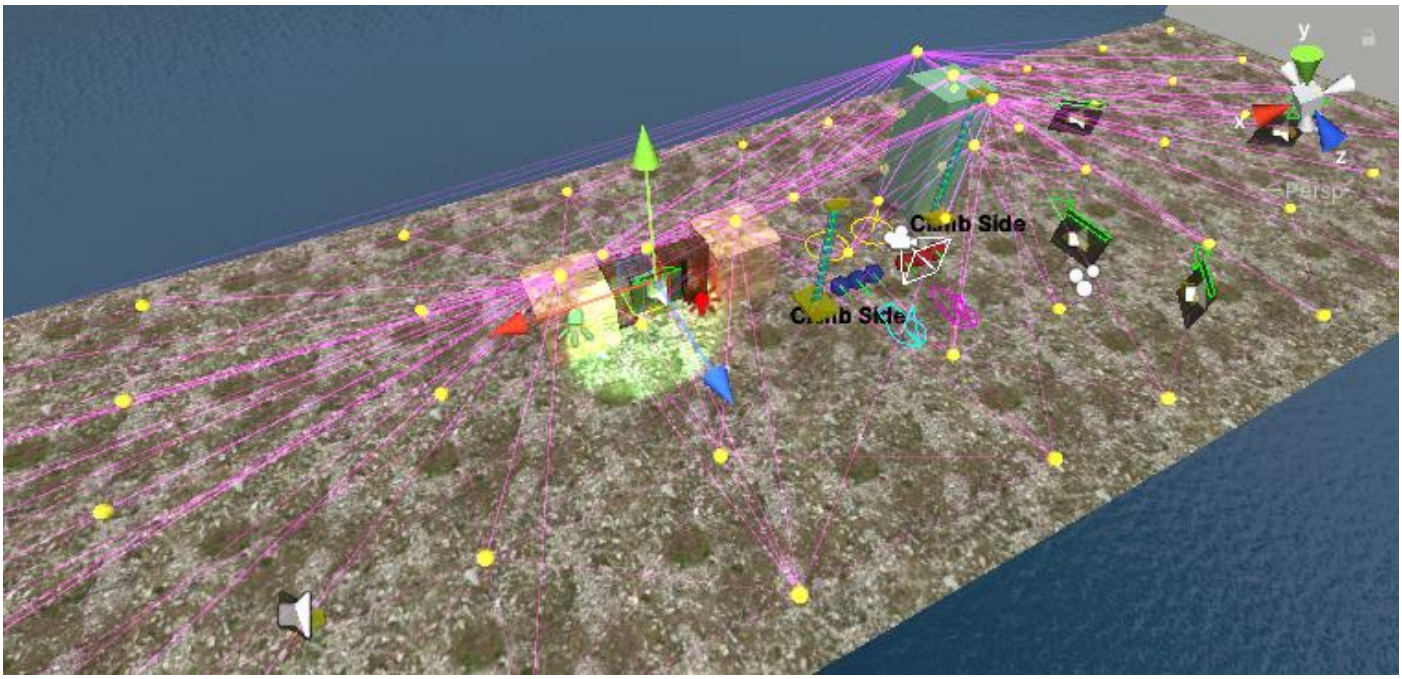


Light Probes

Light Probes provide a way to capture and use information about the light that is passing through the empty space in your scene.

Similar to light maps, light probes store “baked” information about lighting in your scene. The difference is that while light maps store lighting information about light hitting the surfaces in your scene, light probes store information about light passing through empty space in your scene.

Read more here: https://docs.unity3d.com/Manual/LightProbes.html



Video demonstration on how to place Light Probes in your map: https://youtu.be/1cg5UdHyia0

Light maps

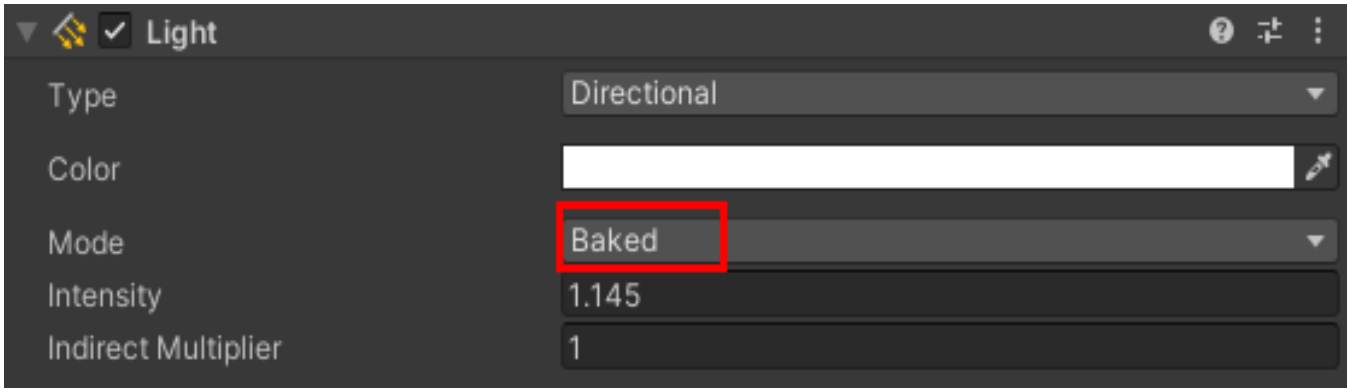
Light mapping is a way to store light data into a texture.

