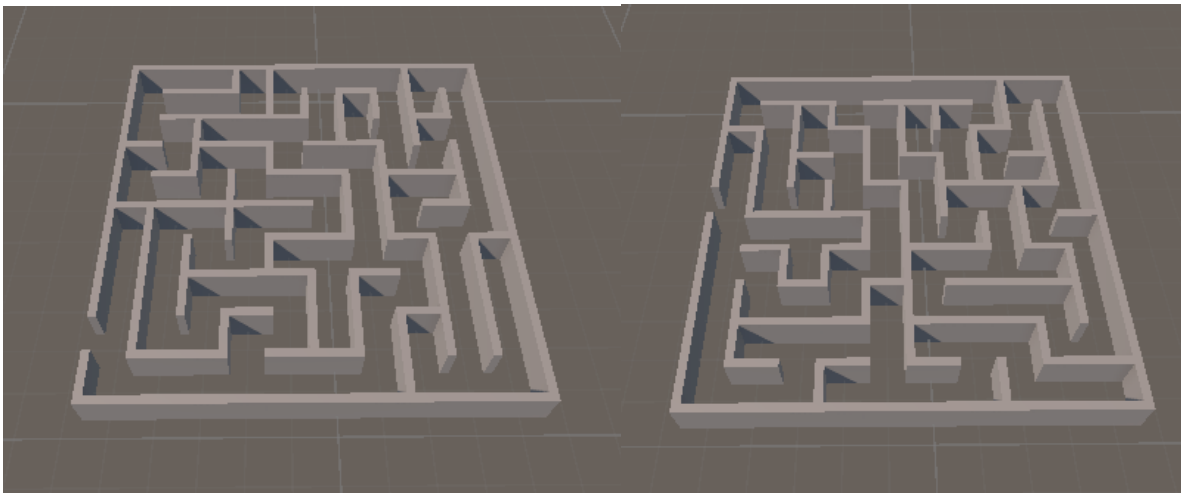
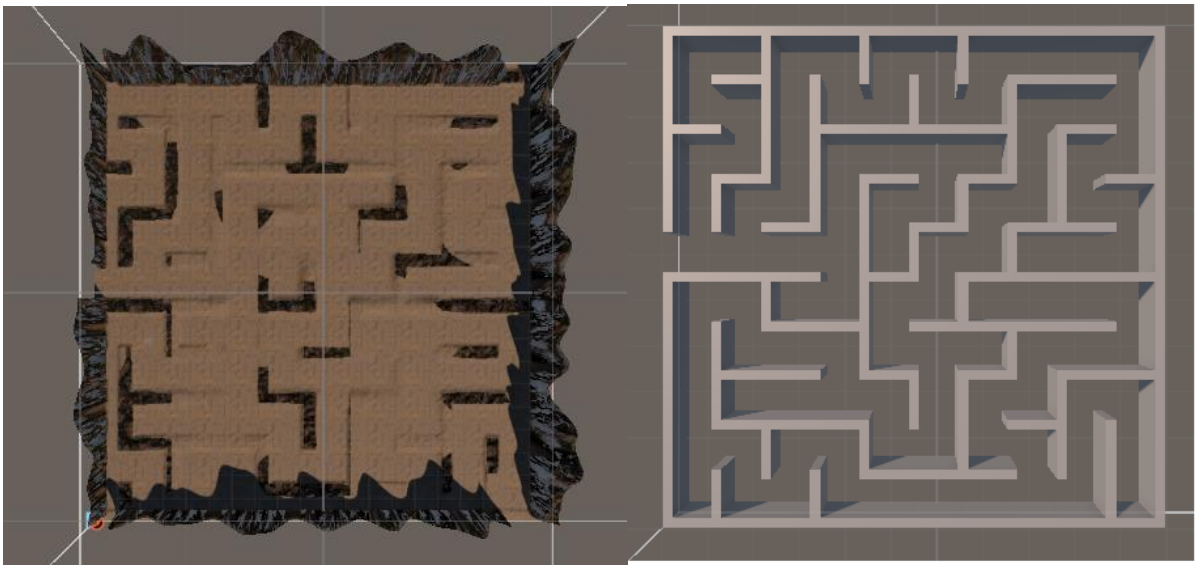
Labyrinth Generation

