Bachelor project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

3D escape room game

Khanh Duong

Supervisor: Ing. Ladislav Čmolík, Ph.D.
January 2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Duong** Jméno: **Khanh** Osobní číslo: **492305**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Specializace: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

3D úniková hra

Název bakalářské práce anglicky:

3D escape room game

Pokyny pro vypracování:

Seznamte se s principy návrhu počítačových her. Proveďte analýzu herních principů používaných v 3D únikových počítačových hrách, kde je cílem uniknout z určité místnosti (typicky na základě vyřešení série hádanek). Dále se seznamte s modulárními komponentami a jejich využitím při návrhu úrovní. Na základě analýzy vytvořte design dokument pro 3D únikovou počítačovou hru. Dále vytvořte modulární komponenty, ze kterých se budou skládat jednotlivé úrovně, jejich materiály a textury. Dle design dokumentu vytvořte s využitím modulárních komponent alespoň tři hratelné úrovně hry. Každá úroveň se bude skládat z alespoň tří místností. Každá z úrovní bude zaměřena na jiné téma (např. matematika, fyzika, chemie). Každá místnost úrovně tedy bude obsahovat hádanky na téma úrovně. Výslednou hru otestujte pomocí kvalitativních testů alespoň s šesti hráči.

Seznam doporučené literatury:

1) R. Koster. Theory of Fun for Game Design, 2nd edition, O'Reilly Media, 2013.
2) J. Schell. The Art of Game Design: A book of lenses. CRC Press, 2008.
3) B. L. Mitchell. Game Design Essentials, John Wiley & Sons, 2012.
4) S. Rogers. Level up! the Guide to Great Video Game Design, John Wiley & Sons, 2014.
5) E. De Nucci and A. Kramarzewski. Practical Game Design: Learn the art of game design through applicable skills and cutting-edge insights, Packt Publishing, 2018.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Ladislav Čmolík, Ph.D. Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **16.02.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

Ing. Ladislav Čmolík, Ph.D.
podpis vedoucí(ho) práce

podpis vedoucí(ho) ústavu/katedry

prof. Mgr. Petr Páta, Ph.D.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.
Datum převzetí zadání

Podpis studenta

Acknowledgements

To begin with, I would like to thank Ing. Ladislav Čmolík, Ph.D., my supervisor, who patiently guided me and gave me a lot of insightful advice on how to improve the game, as well as discussing ideas on how to implement certain puzzles.

I would also like to show my appreciation to my family and friends for encouraging me. Without their support, this project would not come to fruition. Special thanks to those who helped test the game and gave valuable feedback, especially Dan Juříček and Dominik Dinh, who went out of their way to help me with unforeseen complications.

Declaration

I declare that this thesis is a work of my own and that all the sources used are cited and referenced. Prague, January 20, 2024

Abstract

This project focuses on analyzing the principles of game design, the process of making games, and the use of modular components. The game will be an escape room game with a theme focused on school subjects such as mathematics, physics, and chemistry with the intention of educating the player. The game will implement the ideas from the analysis and will be tested by the users to check if the ideas were implemented correctly.

Keywords: Unreal Engine, escape room, logic game, modular components

Supervisor: Ing. Ladislav Čmolík, Ph.D.

Abstrakt

Tento projekt se zaměřuje na analýzu principů herního designu, procesu vytváření her a využití modulárních komponentů. Hra bude úniková hra s téma zaměřené na školní předměty jako matematika, fyzika a chemie s cílem hráče edukovat. Hra bude implementovat návrhy z analýzy a bude otestována uživately, aby se zjistilo jestli byla implementace správná.

Klíčová slova: Unreal Engine, úníková místnost, logická hra, modulární komponenty