When making a map, sometimes there may be a need to add additional light sources. However, adding too many lights can cause flickering due to the optimization process, which limits the number of real-time lights that can render at once.

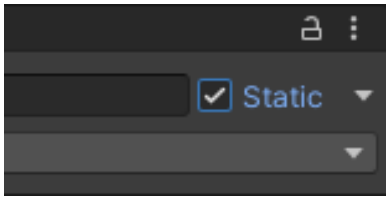
One of the ways to fix the light flickering is by reducing the number of the lights, another, is by baking the light data into a light map.

Below are the steps on how to bake the light maps in your map in Unity:

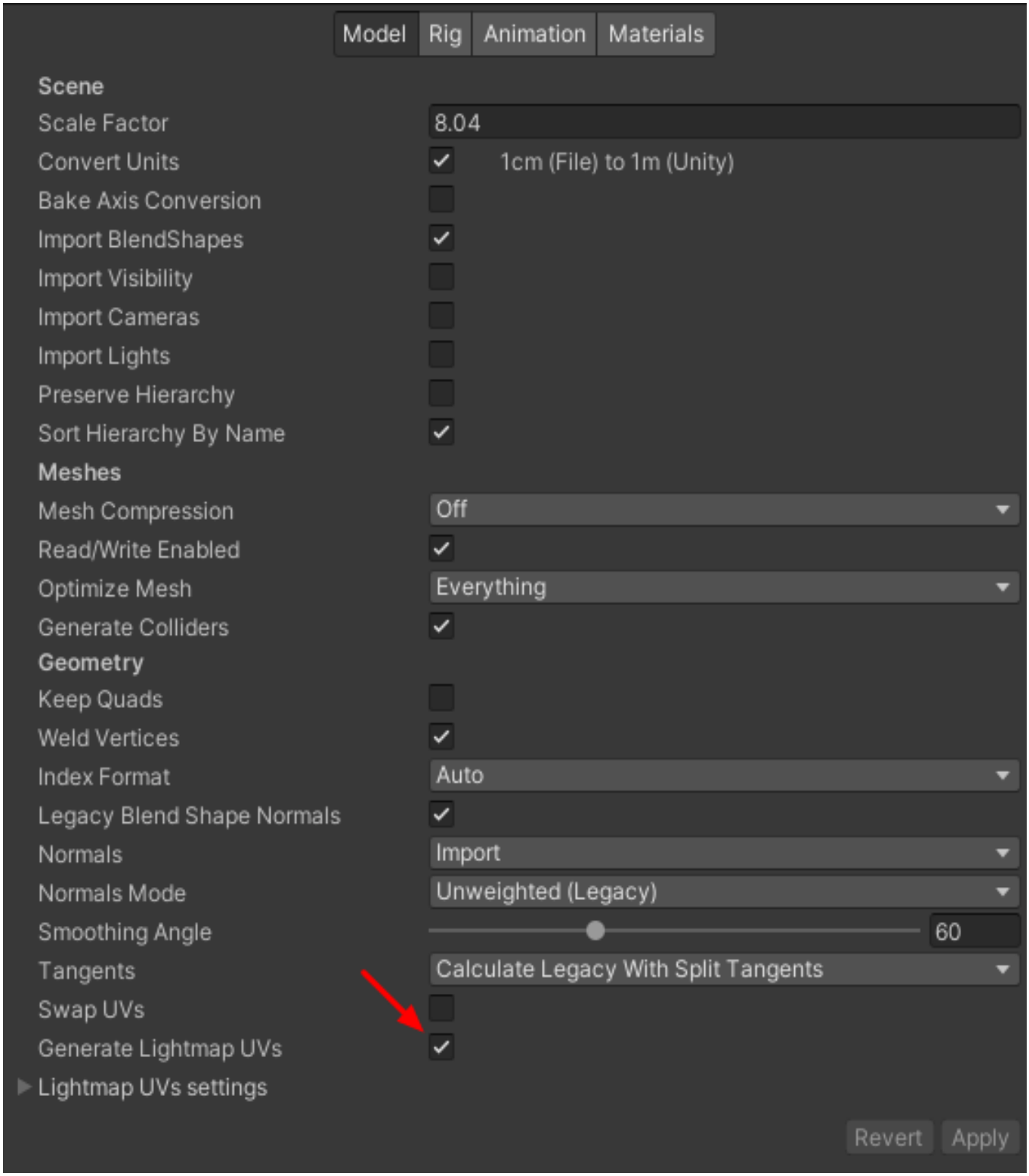
1. Select every active Light in the Scene and in the Inspector view set the "Mode" to "Baked".