The core mechanic that sets this game apart is the randomly generated labyrinth. This creates the necessity that every other GameObject such as the player, the enemies and the power-ups also have to be procedurally generated. Using a backtracking algorithm (which I implemented), we produce a different maze each time based on a seed value from 1 to 1000 (we chose 1000 as an example, this range could easily be increased or decreased). Thus, we can have 1000 unique levels, each with its own maze. For example:



However, the above diagram isn't the actual game environment, it's merely a schematic. We use it to sculpt a terrain mesh whose walls trace the maze's layout.



Note that the outer walls are taller to confine the player's view within the labyrinth and prevent them from seeing Unity's default exterior environment. We applied Perlin noise to vary the heights of these peripheral walls.

The same approach is used for the inner walls, though their heights are much lower (for reasons explained below).

Finally, the ground texture changes based on elevation, making the height variations of the inner walls visually clear.

Random Envrironment Spawning

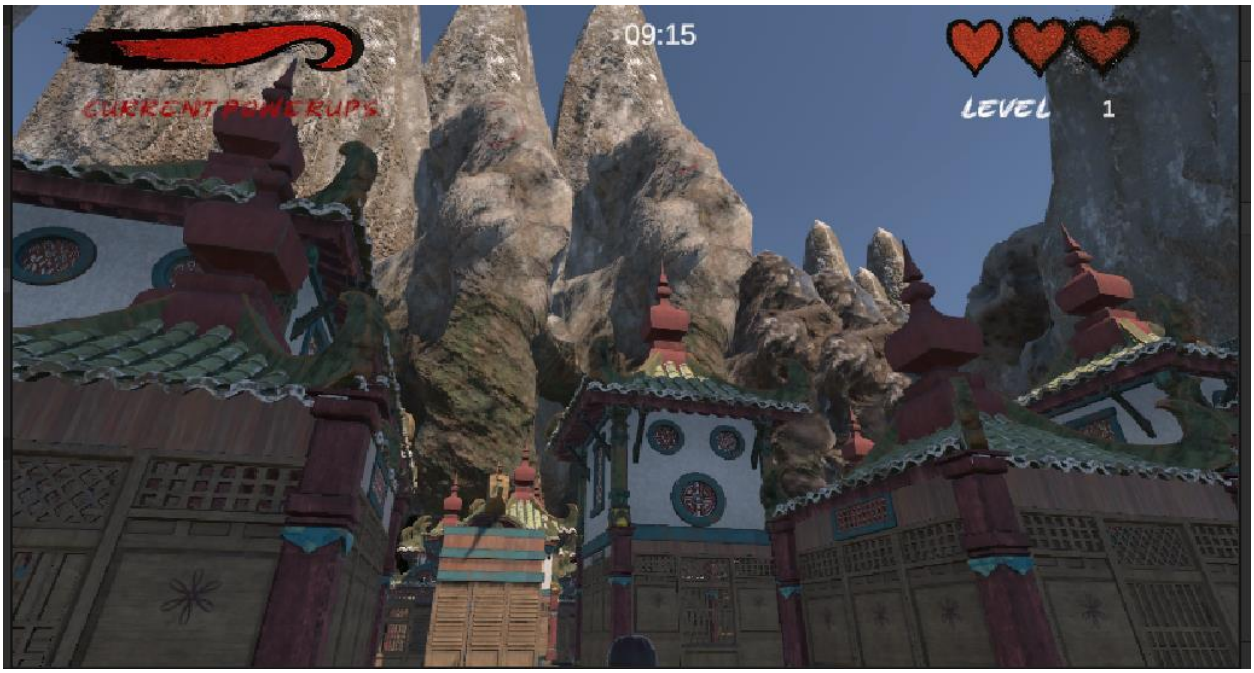
Since the inner walls are low, they don't significantly block the player's line of sight. Instead, houses spawn atop the inner wall positions, on small hills, and serve as the main visual obstacles.

