DunGen documentation

# Instructions:

Take the Grid prefab in DunGen>Prefabs and drag it into the scene, it is already made with default assets added in to be replaced at users leisure

On runtime this will generate a random linear dungeon using pre provided assets

On the grid object there is two key components, the Grid Generator and the Room Generator.

On the Grid Generator you have 3 editable properties:

* Vector2Int extents:

the size of the grid you would like to have

* Int cellSize:

the size of each individual square of the grid, is technichally a setter for float privateCellSize as some operations under the hood require floats to be used for cell values

* Bool drawOnlyExtents:

enabling this will disable drawing the gridline gizmos and only show the bounds of the grid

On the room Generator there are several editable properties:

* Bool mannualySelectNodes:

when true will expose the required property which is an array of serializable structs, that allow for the manual placement of required nodes given their roomType, gridPosition, and rotation

* Int extraNodes:

assigns how many extra random nodes the path must travel to before it reaches the end

* GameObject startRoomFab:

the prefab of the room at the start of the path

* GameObject endRoomFab:

the prefab of the room at the end of the path

* GameObject[] corridorRooms:

the prefabs of possible corner rooms that can spawn

* Bool generateOnRuntime:

checking this box will generate the grid on runtime and disable the generate new dungeon button

The roomGenerator exposes a startRoomPos property which allows access to the start of the path for the purpose of spawning of characters

# Creating a new room type

In this example we will be creating a room from scratch, using probuilder to help speed the process along, and is written assuming the user has experience using probuilder.

**Step 1)**

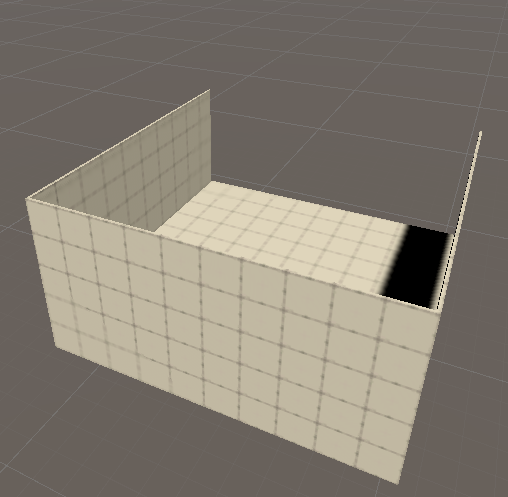
first we will create a cube in probuilder with X and Z dimensions in accordance to your grids cell size, in this example we will be using the default of 10 and make sure your rooms pivot is in the center of the square

Graphical user interface

Description automatically generated

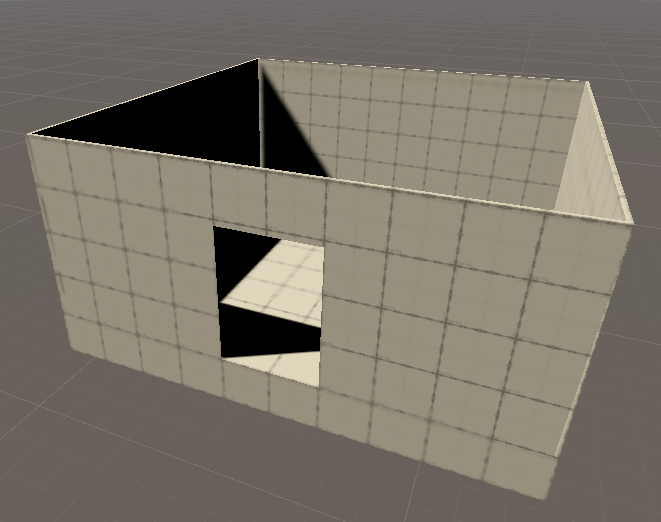
**Step 2)**

Secondly we will give the room it’s primary shape, this room will be a room with only one door for specific use as a start room or an end room so well will give the room its solid walls first

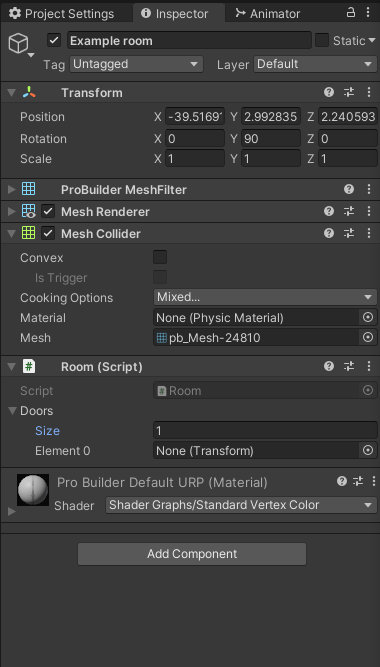


**Step 3)**

Now we make the door, which will be variably shaped depending on how your character is made, be sure to consider such when making your door