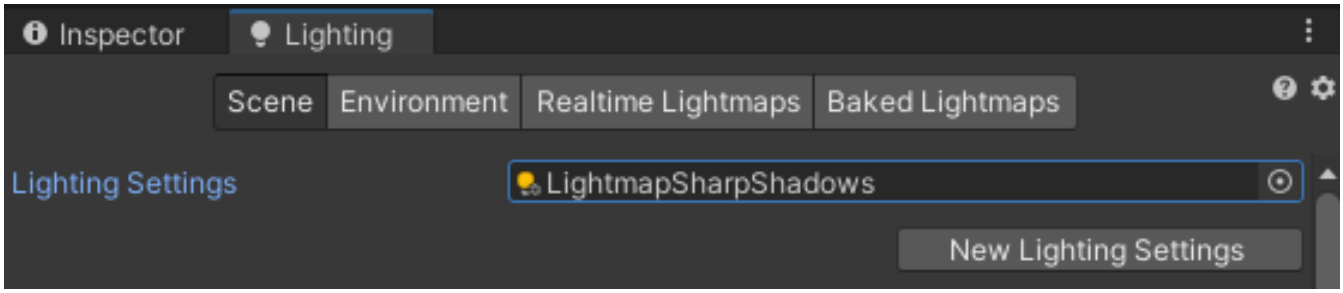
2. Select every map object that you do not plan to animate and in the top right corner of the Inspector view check the "Static" checkbox (Note: If you use the Water circle in your map, do not mark it as Static, since it covers a large area and it would take a long time to process the light maps.).



3. Make sure that the 3D models you're using have Light map UVs generated, for this, select the model in the project view and in the Inspector view under the Model tab check the "Generate Light map UVs" checkbox, and click Apply.



4. Go to Window -> Rendering -> Lighting and in the Window click "New Lighting Settings" or assign the existing Lighting settings preset which are located in the LightmapSettings folder inside the Packages/WarmeriseMapExport.

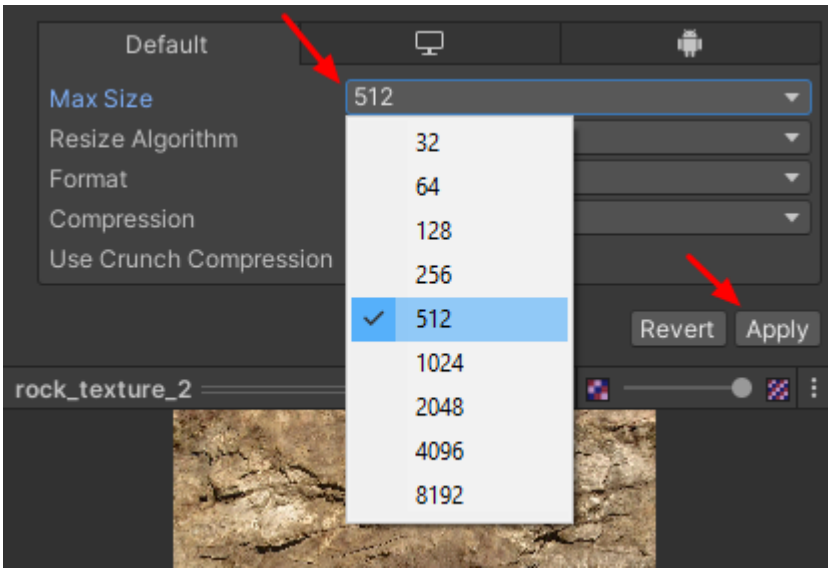


5. Finally, in the Lighting window click "Generate Lighting" and wait for it to finish (The progress will appear in the bottom right corner), after it finishes and you're satisfied with the result, save the Scene.