Note: House placement isn't preserved in the Unity Editor, so the screenshots are in-game captures.

After the terrain is generated, a script iterates over each inner wall point, randomly choosing one of three house models for each. Because the script knows each house's dimensions, they never overlap.

You may notice small gaps where a house could be, this can be fixed by tweaking a variable, but we opted for a lower density to maintain performance. Even with occlusion culling enabled, too many textures and meshes can strain the scene. Nevertheless, the labyrinth's aesthetic remains intact.

To complete the environment, we randomly place large rocks and boulders along the outer walls to evoke surrounding hills, further confining the player in a realistic space.



All this setup runs before the game starts, ensuring the player finds a fully prepared environment.

Start Menu

The main menu offers 4 buttons:

- Start Game: Begins the game.
- Quit Game: Exits the application.
- Controls: Shows a text overlay explaining the controls.
- Settings: Provides simple audio and graphics options.

In Settings, the player can adjust music volume and toggle fullscreen mode.

Pause Menu

Pressing Escape during gameplay brings up the pause menu with the same options as the Start Menu, but here the entire game simulation is frozen

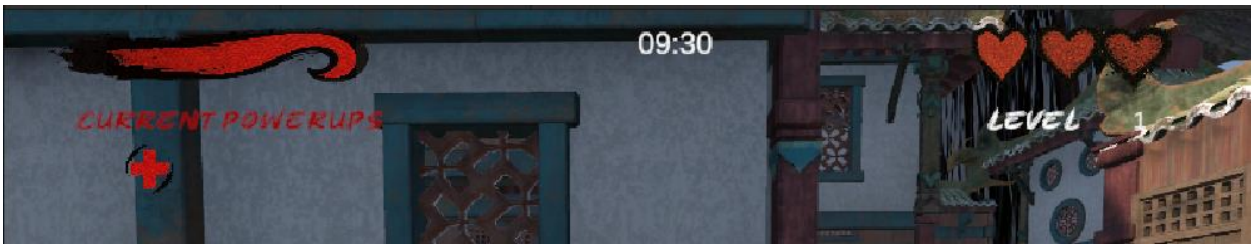
Player Movement

- W: Move forward
- S: Move backward
- Mouse Left/Right: Rotate character
- Mouse Up/Down: Tilt camera (without affecting character orientation)
- Space: Jump
- Left Click: Sword attack

This camera tilt lets the player admire the environment without reorienting the character itself.

HUD

- **Top Left:** Health bar. Each hit from an opponent reduces it, if it reaches zero, the player loses one of three lives.
- **Top Right:** Remaining lives. Losing a life respawns the player in place with full health.
- **Below Health Bar:** Up to three power-up icons appear when obtained, they disappear when their effect ends.
- **Bottom Right:** Current level number.
- **Top Center:** Countdown timer to find the level's exit.



Game Over

Two conditions trigger Game Over:

1. Losing all three lives and then dying once more (four total deaths).
2. Running out of time.

In either case, the Game Over screen offers to return to the Main Menu (restarting at level 1) or exit the game.

