MANUAL

CAVE GENERATOR

A Unity Game Engine Tool
to generate 3D caves

Prepared By:
John Dennis Thottam

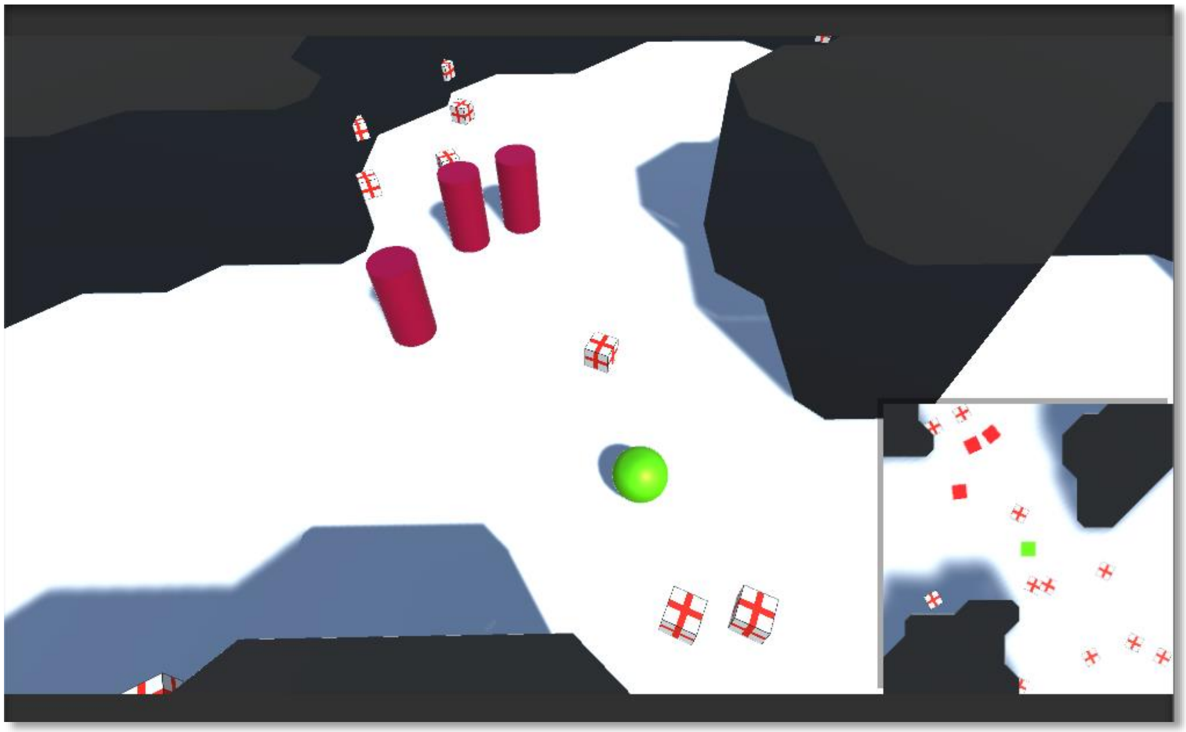
Last Updated:
19-11-2017

TABLE OF CONTENTS

Project Description

Cave Generator is a tool which generates random patters of cave with user defined dimension. Everything is generated programmatically so that no 3d software touch is required in most cases.

It also enables a user to save the generated cave to project folder, add pickup items to cave, add a AI which roams the cave automatically, add a player controlled character and the best of all, a nice mini-map to explore the cave further.



