# Game Proposal: Nowhere to Hide

## 1. Game Description

The main goal of the practical course “Computer Games Laboratory” is to design and develop a video game in teams of three to four people with a defined theme and some technical achievement. This semester’s theme is “Slender Man”.

The original “Slender Man” game is a first-person horror game, where players have to collect randomly placed notes in a dark forest while evading the Slender Man. In the game, the Slender Man is portrayed as a tall entity that wears a black suit and boasts unusually long limbs. When moving, he is not walking explicitly, but teleporting or hovering above the ground. As one of the most disturbing features, the Slender Man lacks any kind of facial features and only has a white blank face. In other media, like films or comics, the Slender Man is also portrayed as a sinister character that the heroes of the story have to run away and escape from. He also often corrupts and overtakes other living beings.

With this characterization of the Slender Man in mind, we came up with the idea of a dungeon runner, where players are chased by the Slender Man and have to cross a labyrinth in a rapid fashion while evading traps and making fast-paced decisions. According to this general idea, we decided to name our project “Nowhere to Hide”.

### *1.1) Gameplay*

The game is designed as a fast to pick up, casual mobile game that instantly pulls players into a flow state with its easy to understand yet challenging gameplay.
This is achieved by presenting players with a pool of limited decisions in the manner of quick-time events while passing the labyrinth. The rapid flow and imminent threat of the Slender Man requires fast thinking that results in immediate feedback.
Gameplay, level design and visual effects are supposed to create a sense of dread in players while running through the labyrinth that is characteristic for the “Slender Man” genre.
The game is level based. The end goal is a door that needs to be unlocked along the way. Each level introduces new mechanics and power ups that can be gathered. If players get caught by Slender Man, they get reset to the first level.
Passing the labyrinth consists of two phases: The running-phase and the fighting-phase.

#### *1.1.1) Running Phase*

Central goal of the running phase is to reach the end of the labyrinth while being chased by the Slender Man. To this end, the player is faced with a series of quick time events where fast decision making is needed. These events include which way to go at a crossroad or to turn into a passage, trying to get treasures along the way,

but potentially being slowed down in the process, and other actions like rolling or swinging on a rope.
Some options are not available when first passing the quick time event (like locked doors, barricaded pathways), and require special items and skills that can be found in the labyrinth.
The expected results of each decision, e.g. whether treasure can be found when running along a longer path, are not random but telegraphed beforehand in order to promote quick decision-making and planning.
The labyrinth layout itself is procedurally generated and differs between each run. We require the generated levels to include loops and shortcuts to give the player the option to go back to previous locations and to open secret passages and locked doors with newly found items or keys. This also encourages the player to pay greater attention to the level's layout and to function as an additional skill required by the player.
As an aid, a mini map showing already visited paths and encountered obstacles might be enabled.

In the running phase, players are automatically moved forward but they have free control to move left and right on the path to evade obstacles in order to gain speed and an advantage over the Slender Man.

The camera is placed over the character's shoulder to enable a third person view and allow for a good overview of the path ahead to evade incoming obstacles and think about upcoming events ahead of time. In the same manner, each labyrinth map should be open to enable players to see ahead. As potential candidates for such levels, settings like meadows or open woods and abandoned villages can additionally introduce a feeling of loneliness and helplessness against the chasing Slender Man.
Additionally, the over-the-shoulder camera position enables players to notice events of the chasing Slender Man (like pushed away boxes) and even the Slender Man himself, when he closes in. Just like in the original Slender Man games this creates a sense of tension and underlines the gravity and urgency of the current gameplay.

One of the prevalent characteristics of the Slender Man is to corrupt and overtake other beings. Thus, we want to include some sort of fighting sequence in our game. To this end, random monster events can appear and switch the game to the fighting phase, while players are traversing the labyrinth.

*1.1.2) Fighting Phase*
In this phase, players engage in a fast paced fight. Players can choose attacks from a small pool of actions. They can also benefit from items and buffs found during the running phase to make this part of the game easier.

Enemies attack in a bullet hell-pattern and spawn projectiles that can be evaded by moving in a 2D fashion.
In order to not change the pacing and flow of the game when changing from the running phase to the fighting phase, players are fighting while running through the labyrinth and moved forward by the game at a constant speed.
All fights need to be finished quickly (e.g. before reaching a crossroad on the path), otherwise the enemies win automatically.