Překlad názvu: 3D úniková hra

Contents

1 Introduction 1

1.1 Structure of the thesis 2

2 Analysis 3

2.1 Escape rooms 3

2.2 Puzzle games 4

2.3 Educational games 4

2.3.1 Gamification 5

2.3.2 Game based learning 5

2.4 Game Concept 5

2.5 Game design 6

2.5.1 Level Design 7

2.5.2 Blockout 8

2.6 Prototype 8

2.7 Similar games 8

2.7.1 Escape Simulator 9

2.7.2 There Is No Game: Wrong Dimension 10

2.7.3 We Were Here 10

2.8 Modular components 11

2.9 Game engine 12

3 Design 15

3.1 Game Idea 15

3.2 Game Mechanics 16

3.3 Levels 17

3.3.1 Math Level 17

3.3.2 Chemistry Level 18

3.3.3 Physics Level 19

4 Technical analysis 21

4.1 Applications used 21

4.2 Unreal Engine 21

4.2.1 Blueprints 22

4.2.2 Custom Events 24

4.2.3 Interface 24

4.2.4 Material 24

4.2.5 Widget 24

4.2.6 Post Process Volume 25

4.3 3D Modeling software 25

5 Realization of the game 27

5.1 Prototype of the game 27

5.2 Use of modular components 28

5.3 Mechanics 29

5.4 Blueprints implementation 30

5.4.1 General Naming Conventions 30

5.4.2 Player Mechanics 31

5.4.3 Interaction 31

5.4.4 Highlighting interactable objects 32

5.4.5 Interactable 33

5.4.6 Inventory 33

5.4.7 Narrator 34

6 Results 35

7 Testing 41

7.0.1 Questions 42

7.0.2 Results 42

8 Conclusions 45

A Bibliography 47

Figures

2.1 Escape Simulator[27] 9

2.2 There is no game[24] 10

2.3 We were here[21] 11

2.4 Module made of modular components[26] 12

2.5 Visual look of a platform game in Godot[5] 13

4.1 An image of the UMG UI designer 25

4.2 Models of molecules in blender . 26

5.1 Creating images in Inkscape 27

5.2 Creating texture for modular wall in Quixel Mixer 28

5.3 Modular walls for the game levels created in Blender. (a) Math level, (b) chemistry level, and (c) physics level. 29

5.4 An item highlighted when interacted with 30

5.5 Using item outlines the item slot in yellow 30

5.6 Select item in BP_FirstPersonCharacter 31

5.7 Line trace in BP_FirstPersonCharacter 32

5.8 Interact with an object in BP_FirstPersonCharacter 32

5.9 Implemented interface BPI_Interact in BP_ItemMaster . 33

6.1 Screenshot from the math level . 36

6.2 Function puzzle in the math level 36

6.3 Chemical bond puzzle in chemistry level 37

6.4 Periodic table puzzle in chemistry level 37

6.5 Screenshot from the chemistry level 38

6.6 Example of a puzzle in the physics level 38

6.7 Screenshot from the physics level 39

6.8 Maze in physics level 39

7.1 Likert scale questionnaire 43

Tables

Chapter 1

Introduction

Nowadays, people seek entertainment more than anything with the evolution of technology, which comes in many forms. One such example would be games. The motivation for choosing a game as the topic of my thesis is that I find the process of game creation fascinating. I will focus on the game creation process and analyze similar games and principles of game design. The workload of creating a game is divided between multiple people with different roles. In the different roles, people develop the game mechanics, the music, the art, the models, and the direction of the game. As I will be working alone on this project, I will be mostly focusing on the game design and game mechanics, while procuring other assets, such as art, models, and music elsewhere, as it is impossible to cover all of the roles in game development alone.

This thesis will delve into the process of creating a 3D educational escape room game. I chose this specific genre because I have experience with escape rooms in real life and I would like to see it materialize in the form of a game and check if it is possible to retain the fun factor of escape rooms. Escape rooms also allow more freedom to the puzzles as they encompass a wide range of possibilities, ranging from riddles to simple mini-games as long as it is related to the theme of the room. As for the educational part, this was inspired by a certain level in a game called Escape Simulator, where there was a level that taught the player knowledge from different fields of subject.

1.1 Structure of the thesis

The following text is divided into five parts: The game creation process begins by forming the game concept, researching game design, and analyzing similar games, and will be explained further in Chapter 2. This chapter will also focus on game principles and puzzle games in general. The information obtained will be used to design my escape room game and will be expanded upon in Chapter 3. Chapter 4 is a technical analysis that describes the tools used and for what reasons with their upsides and downsides. In Chapter 5 is the realization of the game, where I will show how I created the game in Unreal Engine by presenting some key Blueprints, which are a coding asset used in Unreal Engine, and how the modular components, which are independent parts of a bigger system, were used. The last phase of the game creation process is to test the game and receive some feedback, which is included in Chapter 6. I will ask six testers for their opinion and to verify that there are no bugs inside the game to guarantee smooth gameplay. The results and look of the finished product are in Chapter 7. Chapter 8 will consist of my thoughts on the project, as well as the summary of the thesis.

Chapter 2

Analysis

Developing a good game constitutes of more than just an idea. In this chapter, I will focus on detailing the process of creating games as well as introducing game principles specifically for puzzle and educational games. In addition, I will be researching other similar games that have been successful to be able to take in some game elements that would improve my game.

2.1 Escape rooms

With the game being based on escape rooms, I will provide some basic information about it. The concept behind escape rooms is that players are trapped in a room full of puzzles and riddles. The design and content of the room depend on the theme of the escape room, so puzzles and clues will revolve around that topic. In addition to this, escape rooms are generally solved with a group of people, since puzzles are designed with multiple people in mind. The reason for this is that it encourages participants to communicate with each other and work together to achieve the objective. The goal of the escape room is to escape the room within a time limit by solving all the puzzles with clues and hints.

There are many types of escape rooms, ranging from rooms that have linear progression, where players can only progress after solving the previous puzzles, to rooms that allow you to freely explore and start with whatever puzzle they first found. The former is more friendly to the newcomers as they have a clear idea on how to progress, while the latter one is challenged by the more experienced group.