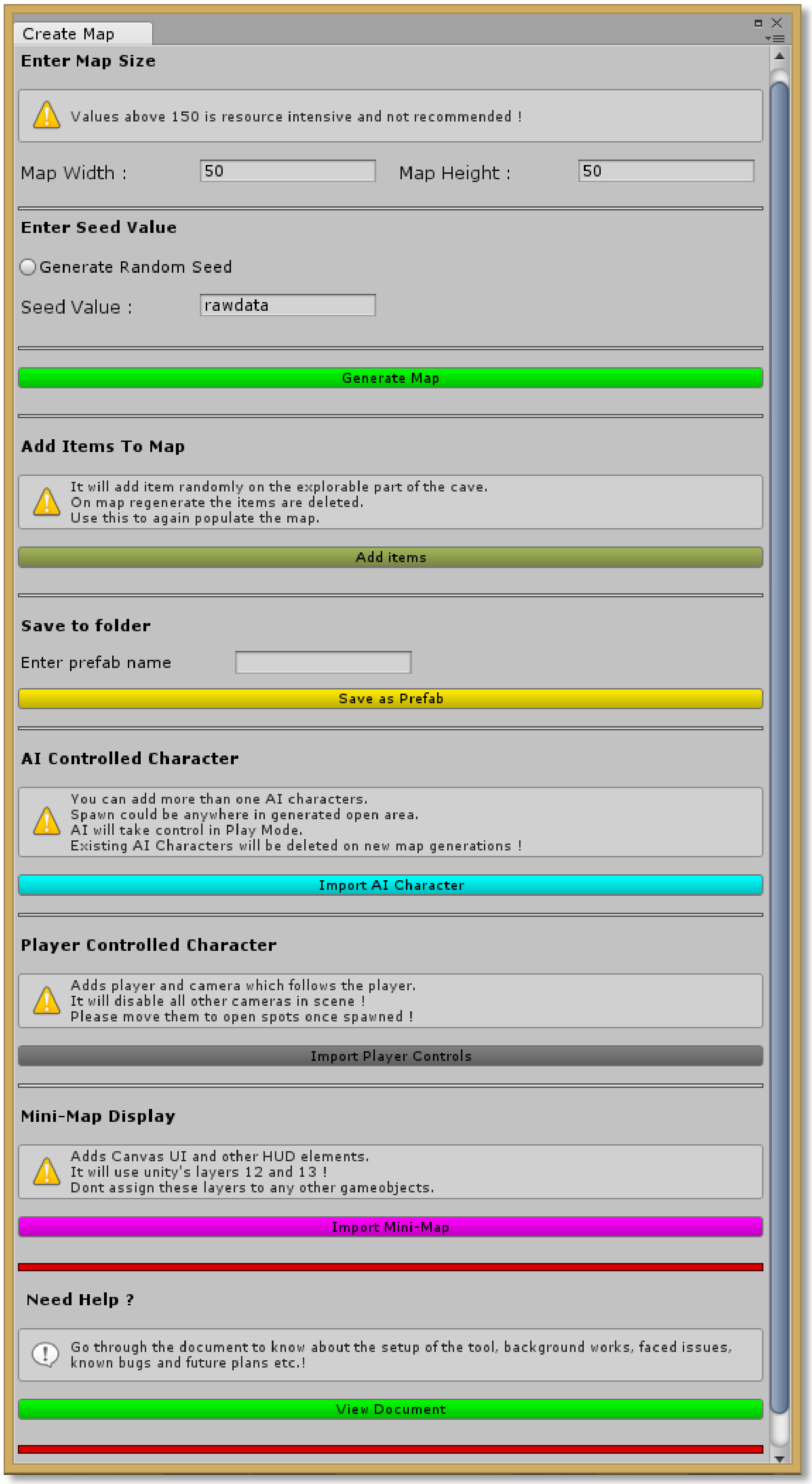
As per the instructions no external assets were used. So most of the objects used would be Unity Primitive shapes ie sphere, cylinder, cube etc.

Getting Started

With this bundle a unity package with name "**cave generator**" is given. Import it to your project to add the Cave Generator Tool to your project. Along with the unity package, a fully functional demo project is given so that you can go through how the system is implemented or in case you come across any issues.

The cave generator tool was **made using Unity 2017.1.1f1**, however it should be working smoothly in later versions as well as older versions (only unity 5).

Screen capture of the Cave Generator Tool in Unity.



Create Map

Enter Map Size

Values above 150 is resource intensive and not recommended !

Map Width : 50 Map Height : 50

Enter Seed Value

Generate Random Seed

Seed Value : rawdata

Generate Map

Add Items To Map

It will add item randomly on the explorable part of the cave.
On map regenerate the items are deleted.
Use this to again populate the map.

Add items

Save to folder

Enter prefab name

Save as Prefab

AI Controlled Character

You can add more than one AI characters.
Spawn could be anywhere in generated open area.
AI will take control in Play Mode.
Existing AI Characters will be deleted on new map generations !

Import AI Character

Player Controlled Character

Adds player and camera which follows the player.
It will disable all other cameras in scene !
Please move them to open spots once spawned !