### *1.2) Items and Skills*

- Open new ways and affect the map
    - Keys in the form of letters in regard of the original Slender Man game
    - Switches
    - Ropes
    - Bombs to get through barricades and upgrade fighting skills
- Boots for faster running and to gain an advantage over the Slender Man, which enables to take more risks and longer ways in the future
- Power Ups to strengthen the player in the fighting phase
    - Wings to move and doge the enemy projectiles faster
    - Sword, Axe, Machete to upgrade the player's attacks and cut through obstacles

## 2. Technical Achievement - Procedurally Generated Map

Every time players start a new run, they should get a different maze. To not have to build hundreds of different levels manually beforehand, the maze will be procedurally generated. We require the generating algorithm to use predefined tiles, for example a crossway with two options in the quick event. Combining a few structures with many different appearances can lead to a variety of environments. The core structure of one tile is always the same, we chose a multiple of 20x20 meters, balancing variety and algorithmic complexity.
New maze parts are built randomly out of a combination of these tiles, fitting together seamlessly . Each tile can be filled with objects at random. These objects can be obstacles, chests and enemies, but all of them in the size of a multiple of 1x1 meters. Items and rewards should be chosen fitting the current situation.
There are two different options to utilize procedural generation in our case. The first one would be that the map is randomly generated while players are running through it, adding new content along their way.
Another approach is to generate the whole map before the run starts.
We decided to choose the second option and aim for a level based maze instead of an endless runner.
We also require the possibility to run in circles to go back to closed doors/chests, where players have now found the key.

By showing players an abstract mini-map, we increase the clarity of the labyrinth. The minimap is an approximate representation of the level, so the relative room sizes don't have to be the same and it doesn't show the whole level at the same time.
Each level ends at a door that has to be unlocked in order to progress.
With each level the difficulty increases. The difficulty can be adapted by running speed, maze size and adding more actions to choose from.

Overall difficulties to consider are achieving a coherent look without being repetitive. Also, tiles need to blend together, so that there is a consistent maze with loops and not too many dead ends.
A balance between the difficulty of the maze and the speed of the Slender Man is crucial for our gameplay.

From an implementation standpoint, the Slender Man is just a value for how far he is away from the player, indicated by sound and graphic effects. The Slender Man Model itself is just to be seen when he catches the player and does not need to be simulated during the run.

## 3. "Big Idea" Bullseye

The core of our game consists of a casual and fast-paced running game, where players have to complete a set of simple quick time events to progress further into the level. When failing these simple but quick events, the Slender Man catches up to them and the level is reset.
The levels should be completed quickly (a couple of minutes per run) and be very challenging. Not performing a required interaction should directly result in a fail state.

Our main technical achievement will be procedural level generation. To begin with, we will create our tileable prefabs that can be put together to a level. The procedurally generated level has to fulfill some basic constraints: It needs to be solvable and should not contain any obstacles, dead ends or traps that are impossible for players to avoid or overcome. When these basic constraints are met, there is another layer of desirable properties: The generated levels should feel interesting and not repetitive. The procedural generation should give rise to challenging situations that require quick decision making. Lastly, the final target would be to support multiple stages with increasing difficulty, introducing some way to generate larger, more difficult or more complex levels.

## 4. Development Schedule

For the development schedule, we decided to go with a responsibility-oriented hour-based kanban board. The tasks are assigned, estimated and pre-assigned based on one of the possible responsibilities (like game design, artwork, scripting,

etc.). If multiple people have similar skills for a responsibility, any one of them can assign the task to themselves. Moreover, if both of them have unallocated time, a parallel development approach can be used by both of them to complete the task faster.

An additional responsibility of the Change Control Overseer is given to someone in the team to give out tasks, track, validate and close them. This responsibility is out of scope for the schedule, as it is a continuous effort throughout the project lifecycle. The person responsible will be cycled after every deliverable, so everyone will improve their soft skills and management.

The responsibility types are:

- Scripting - invoking algorithms and combining level pieces to create a coherent game experience
- Algorithm Development - bringing the technical achievement into life
- Game Design - blocking out the level for pre-viz and decorating it for production, formalizing game mechanics
- Artwork - creating or scouting for textures, sounds, animations, 2d and 3d assets
- UI/UX - creating user interfaces and accessibility options
- Common - doing playtesting, documentation, management, reporting, etc.