**Step 4)**

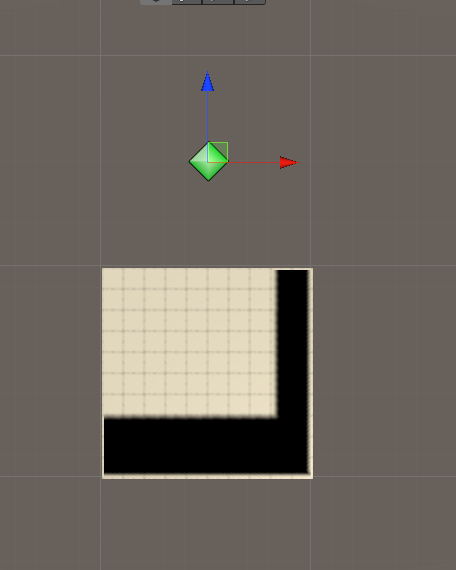
Next we add a room component to the parent game object and give it the amount of door anchors it needs to use, *please note that in the current version of this tool only linear rooms with up to 2(two) doors are supported*.

**Step 5)**

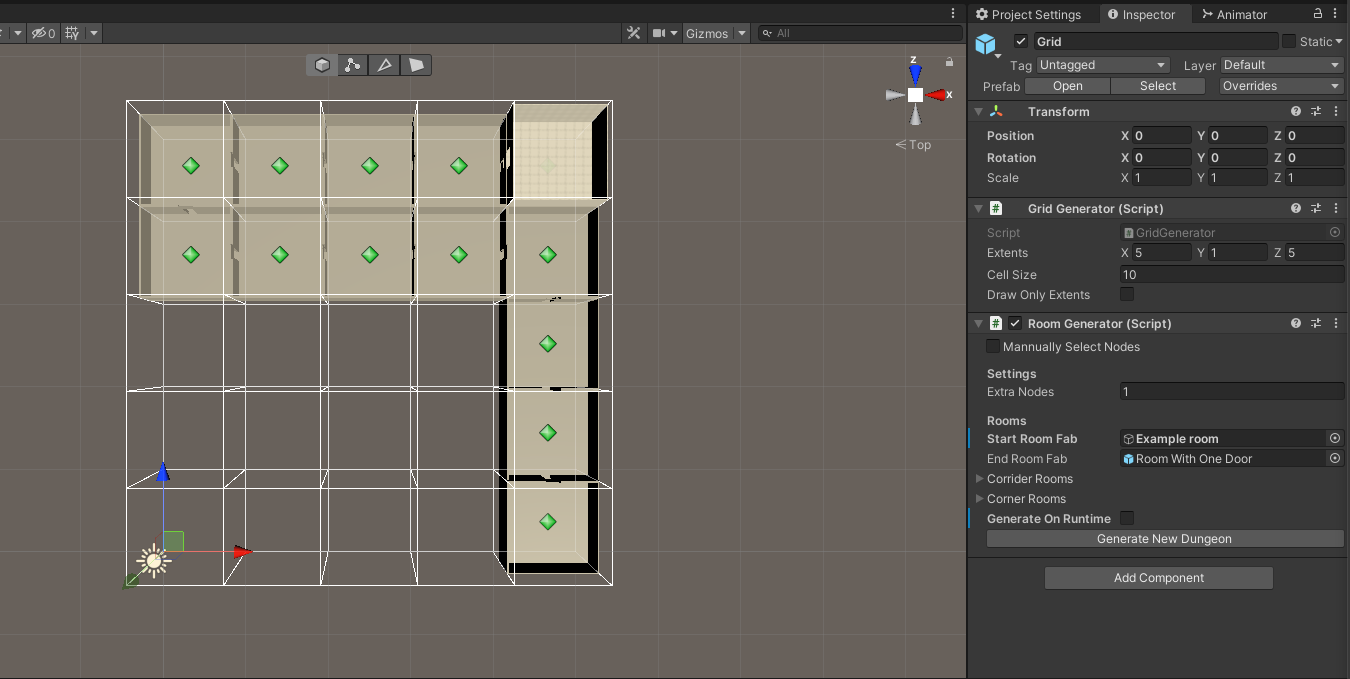
After that you add the door anchors themselves, these are just empty parented to the room itself as empty game objects.

When placing the pivot be sure to place it where another rooms center would be, so if one has a cell size of 10 and you had a north facing door you would place the door anchor 10 units on the positive z axis from the center of the room.

While it is not a necessity for the anchor to be placed precisely as such (the only requirement that the door anchor reaches a neighboring cell), it is strongly recommended as to assure the requirement is met. Do not forget to add the anchor to the doors array on the Room script.



**Step 6)**

after this the room is ready to be added to the room generator component of the grid, in this case this will be used as the new start room

And just like that a new room

added to the grid generation

system