Import Player Controls

Mini-Map Display

Adds Canvas UI and other HUD elements.
It will use unity's layers 12 and 13 !
Dont assign these layers to any other gameobjects.

Import Mini-Map

Need Help ?

Go through the document to know about the setup of the tool, background works, faced issues, known bugs and future plans etc.!

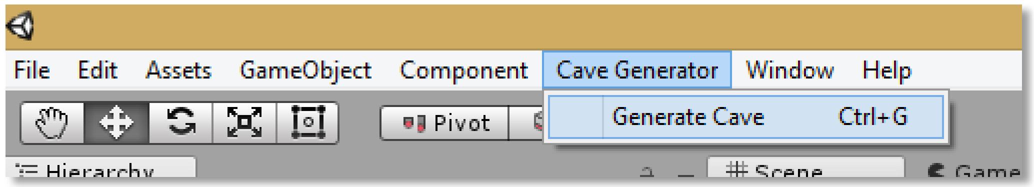
View Document

Set Up Scene

It iss very easy to integrate the tool into your project. Open your project and follow the steps given below:-

i. Import Unity Package “Cave Generator”
ii. A popup window shows list of assets/dependencies it is going to import. Select all.
iii. You will now have a “Map Generator” item in Unity toolbar
iv. Click on “**Cave Generator**” and again click “**Generate Cave**” to load up the Cave Generator tool.
OR
use **ctrl + G** to open the tool using shortcut

The tool window can be docked with any other panel or can be used as a window itself



NOTE 1: - The functionalities of Cave Generator Tool would interfere with some existing game objects such as cameras etc. It is done to ensure the smooth working of the demo so that you will get a better understating of its features. Sorry for any inconvenience caused. However, you can change this functionalities by modifying the source code.

NOTE 2: - Don’t forget lights. Add at least one light source so that we can see what’s happening around!

Thought Process

The objective was to implement a tool which can be used via unity editor to do the following functionality:-

- Make a 3D cave of random patterns without any isolated section
- Save the cave as a prefab so it can be reused anywhere
- Put random items in the cave which can interact with player
- Implement an AI Character which roams the walkable area of generated 3d cave
- Implement a mini-map where the cave, player, AI character and pickup objects are shown.

The primary objective was to implement each functionality as an individual module. Once the functioning of modules are ensured, combine the functionality into a tool so that it will be very easy for a user to use and debug.

The usage and implementation of algorithm which forms the backbone of the cave generation was the top priority. Once it was done, the focus was on implementing the other modules such as AI navigation, minimaps etc. Once those were completed, the work on Editor Tool began. There was some difficulties at the beginning but with the help of unity documentation things went relatively easy.

Algorithms Used

Algorithms played a very important part in implementing the functionality of the tool at various levels. The core algorithms which were used are as given:-

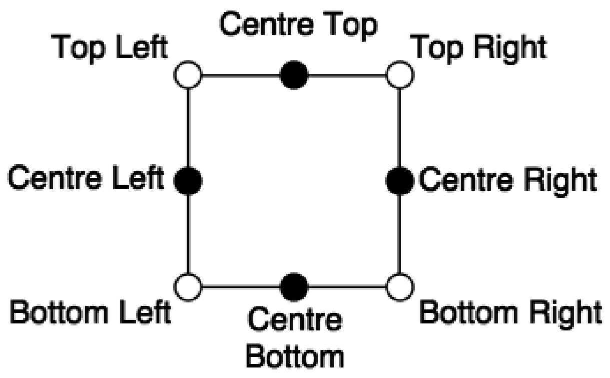
- **Single cell automata**

  Used to create a map with a pattern of fill and empty. Basically 0s and 1s to form a random 2D pattern. Once a random pattern is formed the pattern is smoothened out by assigning 0 or 1 to the cell value by analyzing number of non-empty cells surrounding this cell respectively. If the number or count is above 4, vale 1 is assigned else 0 (depends on the random value too). After that the individual caves are shaped out and in next pass a path is plotted between this island caves to connect them.

