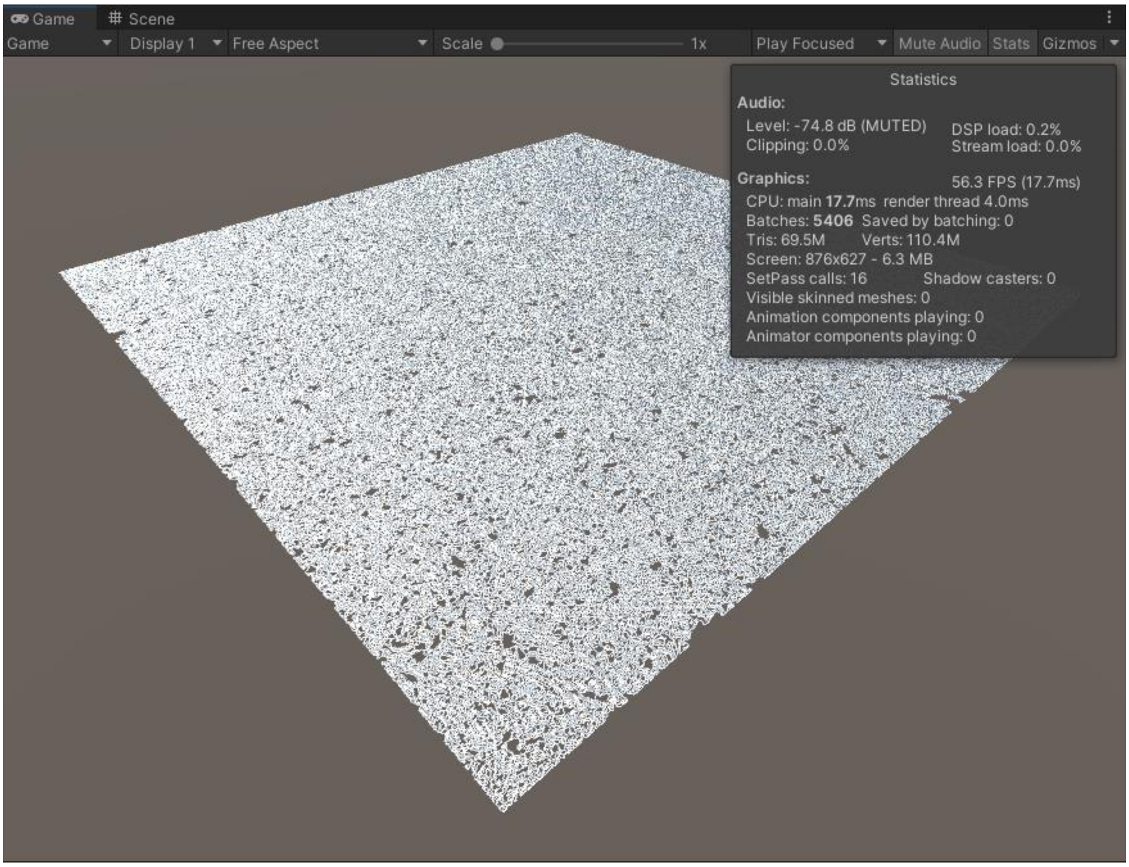
Maze generator guide

More information can be found on https://goldor8.github.io/MazeGenDoc/

1. Introduction

Maze Generator is an asset to make mazes, it uses an agent-based method to generate them, it supports seeds to regenerate exactly same maze with only one integer. It also supports baking which provide a huge performance boost by lowering draw calls count.