Power Ups

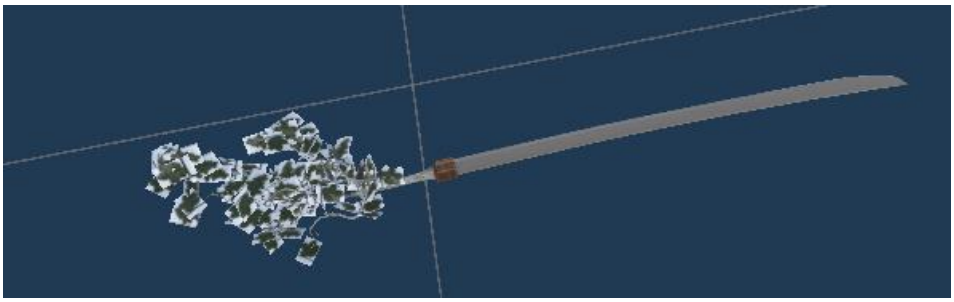
There are three types of power ups:

1. **Health Restore:** Instantly refills the health bar to 100%.

2. **Speed Boost:** Doubles movement speed for 5 seconds.
3. **Top-Down View:** Toggles between third-person and a fixed top-down camera.

Each uses the same sword mesh so the player can't tell which they'll receive. Icons: cross = health, arrow = speed, camera = top-down.

- Speed boost lasts 5 seconds, its icon disappears afterward.
- Health restores is instantaneous but its icon remains for 5 seconds so the player notices they got the power up.
- Top-down view persists for the entire level. Press Q to toggle. When active, the camera can't be tilted on the Y-axis, preventing an easy overview of the maze.



The way the player picks them up is simple. When the player gets close enough to the power-up GameObject, it moves toward them automatically, no input required. As soon as their colliders touch, the upgrade is granted. Moreover, their movement isn't affected by other colliders because they have no Rigidbody, so they can even pass through houses.

Finally, all three are randomly generated before the game starts, and in each level there's exactly one of each type, just placed at different, random positions.

Opponents

For opponents, we use UnityEngine.AI's NavMeshAgent for easy control. They have three states:

- **Patrol:** Move in a random direction, searching for the player.
- **Chase:** Once they've detected the player, they pursue until they're close enough to attack.
- **Attack:** When near the player, they deal damage.

Each successful hit removes 20% of the player's maximum health bar and is the only way to reduce the player's health. Opponents switch states automatically based on their distance from the player.

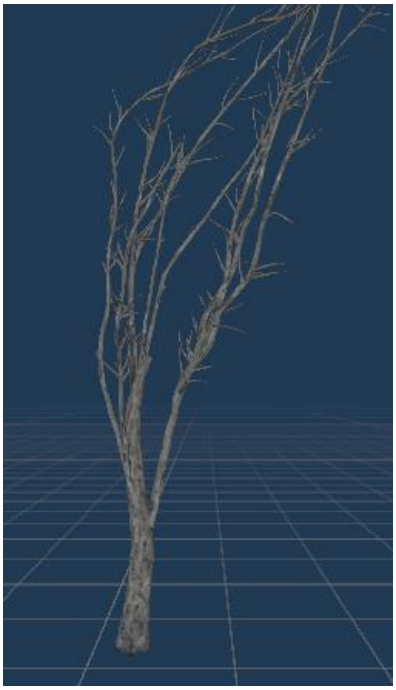
They're spawned at random locations, three opponents in total per level. The player can kill an opponent by left-clicking to attack: if the attack hits one (or even multiple), that opponent instantly disappears, since they have no separate health bar. Opponents can also pass through houses.

Player Spawn

The player object isn't in the scene at start; instead, we instantiate them at the end of the first frame, near the maze's entrance and elevated, ensuring they don't spawn inside a house (at worst, they fall onto one).

Game End

The exit is represented by a tree, placed randomly based on the player's spawn. Touching the tree completes the level and prompts the player to proceed. The HUD's level number then increments for the next stage.



Difficulty

Each new level reduces the time limit by one minute (down to a approximately 2-minute minimum), gradually increasing difficulty.

Assets

1. Houses, rocks, sword mesh: Chinese Modular House HDRP Free - Erbeilo 3D.
2. Player model: Red Samurai it also includes the animations.
3. Opponent models: kcisa - Korean Traditional Martial Arts .
4. The images wer sourced online and edited in Photoshop
5. The music is downloaded from Pixabay.