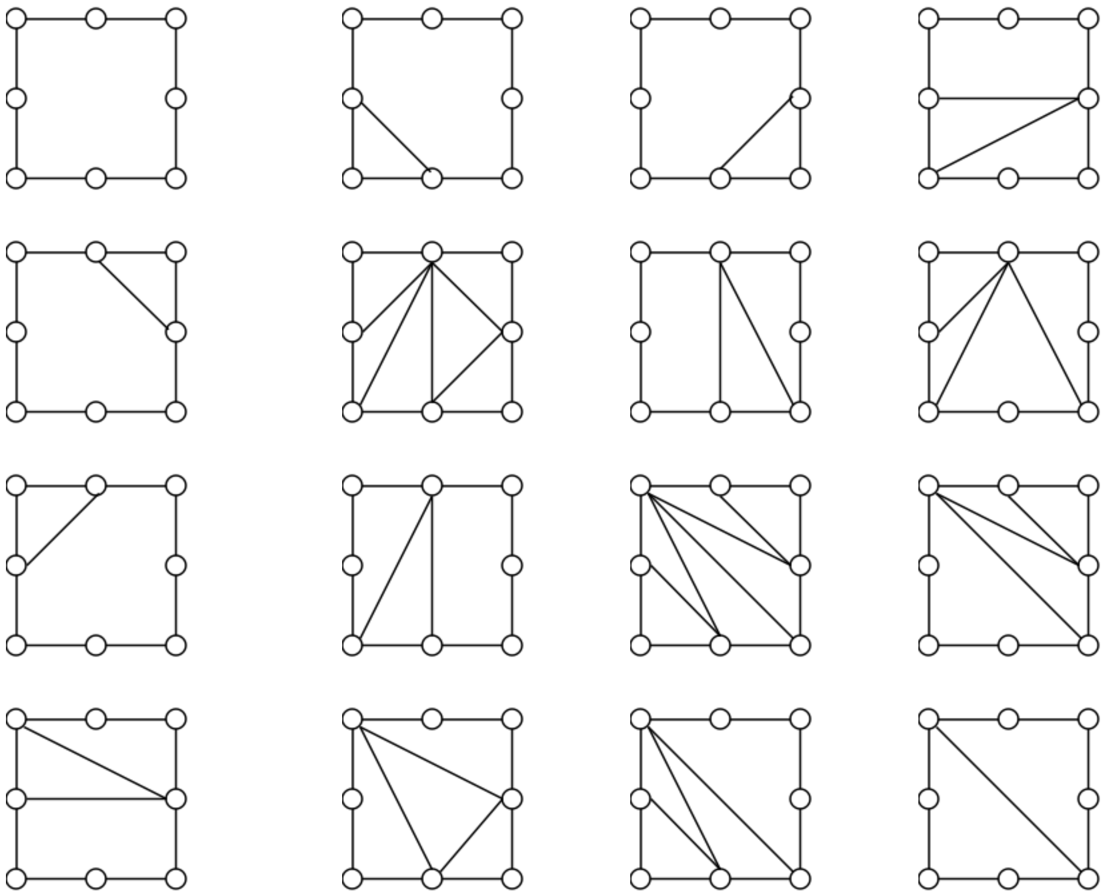
- **Marching Square**

  Here the 2D data is converted to 3D. Each cell is checked if it's a wall or not. And if it's a wall there are 16 types of wall formation and a correct one is placed. Each cell in the map has 4 points and 4 mid points represented in binary. 3D is created by forming triangles using vertices.