Although escape rooms are generally solved in groups, the games are a little different from real life. It is hard to transfer the feeling of having to tackle an escape room with a group in real life. Games should also be widely accessible, so by incorporating both multiplayer and single-player experience, the game will be flexible. The focus of this work will be only on the single-player aspect of the escape room experience due to time limitations and the reason mentioned above.

2.2 Puzzle games

Escape rooms are filled to the brim with puzzles, and while puzzles are part of many games, they can also exist as stand-alone games. Of course, some may not perceive puzzle games as real games, as they are mostly considered a tool that is used in games to enhance the gaming experience. In fact, this is true to some extent. For example, jigsaw puzzles and crossword puzzles are not recognized as games by most people, because the rules and the gameplay are so simple. Therefore, once the solution is found, the puzzle is not worth replaying, since the player already knows the answer.

Most games offer some kind of replayability, even story-based games, in which a player only plays through the game once. The key to success is to add dynamic elements, which prevents the game from getting stale. Some puzzle games achieve this by introducing an intelligent human as an opponent (chess, checker), which gives it some depth and challenge, giving the player a goal to reach (high score), and others do so by generating random challenges (Tetris, Solitaire). Games that implement a single strategy that always wins are called "Dominant strategy," and such games are considered flawed due to the fact that this is not a very exciting and fun design from the player's perspective[?].

2.3 Educational games

Having puzzles in the game introduces the possibility of educating the players in a fun and engaging way. The main purpose of educational games is to incentivize the audience to learn and retain information much more easily. However, games are designed to be enjoyable and entertaining, in contrast to learning, which is considered dull and tedious. Creating an immersive storyline, intriguing puzzles, and exciting challenges makes it possible to turn an educational game into something enjoyable. Striking a good balance between educational and enjoyable is still something companies are struggling with to this day. Currently, the two methods used to help teach in schools are gamification and game-based learning.

2.3.1 Gamification

Gamification is a process of turning learning into a game using game elements, such as points, badges, leaderboards, etc. It essentially turns a lesson into a simple game that encourages students to participate. It does so by taking advantage of the student's competitive spirit to engage in such activities.

2.3.2 Game based learning

Game-based learning, as opposed to gamification, uses games to teach students the content of the material. Many online sites support game-based learning, allowing teachers, parents, and students to use them to learn something new in an interactive way. One such example is the website Kahoot[7], which allows the teacher to create questions with multiple choices and the students compete with each other to get the most points by choosing the correct answer as fast as possible[**?**].

2.4 Game Concept

A game concept is an abstract document with the core ideas of the game. Everything about the game and what makes it enjoyable that you want to tell others should be in the game concept. It is just a cluster of various visions about the game. There is some essential information that is good to follow when making a game concept, and I will include some of them, which I believe are important and worth mentioning.

When creating a game, one of the things to consider is who the target audience is. Depending on the game, not everyone may enjoy it, so it is wise to target a specific group. It is plainly impossible to create a game that caters to everyone. For example, if such a game were to be developed, it would be time-consuming and eat up a lot of resources. For instance, it is crucial to take into account what type of game to produce, whether it should be a role-playing game, a strategy game, or an adventure game. The genre of the game affects the gameplay, as well as the target age group. For example, I believe that there is not a single horror game aimed at children under 15 years of age.

Second, it is fundamental that the purpose of the game is known to the player. The majority of people will not play a game in which they have no idea how to clear or progress through the game. The goal of the game is there as a catalyst that motivates the player to continue playing, as this enables the player to understand and enjoy the game. However, there are always exceptions to this notion. There exist games that do not follow this rule; however, they have a more niche audience.[23].

2.5 Game design

Game design is an integral part of making a game. Simply put, "Game design is the process of creating and shaping the mechanics, systems, and rules of a game"[4].

The basic guidelines for game design are much more intricate than meets the eye. There are many decisions, such as which direction the story should take, whether there should be a safe zone in the area full of monsters, and whether there should be a punishment for players who failed to clear challenges within a certain time frame. Every decision is part of a game design process and should not be made by one person. A game designer should be considered a role shared between those who are creating and influencing the game, not a person. Game design is not something that is written once and is considered complete; rather, it undergoes many revisions and updates. Many variations of the same Game Design can exist, as a lot of variables and changes are made depending on various factors such as hardware limitations, time restriction, or the ideas did not fit the current version of the game. In other words, game design is ever-changing and is always being modified.