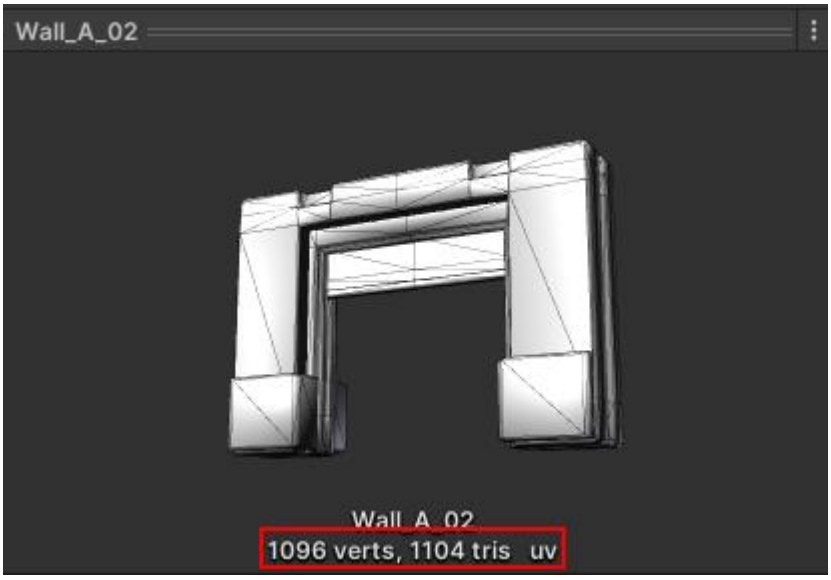
Optimization Tips

Before exporting the map, you need to ensure it's well optimized:

Tip 1: Reduce Texture size to 512 x 512 (or 1024 x 1024, but only for textures which you need to be more detailed)



Tip 2: Only use "low-poly" models (Aim to use models under 10K verts/tris)

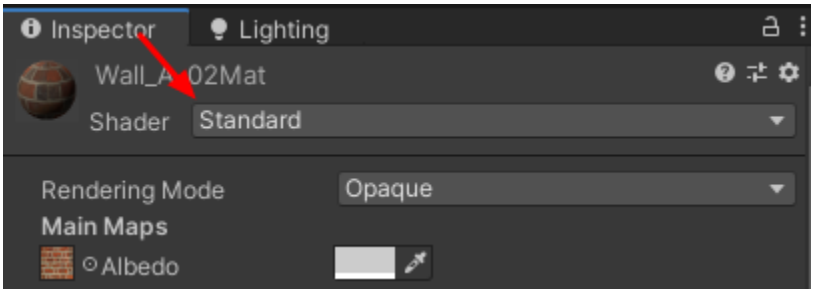


Tip 3: Do not use Normal Map or Bump Map textures in the materials if the Lightmap was generated, since there will not be any real-time lighting, these types of textures will have no effect and just take up space.

Tip 4: If you use unsupported Shaders that have many assigned textures, some of the additional textures will be included in the map file, however in the game, such Shaders will be converted to Legacy Shaders/Diffuse and only the main texture will be used. To prevent that, use the Shaders from the supported Shaders list, this will ensure that the file size is as small as possible.

Supported Shaders

A Shader is a component that is applied to the material and changes the way it looks. Not all Shaders are supported in the game.



Below is the list of supported Shaders:

- Legacy Shaders/Diffuse
- Legacy Shaders/Specular
- Legacy Shaders/Bumped Specular
- Legacy Shaders/Bumped Diffuse
- Legacy Shaders/Parallax Specular
- Legacy Shaders/Parallax Diffuse
- Legacy Shaders/Transparent/Diffuse
- Legacy Shaders/Transparent/Bumped Specular
- Legacy Shaders/Transparent/Cutout/Diffuse

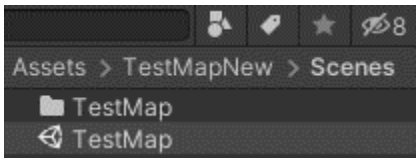
- Legacy Shaders/Particles/Additive
- Legacy Shaders/Particles/Additive (Soft)
- Legacy Shaders/Particles/Alpha Blended
- Legacy Shaders/Particles/Multiply
- Legacy Shaders/Self-Illumin/Diffuse
- Mobile/Particles/Additive
- Mobile/Particles/Multiply
- Nature/Terrain/Diffuse
- FX/Water
- Skybox/6 Sided
- Skybox/Procedural
- Nature/Tree Creator Bark
- Nature/Tree Creator Leaves
- Nature/Tree Creator Leaves Double Sided *(Custom Shader included in the export package)*

Any unsupported Shaders will be converted to 'Legacy Shaders/Diffuse' in the game.
NOTE: As mentioned previously, some unused textures in the unsupported Shaders will be included in the exported file, which will take up space but will not be used in the game. To reduce the file size make sure to use supported Shaders only.