  A single cell representation :-



The 16 variations the cell can be represented in is as given below :-



- **Flood Flow Algorithm**

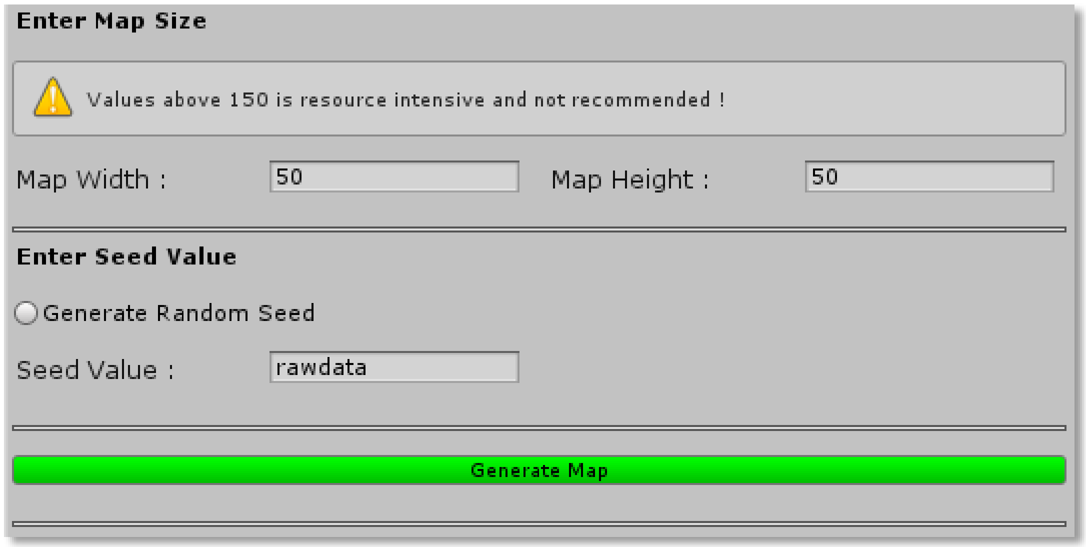
It is used to detect the regions formed by the map generator. Once the regions are detected, they are connected to form a passage between each regions. Thus making a connective cave.

In flood flow algorithm, a known cell is specified and is added to a queue. It then go through a loop and then access cells surrounding it. The accessed or checked tiles are added to a separate collection. Once the loops are complete, the region or room cells are obtained as output.

Core Functionalities

The main functionality of the tool is described here and how it's being implemented.

- **Generate Map**



Enter Map Size

Values above 150 is resource intensive and not recommended !

Map Width : 50 Map Height : 50

Enter Seed Value

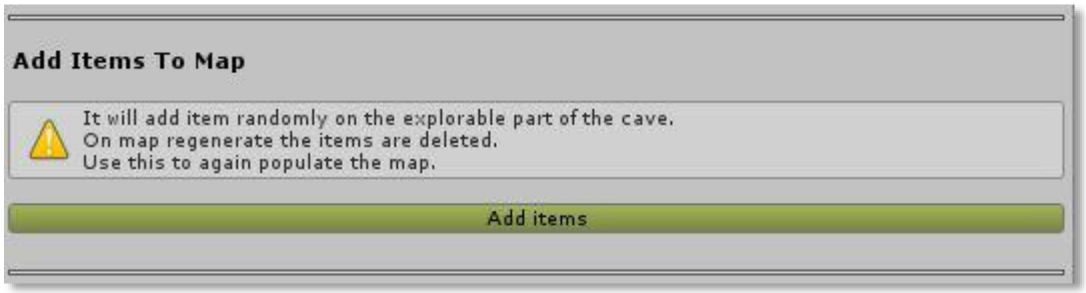
Generate Random Seed

Seed Value : rawdata

The main task of this functionality is to produce a 3d smooth and connected cave in the unity scene with intact colliders. It is implemented using Single Cell Automata algorithm and marching Square algorithm. High map values can cause excessive usage of resources for computational purpose.

Usage is relatively simple. Open the tool , enter **map width (X)**and **height (Y)** , then select **Generate Random Seed** if required. Once done click on **Generate Map.**
The 3d cave is implemented on the unity scene and is accessible via hierarchy.

- **Add Item**



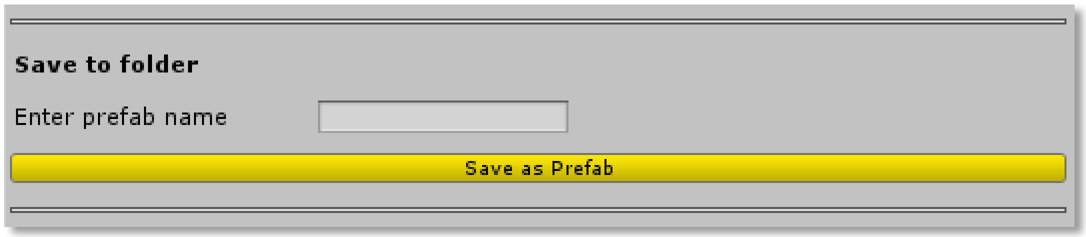
Add Items To Map

It will add item randomly on the explorable part of the cave.
On map regenerate the items are deleted.
Use this to again populate the map.