These notions and key points should be kept and updated somewhere;

therefore, these are written and stored in the Game Design Document, which serves as an encyclopedia and a communication center between everyone working on the project. It is usually built upon and derived from a game concept. While working on the project, the GDD may undergo an overhaul in which the whole idea of the game is thrown away. When working in a team, updating the GDD is necessary as the project evolves and progresses.[**?**]

2.5.1 Level Design

Game design is all about what should be included in the game, while Level Design is dedicated to how everything should be implemented. It is important to take into account the Level Design when creating the game. For example, it should introduce the player to the game's mechanics at the start and progressively have more challenges while taking into account the environment, camera, and objects; thus, placing them correctly in the game is part of level design. It should take into account the player's experience with games in general and create a suitable learning curve for the target audience. For example, Dark Souls are notoriously known for their difficulty. Games that are hard are then compared to Souls games. Although Dark Souls is a difficult game to play, it also has a steep learning curve, making it difficult to learn. It throws a lot of things at the players at once. When creating a character, the game gives you a lot of classes to choose from without any detailed explanation, or when the game teaches you a lot of actions, most of which relate to combat, a few minutes apart from each other, it is a lot of information to take in. On the other hand, the learning curve is rather simple if we look at Hollow Knight, which is also known to be one of the games that are hard to beat. You have only three important actions: run, jump, and attack. The game then gives the player some time to familiarize themselves with the mechanics before reaching the first boss.[25]

Another brilliant way of level design will force the player to use the new mechanics in the level and make the player accustomed to them. Celeste[20] is an excellent example of a great-level design. The game is relatively simple as the character can only move, jump, climb, and dash. As the player progresses to later stages, more mechanics are introduced, such as bumpers, which players can use to bounce in another direction while keeping their velocity, and dash refills, which reset the character's dash count [3]. These mechanics are tied to a single map of the game to get the player accustomed to them, and each newly introduced mechanic can be found in subsequent maps. When a player gains another dash, the map accommodates it by increasing the difficulty, and to reach the objective, the player needs to use all of their dashes.

Finally, Level Design is not only about the placement of objects in the game but also how the game should balance progress such as how the game mechanics are introduced. Considering all that, there is no "right" way to design a level, since every game is different.

2.5.2 Blockout

Level design can be done in many ways, and one of them is by using something called blockout. A blockout is a way to create levels with simple temporary assets, such as cubes and cylinders. This eliminates the need to use complex objects to test various things, such as gameplay and functionality. Games such as FPS games place great importance on the positioning of objects, as this can greatly influence the balance of the game depending on which side the team is on. In this way, there is no need to create assets in the early stages of the game and make the level look organized. The main point of blockout is for level designers to be able to quickly create prototypes of the levels. [2].

2.6 Prototype

Everything mentioned in the game concept and game design sounds good on paper, but the idea may be different in the actual implementation. For that reason, a prototype exists, which could be described as a simple model to prove the design of the game.

The purpose of a prototype is to effectively allocate time and resources as they are limited. For example, the important things to test are whether a mechanic is engaging, if the implementation of the game mechanic is possible, or whether the user interface is intuitive or not.[22]

2.7 Similar games

Creating a good game requires a lot of effort. There are many things that sound good on paper but in practice work differently. Therefore, taking inspiration from existing games and analyzing them is really significant, as this helps us to see what worked in those games. I will look through similar games that are popular or familiar to me. They are all logic games with complex puzzles with a lot of variety. Some focus more on the gameplay, while others have an engaging story.

2.7.1 Escape Simulator

The Escape Simulator [27] is a 3D puzzle game centered on escaping the room within a time limit. The environment is interactive, as the player is able to examine various objects, smash pots, or throw objects around. The levels have specific themes, such as ancient Egypt or outer space. The game can be played solo or co-op with friends for an even better experience. Players can also create their custom levels in the editor available to them and publish the rooms they created for others to enjoy.

What I liked about this game is that it realizes the idea of escape rooms really well. You can do various things that you do in real life, such as crouching to see what is hidden underneath some objects, dropping things when your hands are full or when the things you are holding are not useful, picking up objects, and examining them thoroughly. This game may as well be a digital representation of a real escape room experience in my opinion, since the game allows the player to start with anything they want. In normal escape rooms, players are thrown into an escape room without a clue on where to begin and will have to investigate the rooms carefully for clues and objects that will help them escape. Although the game does not have a storyline, the interconnectivity of the rooms within the level creates an immersive experience.

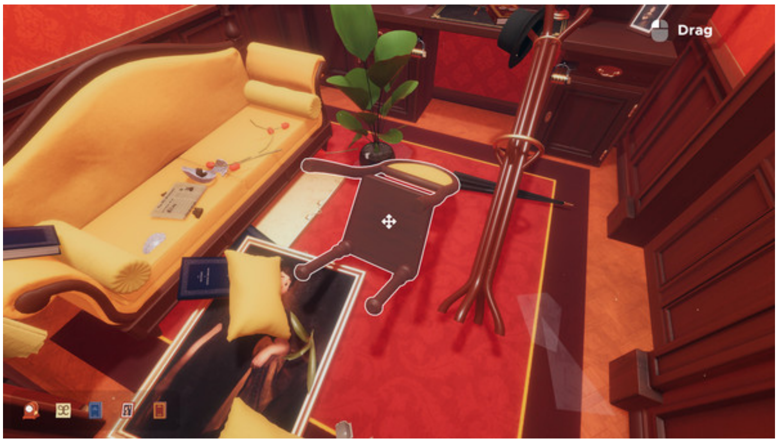


Figure 2.1: Escape Simulator[27]