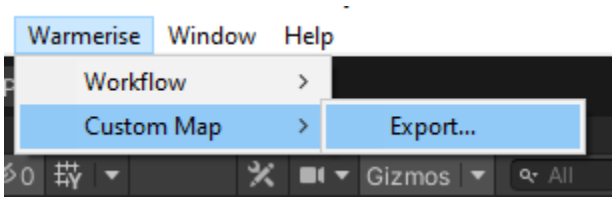
Exporting The Map

After the map was configured, it's ready to be exported:

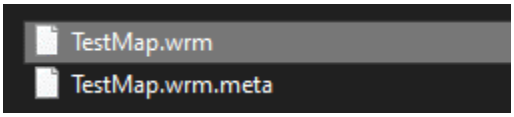
- Select Scene in the Project view
- Rename the Scene to whatever name you want your map to have (NOTE: Once the map is uploaded and published, its name cannot be changed)



- Click Warmerise -> Custom Map -> Export… (Exporting should take a few seconds)



After the exporting is done, the new folder will be created in Assets folder called '__Maps' where the exported file with extension .wrm will be saved, this is the file that can be used in the game.



Testing The Map

Now it's time to test the map, to verify that it works:

- Go to the game page (Make sure you are logged in)
- Click PROFILE -> CUSTOM MAPS -> Test New Map…

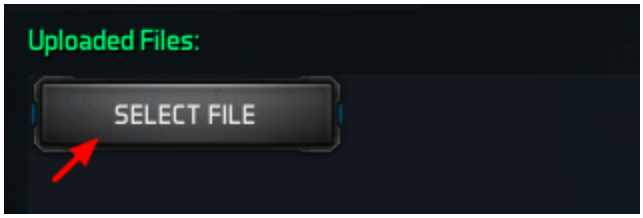
- Select the exported .wrm file.
- Walk around the map to make sure everything works correctly:

Uploading The Map

After the map was tested it can now be uploaded:

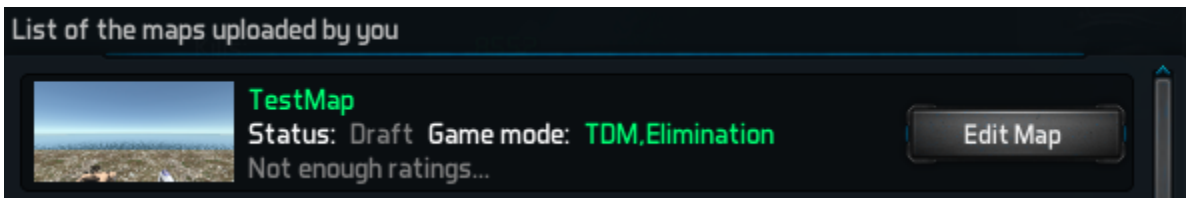
- Click 'Add New Map'

- Click 'SELECT FILE' and select the exported .wrm file



- After selecting the file, the game will quickly load the map and generate the preview image:
- Enter map description, click ‘SUBMIT’ then wait for it to upload.

- Once the upload is completed, the map will appear in the map list



Publishing The Map

After the map was uploaded it can be then published so other players can try it:

- Click ‘Edit Map’
- Near the map file click ‘Options’

Delete is self-explanatory, clicking it will delete the map file

Publish will publish this map file to a pending section, where it can be reviewed and voted by the players.

Changes will open the Changelog window where you can type what's new in this version.

If the map was just created, the appropriate changelog would be "- Initial release", otherwise, you should include the list of the changes, for example:

- Removed object A
- Added new corridor

  Etc.

- To publish the map, click Publish
- To unpublish the map, click Unpublish

Published maps will appear in 'Published Maps' (once approved) and 'Pending Maps' (if under review) lists, when creating a server.

Updating The Map

- To update the map description, go to your maps, click 'Edit Map', make changes to the description, then click 'SUBMIT'
- To upload the updated version of the map, click 'Edit Map' then click 'Select File' then click 'SUBMIT'

- After the file is uploaded, click ‘Options’ then click ‘Changes’, and type what’s new in this version
- Lastly, click ‘Options’ then ‘Publish’ (NOTE: If there is already a published file waiting for approval, you need to unpublish it first).