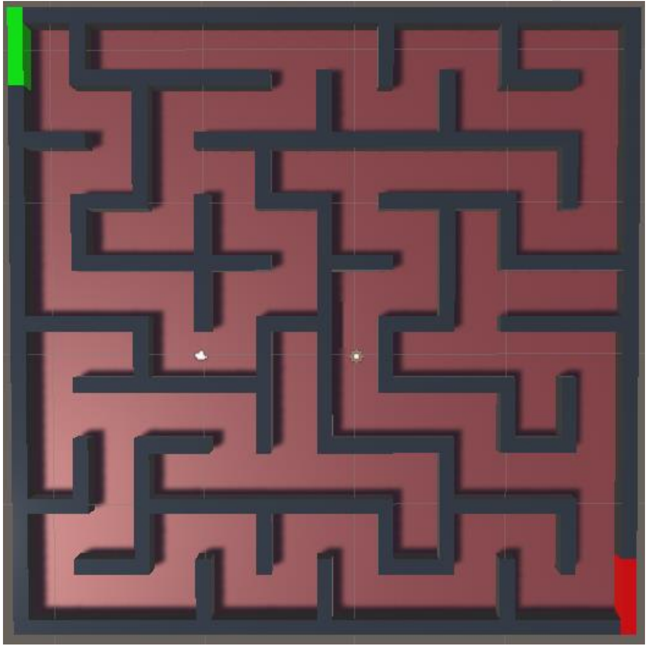
How to use MazeGenerator

Description

MazeGenerator is free demo version, it is using for creating custom 3D maze. You can use custom wall, floor and ceiling textures to create your custom 3D maze. Maze is using backtrack algorithm.
Maze have one entrance and one exit, as it shown on the picture 1.



Picture 1. Maze example with entrance(green wall) and exit(red wall).

You can get access to maze, when scene is active, example: you want change level in your game. Maze script will help you to solve this task.

- At first step you need to get gameObgect "MazeGenerator"
- At second step you need to get maze script
- And at third step you can use public method's of MazeScript for your tasks.

Main functions of script:

1. Delete maze
2. Create maze new width and height
3. Create maze with seed(to create same mazes)
4. Set new prefab for wall, ground, ceiling, start and end walls
5. Get all impasses (cell center positions in array of Vector3)
6. Get all crossroads (cell center positions in array of Vector3)
7. Get all cell centers (cell center positions in array of Vector3)
8. Get way from start to end (cell center positions in array of Vector3)

9. Get gameObject of the start and end wall
10. Verify the position on the wall
11. Add custom rooms with width and heigth

In demo version you can see how to use maze generator and maze script to create your game!

User’s manual

If you want add a maze to the scene, you need to use MazeGenerator prefab(pic.2), you must transfer it to scene from folder “Prefabs”.

Picture 2. MazeGenerator prefab.

On next step you can chose: use prefabs or get skeleton of maze.

If you chose prefabs mode, you must set wall prefab(size 1x(your depth)x1), floor(ground) prefab, also you can set ground, ceiling, start and end wall prefabs(pic.3).

Picture 3. Maze options. Also you can set maze width and height, maze start position(position maze center, pic.4). If flag “Create maze on start” is set – maze will be created after initialization of the MazeGenerator object. If flag “Create ceiling” is set – ceiling will be created.

Picture 4. Start position.

Scripting

List of public variables:

Prefabs:

```
public GameObject wallPrefab, groundPrefab, ceilingPrefab, StartWallPrefab,
EndWallPrefab;
```

Maze size in cells:

```
public int MazeWidth = 10, MazeHeight = 10;
```

Position of left top maze corner

```
public Vector3 startPos = new Vector3(0, 0, -10);
```

Flags to create maze on start and flag to create ceiling:

```
public bool CreateMazeOnStart = true, createCeiling = false;
```

Tunnel width and wall width(set wallWidth = 1 as constant)

```
public float TunnelWidth = 1;
public float WallWidth = 1;
```

Seed variables for random creation(if UseSeed is false, all mazes will be random)

```
public bool UseSeed = false;
public int MazeSeed = 1000;
```

List of public functions:

Function to set flag for create Ceiling

```
public void CreateCeiling(bool create)
```

Function for set wall prefab

```
public void SetWallType(GameObject newWall)
```

Function for set ground prefab

```
public void SetGroundType(GameObject newGround)
```

Function for set ceiling prefab

```
public void SetCeilingType(GameObject newGround)
```

Function for set start wall prefab

```
public void SetStartWallType(GameObject newGround)
```

Function for set end wall prefab

```
public void SetStopWallType(GameObject newGround)
```

Function returns size of the maze(in cells)

```
public Vector2 GetMazeSize()
```

Function for deleting all maze objects

```
public void DeleteMaze()
```

Function for delete maze and create new maze with new width and height(in cells)

```
public void ClearAndGenerateNewMaze(int newMazeWidth, int newMazeHeight)
```

Function for creating new maze with new width and height(in cells)

```
public void GenerateNewMaze(int newMazeWidth, int newMazeHeight)
```

Function for creating new maze with new width(in cells), height(in cells), position(in cells) and seed for random.

```
public void GeneratemazeWithSeed(int newMazeWidth, int newMazeHeight, Vector3 position,
int seed)
```

Function return list of all impases in maze (cell center positions, pic.5)

```
public List<Vector3> GetImpasses()
```

Picture 5. Shows all impasses.

Function return list of all crossRoads in maze (cell center positions, pic.6)

```
public List<Vector3> GetCrossRoads()
```

Picture 6. Shows all crossRoads.

Function return list of all cell centers in maze (cell center positions, pic.7)

```
public Vector3[,] GetGridCenters()
```

Picture 7. Example of way from start to exit. Function will return start position in maze (cell center position)

```
public Vector3 GetCenterPositionOfStartCell()
```

Function will return end position in maze (cell center positions)

```
public Vector3 GetCenterPositionOfEndCell()
```

Function will return name of the start wall

```
public string GetEndWallName()
```

Function will return name of the end wall

```
public string GetStartWallName()
```

Function will return EndWall GameObject(to delete or change properties)

```
public GameObject GetEndWallObject()
```

Function will return StartWall GameObject(to delete or change properties)

```
public GameObject GetStartWallObject()
```

Function will return way(array of Vector3 cell center position, pic.8) from start position to end(exit from maze) position

```
public Vector3[] FindWayToEnd()
```

Picture 8. Example of way to exit.

Set seed for Random function

```
public void SetSeedForRandom(int s)
```

Set using seed random

```
public void SetUseSeed(bool useSeed)
```

Check position, return true if wall is at position

```
public bool IsThereWall(Vector3 position)
```

Get room position function(returns list of all cells from all rooms)

```
public List<Vector3[]> GetRooms()
```

Get walls position function(returns list of Vector3 position of walls to create them)

```
public List<Vector3> GetWallArray()
```

Help

Next code will help you to get the gameObgect “MazeGenerator”:

```
private GenerateMaze maze;
void Start () {
```

```
maze = GameObject.FindGameObjectWithTag("Maze").GetComponent<GenerateMaze>(); }
```

Next code shows how you can get start position:

```
Vector3 startPos = maze.GetCenterPositionOfStartCell();
```

Next code shows how you can create new maze:

```
public void OnGenMaze()
{
mazeScript.ClearAndGenerateNewMaze(20, 20);
Vector3 startPos = mazeScript.GetCenterPositionOfStartCell();
// teleport player to start position
player.transform.position = new Vector3(startPos.x, startPos.y+3, startPos.z);
}
```

If you have some questions, you can write me on e-mail:

radiomaster71@gmail.com