2.7.2 There Is No Game: Wrong Dimension

There Is No Game: Wrong Dimension [24] is a continuation of a web game, There is No Game. It is a 2D puzzle game where the player solves puzzles with an unreliable narrator accompanying them while convincing the player that there is no game. A user, as the narrator refers to the player in the game, tries to play the game while the narrator tries to dissuade the user from playing by using any means necessary. The point of the game is that the player is breaking this so-called "non-game" in any way imaginable by playing the game. The game is very story-oriented, with puzzles that allow players to get creative when solving.

This game, while being a 2D game, caught my attention since the puzzles in this game are really unique and encourage the player to think outside the box. The solutions in this game seem nonsensical initially, but somehow they still make sense. The way the story is carried on by a narrator is really interesting, as the narrator comments on the actions of the player, so it feels the game is engaging. There is also a story with a clear objective, which is simply "play the game within the game."

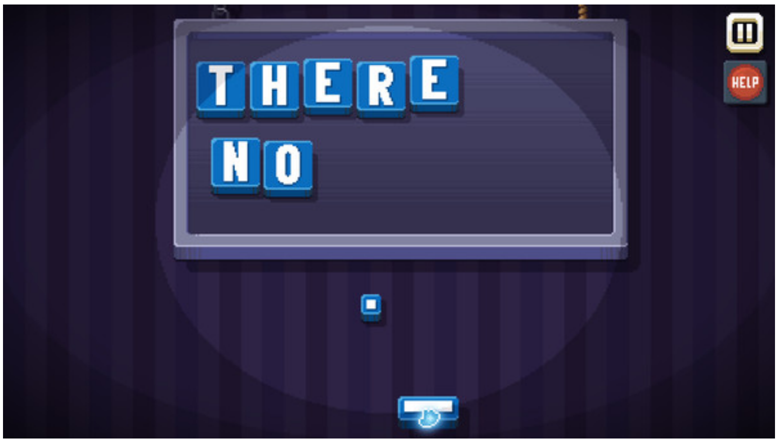


Figure 2.2: There is no game[24]

2.7.3 We Were Here

We were here [21] is a two-player game in which players need to cooperate to solve various puzzles. The players can communicate through a handheld receiver; however, only one person can talk at a time. This adds another layer of difficulty to the already challenging puzzles. The game uses mainly linear progression, which means that players must first solve the current puzzles to progress. The gameplay changes with the release of the sequels, where each game is about a different pair of characters trying to uncover the same truth. The game's sequels are We Were Here Too, We Were Here Together, and We Were Here Forever.

What really stood out to me was that they tackled the idea of puzzles in a different way. They separated the players into different rooms and the only way to solve the puzzles was to communicate the information with each other. The story is really well integrated into the puzzles, as well as the game itself.

Figure 2.3: We were here[21]

2.8 Modular components

As some games can get large, they need a lot of objects or models. One of the ways to go about this is by using modular components. Modular components are simple objects or models that can be reused to make something more extensive and complex. The parts are independent of each other, so hanging or removing the module will not affect other parts [22]. A definition that describes this well is: "Modular programming is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules such that each contains everything necessary to execute only one aspect of the desired functionality."[10] It is widely used in programming due to its advantages: reusability, ease of use, and maintenance[9]. While this is about modular programming, the same principles are used for modular components.

The use of modular components does not come without a cost, as reusing models will create a sense of monotony within the world. One way to avoid this is by adding more elements or unique models to the components. Some functional problems may also arise because components need to be more flexible. This problem is solved by expanding more features.[26]