It basically populates the map with collectible kind of objects which a player can collect. Implementation is done by storing coordinates of empty caves in an array and via a randomize function, implementing the collectible on random locations. It is regenerated for each map variations.

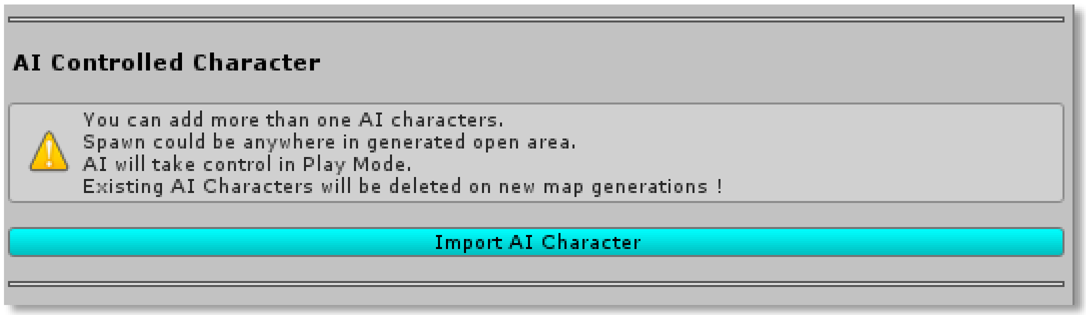
Once map is generated, click on **Add items** to populate the map with collectibles

- **Save as prefab**



It saves the generated cave as a unity prefab in the game directory which then can be reused. Input a **prefab name** and click on **save as prefab** and it will save the cave prefab to the unity resources folder

- **Import AI Character**



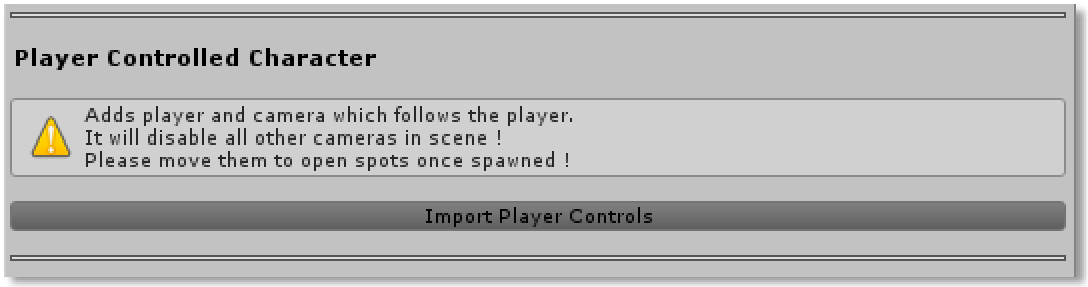
You can add more than one AI characters.
Spawn could be anywhere in generated open area.
AI will take control in Play Mode.
Existing AI Characters will be deleted on new map generations !

An AI character spawns in the map using the same coordinates stored and used by Add Item functionality. You can place as many AI characters as you like. On play mode the AI characters will automatically roam around the map.

Click on **Import AI Character** to place the AI character in the generated cave. On each map regeneration the AI characters are deleted and you have to place it again.

The AI character game object consist of a cylinder and a quad. Quad is used to represent the character via mini-map and uses unity layers functionality.

- **Import Player Controls**


Player Controlled Character

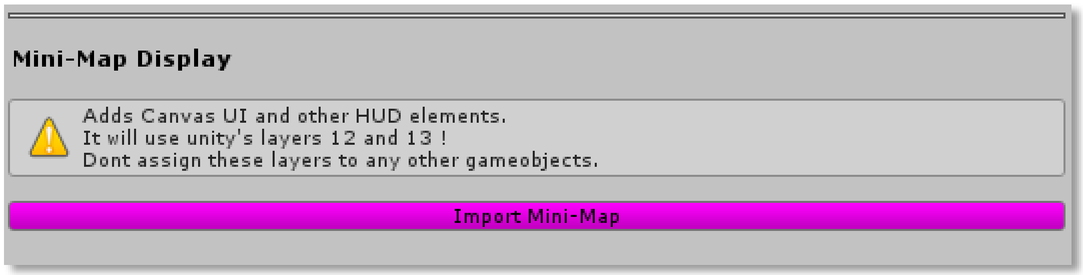
Adds player and camera which follows the player.
It will disable all other cameras in scene !
Please move them to open spots once spawned !