The Work Breakdown Structure is based on 5 layers (minimum, low, desirable, high target, extras) and includes all of the development steps. It is combined with the WBS Dictionary for the sake of brevity.

| Work Breakdown Structure Dictionary | | | | | |
|---|---|---|---|---|---|
| Task Name | Assignee | Responsibility | Layer | Deliverables | Estimate / Reality, Deadline |
| Come up with the General Game Idea | All | Common | Minimum | Game Idea Overview | 8h |
| Compose the Game Description | Maarten | Common | Minimum | Game description text document | 2h |
| Describe the Technical Achievement | Anja | Common | Minimum | The surface-level tech. achievement description | 1h |
| Create a “Big Idea” Bullseye vision | Daniel | Common | Minimum | The Bullseye diagram and description | 1h |
| Structure the Development | Nickolas, Daniel | Common | Minimum | The responsibility description, the | 4h |

| | | | | | |
|---|---|---|---|---|---|
| Schedule | | | | Work Breakdown Structure | |
| Prepare the Assessment | Daniel | Common | Minimum | The Informal Assessment Report | 1h |
| Prepare the Gameplay sketch | Nickolas | Artwork | Minimum | Layered Gameplay Sketch Image | 2h |
| Prepare the Level & Game Over Sketch | Anja | Artwork | Minimum | Level and Game Over Sketch Image | 2h |
| Prepare the Artstyle Moodboard | Maarten | Artwork | Minimum | A set of images that represent the desired artstyle | 2h |
| Prepare the Presentation Slides | All | Common | Minimum | A Google Slides Presentation of the Game Idea Pitch | 2h |
| Proofreading, correction and validation | All | Common | Minimum | Written Report | 2h |
| Form the Game Idea Pitch | All | Common | | Game Idea Proposal, Game Idea Presentation | 10.11.2021 |
| Translate game mechanics to the paper showcase | | Game Design | Minimum | Informal Mechanics Overview | 3h |
| Train the Game Master role execution | | UI/UX | Minimum | Game Master Notes | 2h |
| Create physical character assets | | Artwork | Minimum | Character models | 8h |
| Create physical level assets | | Artwork | Minimum | Level models | 12h |
| Physical Playtesting | | Common | Minimum | Model corrections | 2h |
| Form the Prototype | All | Common | | Physical Prototype, Prototype Presentation | 24.11.2021 |
| Implement the Finite Labyrinth generation algorithm | | Algorithm Development | Minimum | A plug-and-play generation algorithm | 20h |
| Create level prefab blockouts | | Game Design | Minimum | A set of prefab blockouts for playtesting | 4h |

| | | | | | |
|---|---|---|---|---|---|
| Create or scout level assets | | Artwork | Minimum | A set of assets used for level design | 8h |
| Implement the stitching between level prefabs | | Scripting | Minimum | A continuous level generation system | 2h |
| Implement basic keyboard character movement | | Scripting | Minimum | A 1-2 axis player movement system | 2h |
| Create character models | | Artwork | Minimum | A set of models | 8h |
| Create placeholder character animations | | Artwork | Minimum | A set of generic character animations | 4h |
| Create QTE blockout choice UI | | UI/UX | Minimum | A UI for showing possible QTE choices | 2h |
| Write Interim Results Report | All | Common | Minimum | The report document | 6h |
| Create Interim Results Presentation | All | Common | Minimum | The report presentation | 3h |
| Create player stat components | | Game Design | Low | A set of stats the player has | 2h |
| Implement inventory | | Scripting | Low | A show-don't-tell inventory system | 4h |
| Create temporary item placeholders | | Game Design | Low | A set of items with filled attributes and placeholder assets | 4h |
| Implement item generation | | Algorithm Development | Low | A system that contextually spawns items | 8h |
| Place item icon placeholders | | UI/UX | Low | A set of on-screen item visuals | 1h |
| Create QTE blockout action UI | | UI/UX | Low | A UI for showing quick input actions | 2h |
| Implement character stats | | Scripting | Desirable | A stats system that affects other systems | 4h |
| Place stat icon placeholders | | UI/UX | Desirable | A set of on-screen stat visuals | 1h |