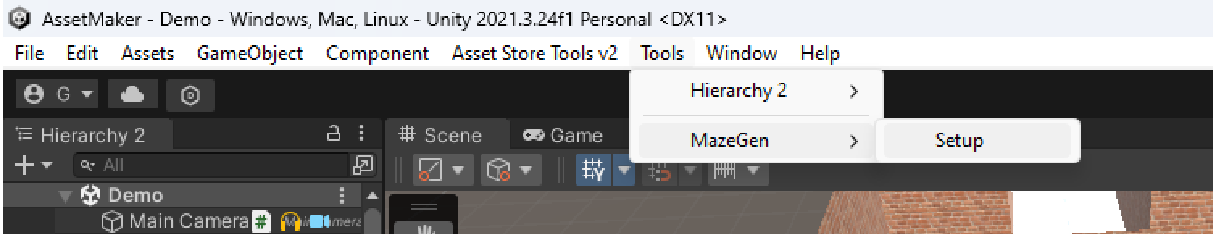


```
Statistics
Audio:
  Level: -74.8 dB (MUTED)      DSP load: 0.2%
  Clipping: 0.0%               Stream load: 0.0%
Graphics:                      56.3 FPS (17.7ms)
  CPU: main 17.7ms  render thread 4.0ms
  Batches: 5406  Saved by batching: 0
  Tris: 69.5M      Verts: 110.4M
  Screen: 876x627 - 6.3 MB
  SetPass calls: 16          Shadow casters: 0
  Visible skinned meshes: 0
  Animation components playing: 0
  Animator components playing: 0
```

It can hold 1500 by 1500 parts maze with low detailed parts at almost 60 fps with the baking. This maze couldn't even load without baking.

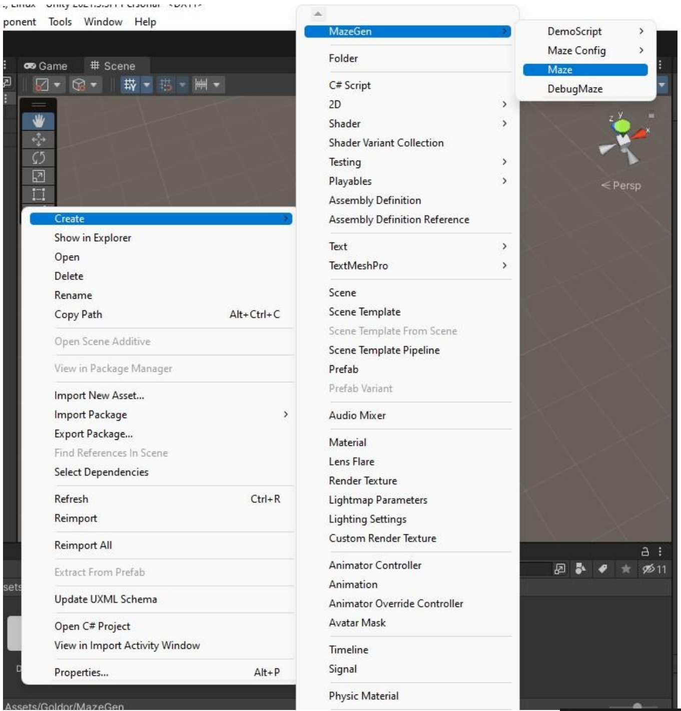
This asset is highly customizable with scripts. Maze processors are an easy way to modify the maze with some specific logic in the demo, trap and treasure are integrated to the maze with this method. You can even make a new maze generator script and keep the maze instantiator to make a completely new type of generation.

A guided setup can be found in the Tool dropdown menu



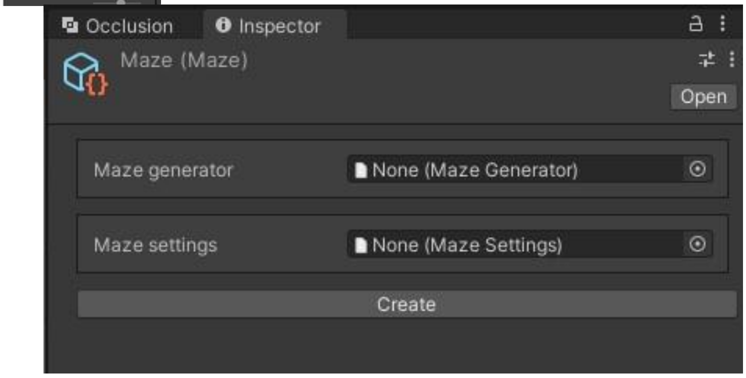
2. Maze representation

Mazes are represented by scriptable objects. These scriptable objects hold the maze data, the generator, and the settings to generate it.