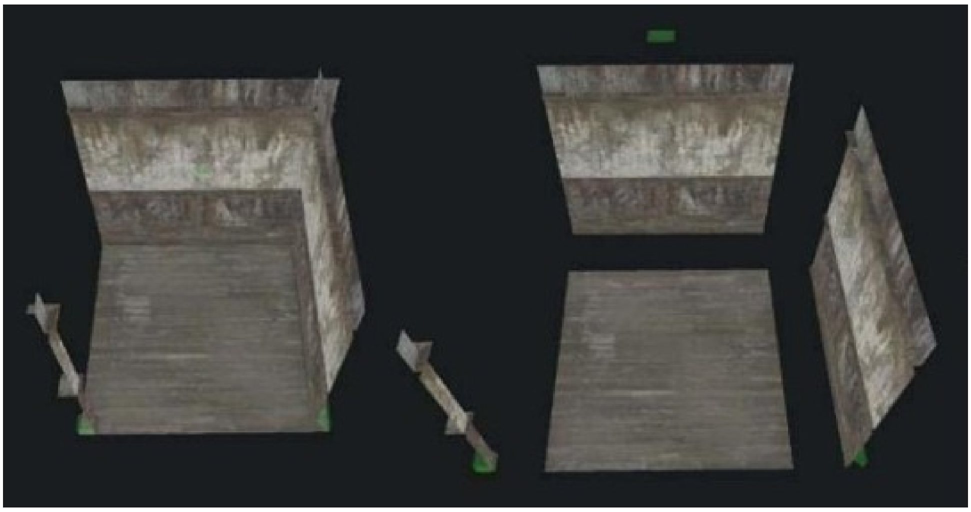
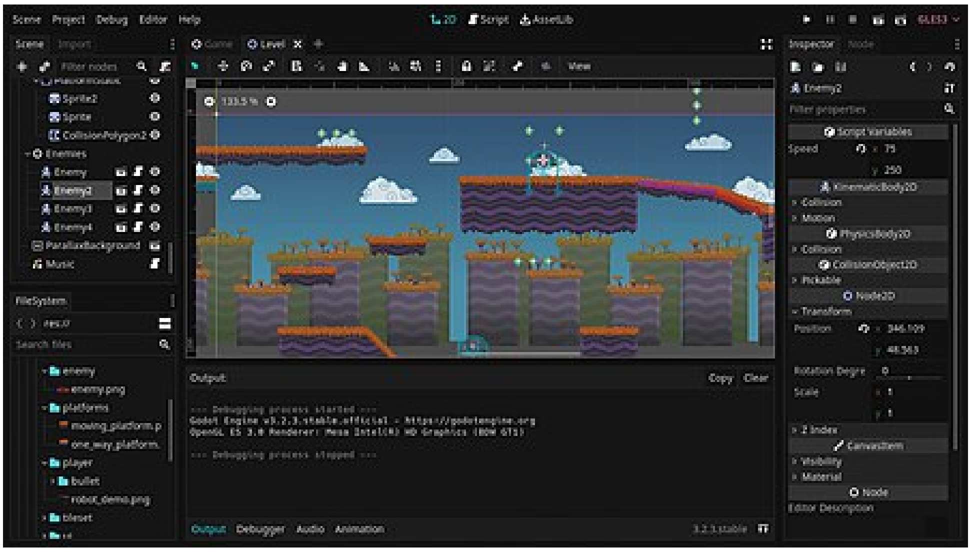


Figure 2.4: Module made of modular components[26]

2.9 Game engine

Creating a game from scratch is an arduous process. For this reason, game engines exist, which is basically a software framework designed primarily to support game development. It offers the necessary tools and features, such as collision detection, animations, and physics simulations. It also provides developers with a real-time preview, allowing them to see the changes they made right away. Furthermore, it also offers a visual environment, allowing even artists and designers to work on the project inside the engine. Thus, this makes them accommodating, and game developers are able to push out high-quality games at a fast pace [5]. As the specifics of each game are different, there is no perfect game engine with everything needed the way the game developers want it. That is why many companies have their own game engine, which they have created, or they modified existing game engines. An example of most known game engines at the moment are Unity[12] and Unreal Engine[13].



```
--- Debugging process started ---
Godot Engine v3.2.3.stable.official - https://godotengine.org
OpenGL ES 3.0 Renderer: Mesa Intel(R) HD Graphics (BDW GT1)

--- Debugging process stopped ---
```

Figure 2.5: Visual look of a platform game in Godot[5]

Chapter 3

Design

I will be designing my game around the analysis I made in Chapter 2, such as what type of puzzles to add, basic game mechanics, or how the direction of the story should evolve.

3.1 Game Idea

Since most escape rooms consistently have a theme and with my intention of making it educational, I have decided to create an escape room game with puzzles based on school subjects. The game will educate players about high school subjects in an enjoyable way. As learning itself is not fun, I will make it interactive and visually appealing. The target audience for this game is people with at least a high school education, as teaching new knowledge is challenging; however, reminding them of what they learned in high school should be more manageable.

Level design2.5.1 is an important part of game creation. I will divide the game into three levels, each with a different school subject. In this way, I can adjust the difficulty and assign a different school subject to each level. This will help the player feel satisfied since this is a clear indication of the player's progression for each level cleared.

The next thing to consider is how the levels should be designed. To avoid overwhelming the players with many rooms, each level will consist of a few compact rooms full of things with which they can interact to make the player disoriented. By doing this, the game will not feel monotonous, since the progression will not be so linear and the player will have to figure out where to start on their own. As many good games are accompanied by a story, there will be a comedic narrator who will mislead the player in the wrong direction in hopes that this will add an enjoyable experience for the players progressing through the game.