It imports a character which a player can control using WSAD key. The player character is represented as a sphere which can move around the generated cave. Because of the complexity involved with the camera, you might have to manually position it. The spawn point is at origin.

On clicking **Import Player Controls**, a sphere with script attached, a quad used to denote player in mini-map, a perspective camera with respect to player and a top down camera are spawned. Perspective camera acts as the main camera in play mode and top down camera is used for mini-map functions. Quad is assigned a layer so as to enable visibility.

Going to play mode, you can control the **sphere** using **WSAD keys** and collect all the collectibles.

- **Import Mini-Map**

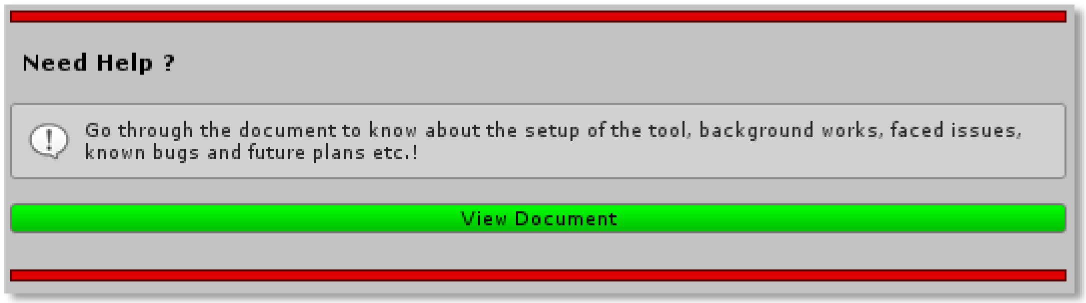

Mini-Map Display

Adds Canvas UI and other HUD elements.
It will use unity's layers 12 and 13 !
Dont assign these layers to any other gameobjects.

This functionality adds a canvas UI, a shared material asset which are required for the functioning of mini-map. A quad layer is assigned to represent the AI Character, collectibles and player on map.

Mini-map is visible via main camera on bottom right. Player, AI Character and collectibles are represented and updated in real time.

- **View Document**


Need Help ?

Go through the document to know about the setup of the tool, background works, faced issues, known bugs and future plans etc.!

Access this document via clicking the button.

Code Priority

The priority is formulated on the basis of execution and dependencies. Also to help you modify the source code as per your requirement. Scripts are located in **Assets/CaveGenerator/Scripts** directory

Level 1 priority: CustomEditorCaveGenerator.cs

This script controls the Unity tool UI and other UI related functionalities. It shouldn't be moved from this directory. Location of this script is different. It is located at **Assets/CaveGenerator/Editor** directory**.**

Level 2 priority: MapCreator.cs & MeshGenerator.cs

These scripts controls the algorithms which are being used to implement the 3d cave. Access starts from MapCreator.cs and then extends to MeshGenerator.cs .

Level 3 priority: rest of the scripts in "**Cave_Maker"** folder

They contribute to the functionality of Level 2 scripts

Level 4 priority: scripts attached to various gameobjects

These scripts play a part in the perfection of the tool but does not interfere in the core functionalities of the tool. For eg:- mini-map system etc.

Known Issues

- **Prefab save bug**
  The generated cave prefab is saved as the name specified in the input box. An individual map can only be saved once. That is once a map is created and saved, it can't be saved again because of reference link breakage.

  Note: - Issue occurs only with saving the same generated cave pattern two times in a row.

- **System hangs at high map values**
  If the input map values are high (more than 100) , the Unity Engine or system would freeze for few seconds. This is because of the computation involved in generating the procedural cave. Everything will be back to normal after few seconds.

Future Scope

The tool has immense potential for future applications in generating procedural cave. However it would be great to have these as well in future.

- Use of multi-threading for fast creation of map
- Adding different type of objects in the cave to make a real 3d cave simulator
- Generating 3D cave from image input

References

Many online resources such as Wikipedia, Unity documentation and tutorials by catlike and Sebastian Lague were referred to know more about algorithms and to implement this project.