| | | | | | |
|---|---|---|---|---|---|
| Form the Interim Demo | All | Common | | Interim Results Executable, First Programming Results Presentation | 15.12.2021 |
| Write Alpha State Report | All | Common | Minimum | Report Document | 6h |
| Create Alpha State Presentation | All | Common | Minimum | Report Presentation | 3h |
| Create concrete QTE UI | | Artwork | Low | A set of UI elements to replace blockouts | 4h |
| Create Item Icons | | Artwork | Low | A set of textures for player stats to swap with placeholders | 6h |
| Scout character sounds | | Artwork | Low | A set of sounds for character movement | 4h |
| Scout item sounds | | Artwork | Low | A set of sounds for item usage | 6h |
| Implement level difficulty curve | | Algorithm Development | Desirable | An algorithm that adapts difficulty to ease the player in | 8h |
| Create Stat Icons | | Artwork | Desirable | A set of icons for stats to swap with placeholders | 4h |
| Implement a variety of Items | | Scripting | Desirable | A template item system for easy item creation | 8h |
| Create item animations | | Artwork | Desirable | Animations that show item usage | 12h |
| Implement choice visuals | | Scripting | Desirable | A system to highlight the results of player choice | 12h |
| Create stat visuals | | Artwork | Desirable | A set of models and effects to visually convey stats | 6h |
| Alpha Playtest | | Common | Desirable | Playtest Report Data | 5h |
| Playtest-based modification | | Scripting | Desirable | Improved game mechanics | 10h |
| Implement Skills | | Scripting | High | A skills system that affects levels and | 4h |

| | | | | | |
|---|---|---|---|---|---|
| | | | | movement | |
| Invent a variety of Skills | | Game Design | High | A set of skills the player can possess | 4h |
| Create Skill Icons | | Artwork | High | A set of icons that represent active skills | 4h |
| Create skill animations | | Artwork | High | A set of generic animations for skill usage | 12h |
| Place skill UI | | UI/UX | High | A UI for applying skills | 1h |
| Create item visuals | | Artwork | High | A set of models and effects to visually display items | 10h |
| Implement enemy AI | | Algorithm Development | High | A set of AI behaviors an enemy can possess | 8h |
| Create enemy templates | | Game Design | High | A set of enemy types that can be spawned | 12h |
| Implement enemy spawning | | Algorithm Development | High | A system that contextually spawns and fights enemies | 4h |
| Form the Alpha Release | All | Common | | Alpha Release Executable, Documentation and Presentation | 19.01.2021 |
| Final Playtest | | Common | Minimum | Playtest Report Data | 5h |
| Playtest-based Modification | | Scripting | Minimum | Rebalanced game mechanics | 5h |
| Write Playtest Report | All | Common | Minimum | Playtest Report Document | 3h |
| Write Final Report | All | Common | Minimum | Final Report Document | 6h |
| Create Playtest Presentation | All | Common | Minimum | Playtest Report Presentation | 2h |
| Create main menu, options, about | | UI/UX | Desirable | A set of UI pages | 8h |
| Scout UI SFX | | Artwork | Desirable | A set of sounds for UI interaction | 3h |

| | | | | | |
|---|---|---|---|---|---|
| Improve UI assets | | Artwork | High | A set of textures for UI elements | 5h |
| Create enemy models | | Artwork | Extras | A set of models the enemies will be | 13h |
| Scout enemy sounds | | Artwork | Extras | A set of sounds for enemy attacks | 3h |
| Implement phase transitions | | Scripting | Extras | A system that smoothly transitions between running and fighting | 8h |
| Implement other input methods | | Scripting | Extras | Support for touch and gamepad | 8h |
| Form the Final Release | All | Common | | Final Executable, Documentation and Presentation | 16.02.2021 |

## 5. Assessment

Each team developing a video game sets out to create the magical “Flow” state, where the game’s difficulty and the players’ skill grow in just the right proportion. Our approach to solving this issue is to make our game easy to pick up for beginners but challenging enough to capture players from the very start and over longer periods of time.

It will be the kind of game that is easy to start playing, even when you just want to entertain yourself for five minutes waiting for the bus. On a technical level, that means startup and loading has to be quick. This requirement bleeds over into UX design as it has to be as quick as possible to start the initial round and also to restart after a fail state.