I took inspiration from the Escape Simulator where some levels taught you simple things, such as how the for-loop works in programming or how the resistors are color-coded. This would pose an interesting challenge to me, as I have to maintain the enjoyable aspect of the games while trying to impart useful knowledge to the player. As learning new things is difficult, instead I will help them recollect the knowledge they already learned from school. Therefore, school subjects will be the main theme of this escape room game.

As mentioned in Chapter 2 2.1, most escape rooms are designed for multiple players. Since this is a computer game and it is harder to transfer the experience of a group clearing the escape room together, and with other problems such as synchronization and scalability, I will instead focus on the single-player aspect.

3.2 Game Mechanics

With the basic idea of the game explained, it is time to introduce the game mechanics. As this game is based on a real-world escape room, actions such as crouch, drop items, pick items, and examine items are important. This is really well implemented in the Escape Room. Simulator2.7.1 so I will take my inspiration from this game. Having the ability to drop items, and therefore simulating a real-world escape room, having a simple inventory with a few slots is preferable in this case, since it pushes them to manage what they should hold on to and what things they should discard. Knowing what items a player can interact with is important so that they do not try and interact with objects that cannot be interacted with and waste their time. Interactable objects should ideally be highlighted or outlined. With the game having a narrator, I will also implement an event system that will trigger based on the player's actions, and the corresponding dialogue will trigger. Puzzles will be implemented in various ways: Some of them will use user interfaces, while others will be done by interacting with the environment.

To simulate the real-world experience, the player will not have a large inventory. There will only be a few slots in the inventory, so the player tries to manage their inventory by picking only the important items, while dropping unnecessary things or things not needed anymore.

3.3 Levels

Each level will be related to a school subject, which is mathematics, chemistry and physics. The puzzles will reflect that and will also be based on how the real world works, such as magnets having a north and south pole or gravity pulling us to the core of the Earth. While being educational, the game also needs to be enjoyable, so there will also be some illogical or peculiar things not related to the puzzle, such as a moving wall without any hidden mechanism. There will be a level progression where each level will be more complicated and different from the previous one. This will help the player to adapt to the puzzles.

The basic design of the rooms will be the same in all the rooms. Each room contains a few different puzzles that the players will need to solve by using hints in the form of books and notes. These hints are hidden in various places across the rooms, and it is up to the player to find them. If they have knowledge regarding the puzzles, then they may not need the hints provided. In each level, the narrator will accompany the player and provide entertainment through comedic dialogue.

3.3.1 Math Level

The mathematical-themed rooms will be the tutorial level. The level will progress linearly, meaning that the player has to solve the puzzles in the current room before moving on to the next one. The reason for doing this is to allow the player to explore the rooms entirely before moving on to the next ones to ensure that they understand where they should start. The explanation for making this room a tutorial level is that puzzles related to mathematical problems are simple and easily understood, such as functions and equations. This will also serve as an introduction to the upcoming puzzles, considering that they will be more complex in the following levels. In this level, there will be four mathematical problems in different rooms. The player will have to solve all of them to get to the next level. The first room will contain a function problem in the form of a code with a hint in a nearby drawer. To solve this, the player will have to find the value of a and b in the function $f(x) = ax + b$ and solve it for given x. This will give the player a six-digit number to input to open the door. In the same room, there will be a locked case in the closet that will need the digits of the third, fifth, and seventh numbers of the Fibonacci sequence to open. The clue to which part of the sequence and how the Fibonacci sequence looks like will be placed in the drawer of the second room. After opening the briefcase, the player will get a key to unlock the third room, where there will be four levers with inputs I and O. The last puzzle is a logic proposition in which the answers are ones and zeros corresponding to I and O on the levers. The questions are hidden in the previous room. When solving this, the player can advance to the next level.

3.3.2 Chemistry Level

The second level will provide more challenges by having all rooms available for players to explore right away. The rooms will be more furnished to hide the hints in inconspicuous places, forcing the player to search carefully around the place. This level will focus more on the education part by providing hints with detailed explanations of the chemistry puzzles. The level consists of three rooms and there are five puzzles in total. There is no particular order in which to solve them except for the pressure plate puzzle, which needs all the cubes obtained from previous puzzles. The main puzzle of this level is the aforementioned pressure plate puzzle, which is in the room in front of the player when they spawn. There will be a note that describes the weight needed to activate the pressure plates inside the room in which they spawn. The player has to find various cubes by exploring the rooms and by solving the puzzles then put them on the scale in the closet to find the correct combination of weight. There are two puzzles involving molecules in this level. The first puzzle is to match the colors and chemical bonds according to the chemical formula, while the other is to look at the colors of all molecule models scattered across the rooms, count the number of red, blue, and white atoms, and then input those numbers to open the red chest. Inside the chest there will be a briefcase with a periodic table puzzle. The solution is to find a diary of a child talking about bronze and to find a note showing the mixture of chemical elements to create various alloys, with bronze being one of them. The last puzzle is to put the correct number of reactants and products in the chemical equation to open the safe. After getting all the cubes and placing them on the pressure plates correctly, the player can move on to the next level.