A new maze representation can be created by the menu when right clicking

Here is what the inspector look like

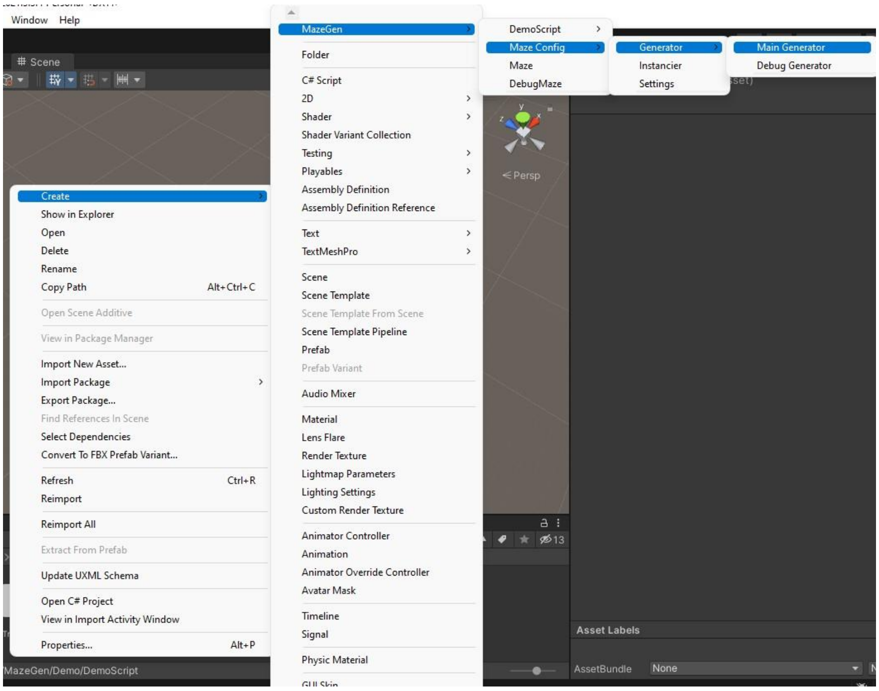


Maze generator: None (Maze Generator)

Maze settings: None (Maze Settings)

2.1. Maze generator

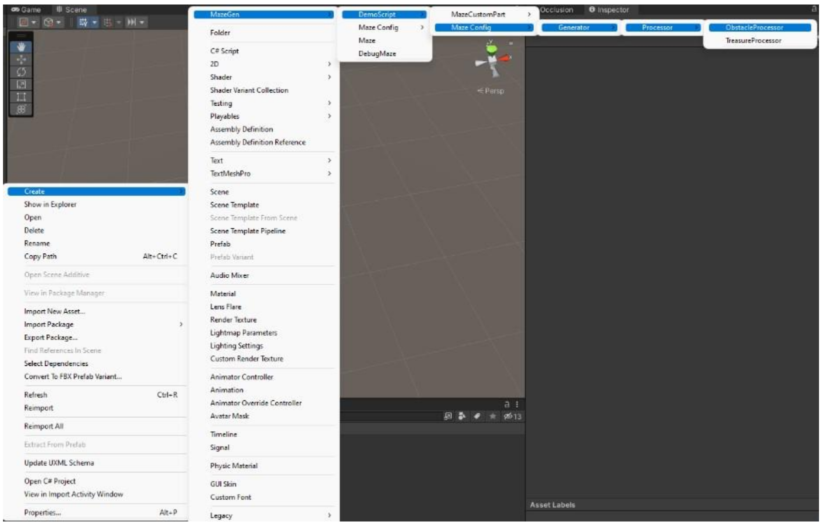
The generator is a scriptable object that manage the creation of the maze new one can be created to change the creation behavior.



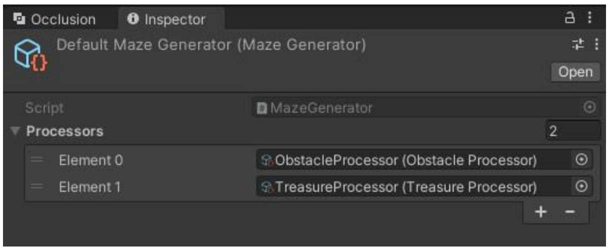
The path to make a new generator scriptable object.

2.1.1. Maze processors

Maze processors are additional script that run over the maze when the first generation is finished to modify it, they can be used for example to replace some part. To use it you need to attach the maze processor scriptable object to the maze generator, and it will run automatically.