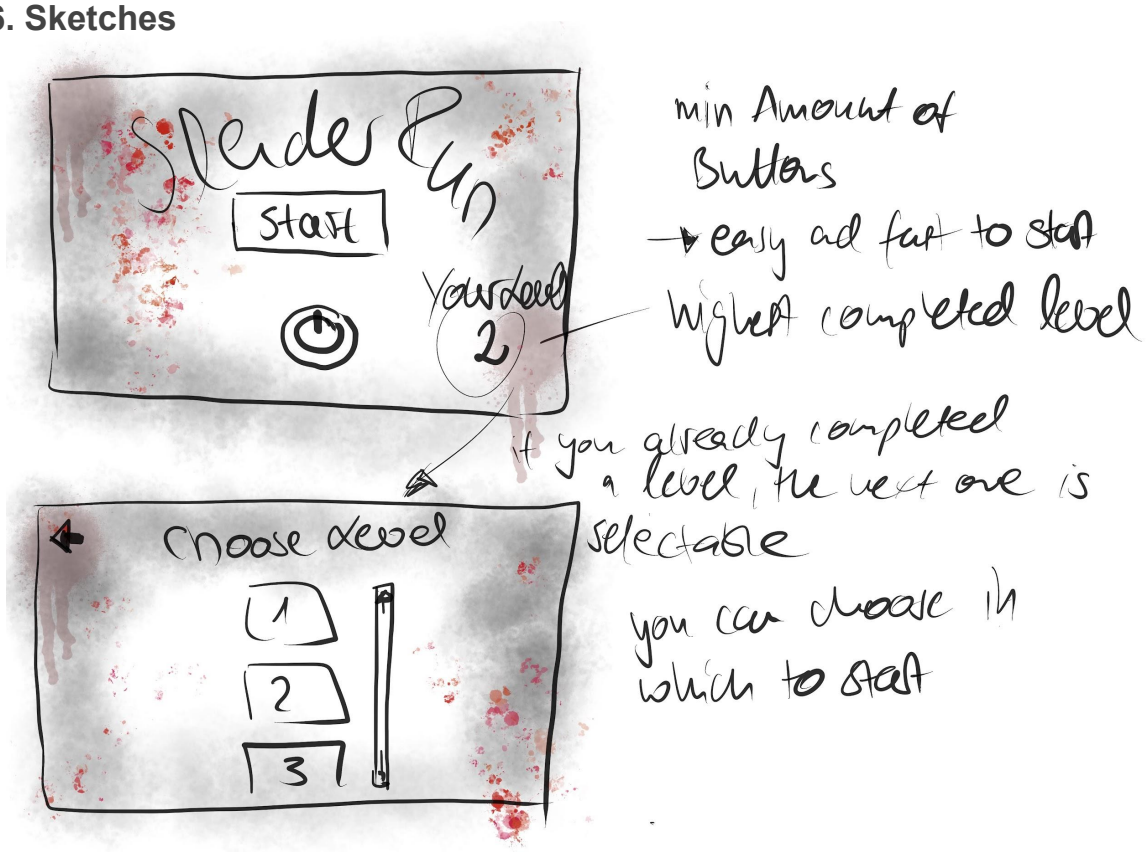
In terms of game design, initial complexity has to be kept low to minimize entry barriers. Unique inputs are few, intuitive and clearly communicated. The meaning and function of items and entities must be obvious without having to be learned beforehand.

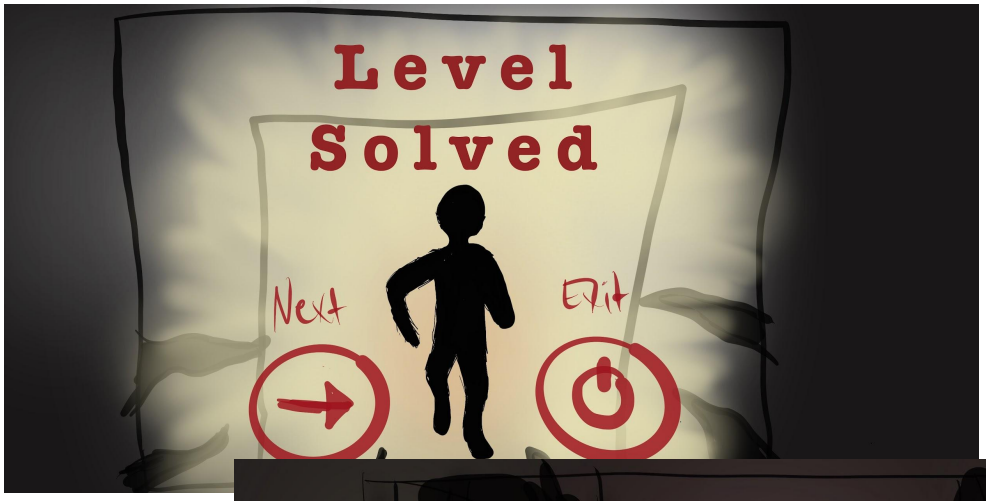
Even though the complexity will be low, the difficulty will be high from the onset. This is mainly achieved using a fast pace, forcing players to not just react but to make decisions on an instance. Mistakes quickly lead to being caught by the Slender Man. This high difficulty is intriguing, as players want to do better than in their previous runs. At the same time, frustration is avoided by having rounds not last very long and making it easy and most importantly quick to try again. Players don’t get too invested in a round when it lasts at most for a couple of minutes. Before the thought of stopping even crosses their mind, they have already pressed the retry button.

As soon as players learn to make the right moves on a moment by moment basis, a new level of complexity opens up. They will be able to plan their route through the dungeon in a way that maximizes rewards. Furthermore, remembering the run’s dungeon layout and backtracking will become more important for survival.

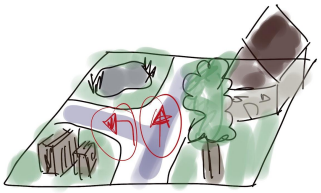
In summary, how should the game’s success be judged? First, the initial complexity must be low enough to be accessible even for people who do not have extensive experience with video games. Secondly, the game must be challenging to engage and motivate players while at the same time avoiding frustration. Thirdly, when players are mastering the basic mechanics, more complexity should be revealed to give players reasons to keep on playing in the long term.

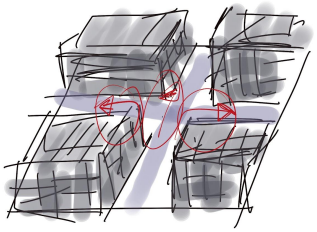
## 6. Sketches

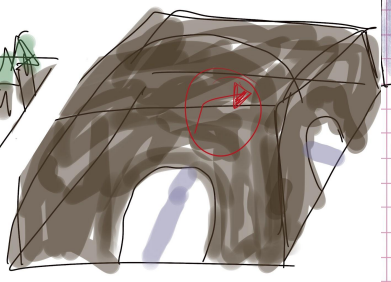

Start
Yourlevel
2
min Amount of Buttons
→ easy and fast to start
highest completed level
if you already completed a level, the next one is selectable
you can choose in which to start
Choose level
1
2
3