You can find some of them in the demo scripts.



Occlusion | Inspector

Default Maze Generator (Maze Generator)

Open

Script: MazeGenerator

Processors: 2

Element 0: ObstacleProcessor (Obstacle Processor)

Element 1: TreasureProcessor (Treasure Processor)

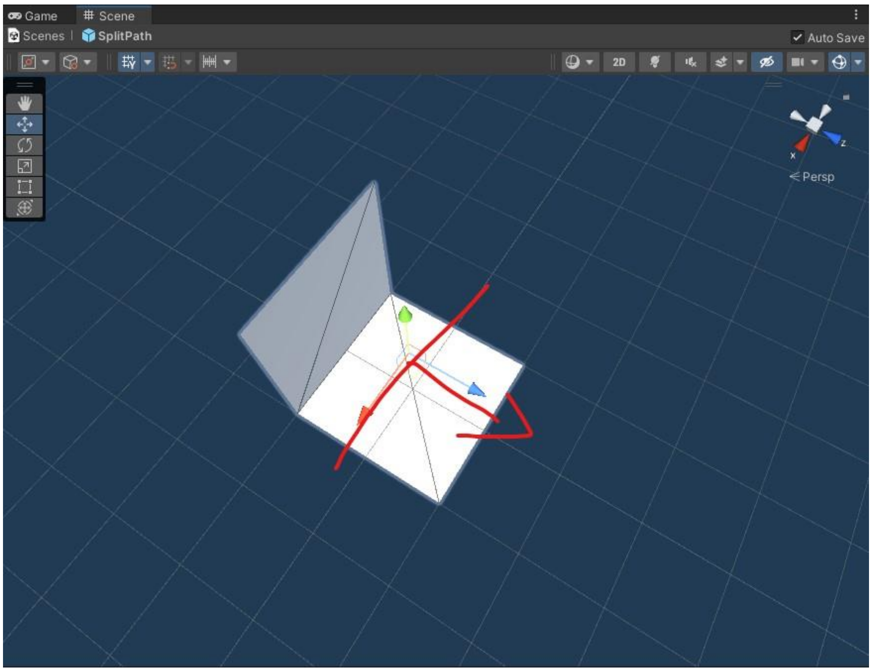
You can add the processors through the maze generator editor.

2.2. Maze Settings

Maze settings is a scriptable object that hold the settings for the generation of your maze. To use settings put the scriptable object in the maze representation.

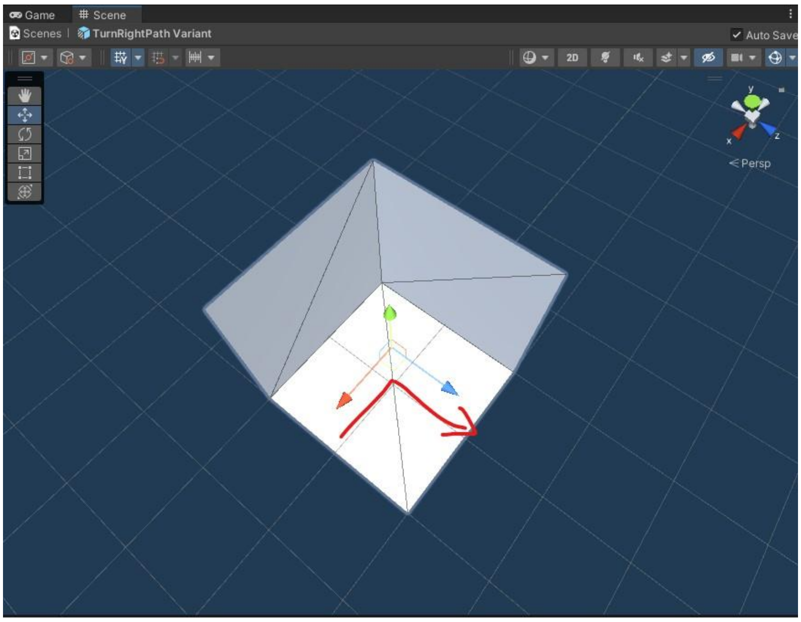
3. Maze instantiator

The maze instantiator determine how you maze will be represented in the scene by storing the different basics parts prefab of the maze. Maze part orientation is important. The Z axis of each prefab must be the exit way of the prefab except for the 3-way and 4-way parts the 4-way part don’t care about the direction but the 3-way one also named “split path” in the maze instantiator must have the Z axis pointing to the new path created (see image below).