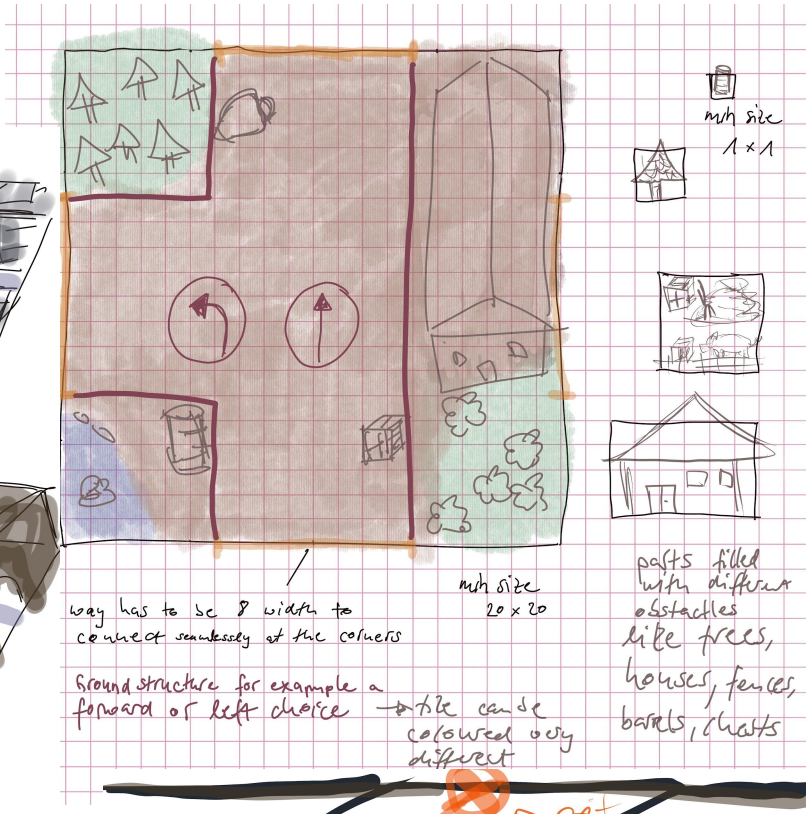

Level Solved
Next
Exit

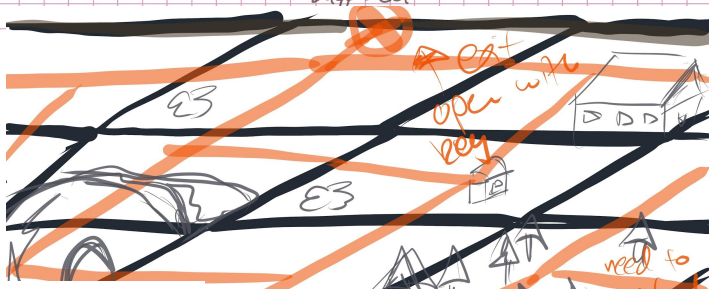

GAME OVER





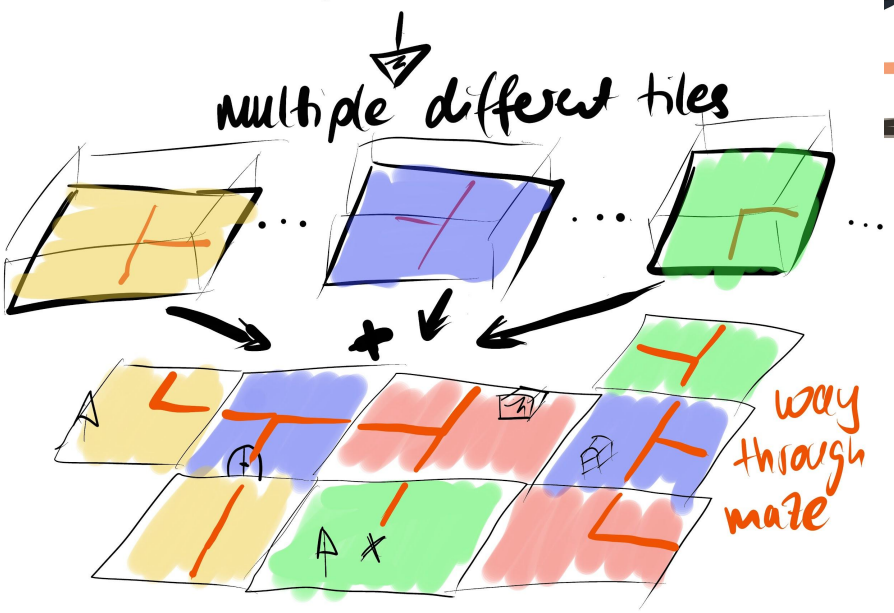


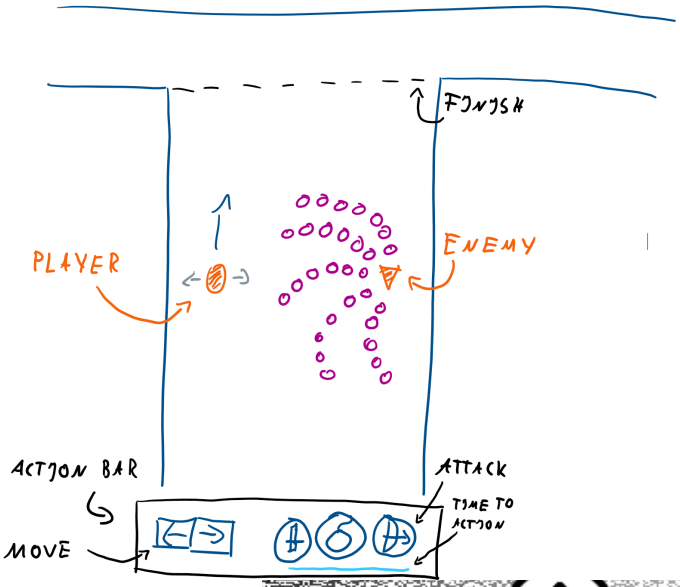

min size
1 x 1
way has to be 8 width to
connect seamlessly at the corners
ground structure for example a
forward or left choice
tile can be
coloured very
different
min size
20 x 20
parts filled
with different
obstacles
like trees,
houses, fences,
barrels, chests


exit
open way
key


need to
start
exit


ground
structure
quick event,
directions
obstacles,
environment
textures


multiple different tiles
way
through
maze


FINISH
PLAYER
ENEMY
ACTION BAR
ATTACK
TIME TO ACTION
MOVE