Here the split path Z axis is pointing to the new path created

And here for the “turn right” path the Z axis is pointing to the exit way

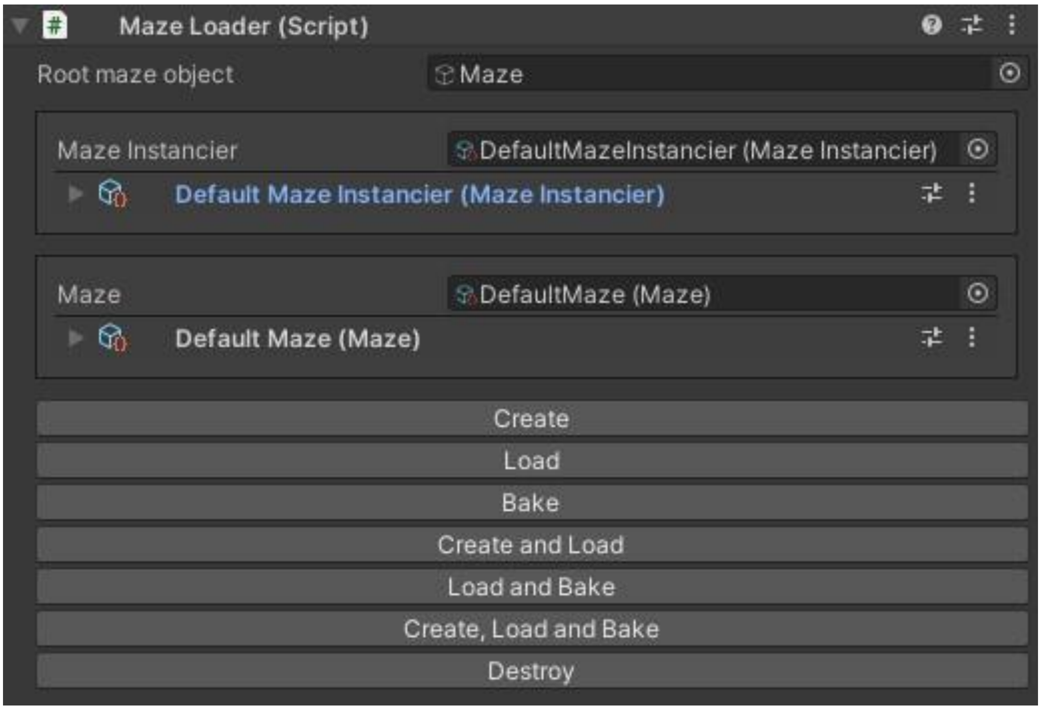


The maze instancier also determine the size parameter depending on your prefabs size, your prefabs must have the same size on X axis and Z axis and all prefabs must be the same size, the 2 previous parts are 2 unit on X and Y Axis so the size parameter of the maze instancier must be 2.

The last parameter is the group resolution it is used for optimization, if you chose to bake the maze, all static object will be combined to limit the number of draw calls with a group. With a resolution of 50 it will get a group of 50 by 50 maze part and combine the objects (this method use ram and vram to store the object so backing to much parts will increase the usage of ram and vram).

4. Maze loader

The maze loader is a component to load the maze representation created by all the scripts described before, you can load a maze by your own script, but it is a basic script if you don't want specific loading method



Maze Loader (Script)

Root maze object: Maze

Maze Instancier: DefaultMazeInstancier (Maze Instancier)

Default Maze Instancier (Maze Instancier)

Maze: DefaultMaze (Maze)

Default Maze (Maze)

It holds a maze instantiator and a maze representation that determine the generation and the instantiation of the maze.

You can also specify a root maze object which all instantiated part will be child.

It contains 4 action you can do, “Create” which create the maze with the behavior defined in the maze representation, “Load” which instantiate all prefabs put in maze instantiator depending on the maze created, “Bake” which is a performance boost by combining multiple parts together, and “Destroy” to simply destroy the children of the root object.

Using button which load and bake at the same time help loading big mazes otherwise it would be impossible to load them.