3.3.3 Physics Level

This is the last designed level and should be the hardest. It introduces a different way to solve puzzles by having an interactive environment with which the players can play around, and unlike the previous levels, almost no hints are involved, and everything is left for the player to experiment with. At the beginning of this level, the player can only explore two rooms since the third room is locked. Similarly to the last level, there is a main puzzle, which is the stand holding up to three lenses and a light source on the table. The player has to find and combine the correct lenses to stop any light from passing through the sensor on the wall. Some of the lenses are locked behind puzzles. By exploring the rooms, the player will find some cogwheels, lenses, and a magnet. There are five puzzles in this level altogether. One of the puzzles is to find a note that shows the code only after putting it in front of the mirror. The code will be used to open a lock on the chest that contains a steel ball and one of the lenses. The next puzzle is to connect the pipes. After connecting the pipes, the player will receive the key to unlock the last room. This room will have a large maze that the player cannot enter and a large mirror on the ceiling. In this room, the player has to throw the steel ball into the maze and navigate the ball with the magnet and the mirror on the ceiling to the finish. After getting a cogwheel from this puzzle, the player can solve another puzzle that requires putting the cogwheels on the correct pegs and getting the last lens. After getting all the lenses, the player has to put the correct lenses on the stand and stop the light from reaching the sensor.

Chapter 4

Technical analysis

This Chapter will focus on the tools that were used to create the game assets and the game itself. This includes modeling software, art tools, and the game engine.

4.1 Applications used

The game is made in Unreal Engine version[13] 5.1. Walls and some other small models were created in Blender[1] 3.4.1. The materials for the models were found and made in Quixel Mixer. All images found in the game were made in Krita 5.1.5[8] and Inkscape 1.2.2[6].

4.2 Unreal Engine

Unreal Engine uses an event-driven programming system, which means that the flow of the program is determined by inputs and events, such as from the keyboard and mouse. Events can range from collision-based events, which trigger when objects collide against each other, to time-based events.

There are several other engines to choose from, which I briefly mentioned in Chapter 2 2.9. Each of the game engines has its strengths and weaknesses. While I do have experience with Unity, for this project I chose Unreal Engine, It is a game engine that allows high-quality 3D games to be made while allowing the usage of Blueprints, which is a visual scripting system. Blueprints are inherently limited in functionality and performance, so they are usually used along with C++ code. In this project, I will only use Blueprints, as the functionalities they provide are more than enough for my current game. This will allow me to improve my workflow, as creating Blueprint Classes and modifying their logic is faster than C++ code. I do not need any extra functionality extended beyond the use of Blueprints.

There are a few more reasons why I chose Unreal Engine over other engines, such as Unity. For example, a relatively new technology, called Nanite, was introduced, which is a virtualized geometry system that allows one to render models with a high polygon count without consuming a lot of memory space, as opposed to regular static meshes[18].

Another reason is that Unreal Engine is open source, which means the code is available for inspection or modification. This will allow me to understand and check how the code works or what parameters are needed for certain functions.

Next, I will be introducing some important concepts of Unreal Engine that I will be using in my game.

4.2.1 Blueprints

The Blueprint Visual Scripting System is a scripting system that uses a node-based interface to create gameplay. In Unreal Engine, everything is done through C++ code and Blueprints, so programmers can use both while developing their game, since Blueprints are just an extension of C++ code and are used mostly for convenience [14]. As mentioned above, I will only use Blueprints to create my game.

The two common types of Blueprints are the Level Blueprint and Blueprint Classes. The Level Blueprint is commonly used to reference actors in the level. It is mainly used for level-specific stuff, such as custom events, changing variables, and managing checkpoints. Using the Level Blueprint is generally not recommended for anything else, as the logic used will only exist in that level and thus cannot be easily reused somewhere else, and only one instance of the Level Blueprint can exist for each level. In contrast, Blueprint Classes are assets that can be reused later on, such as door, switch, and drawer logic. Although they do not have references to all other assets in the current level, as Level Blueprint does, you can manually assign references to other objects. These blueprints can be used simply by placing them in the level. Objects that implement Blueprint Classes are commonly referred to as Blueprints.

There are many various assets of Blueprint Classes, and here are some examples that I used in my game:

Actors

Actors are objects that can be placed in the scene in Unreal Engine. An example of an actor would be a camera or a mesh. They support 3D transformations, such as translation, rotation, and scaling. Creating a new actor is called spawning in Unreal Engine and is typically done through code or blueprints.

Pawns

Pawns are a subclass of Actors that can be controlled by either the player or AI. They have the same attributes as Actors, such as collisions and meshes, but are also possessed by a Controller. In my game, I will be using a Character class, which is a subclass of the Pawn with the ability to walk. [19]

Controller

Controllers are non-physical Actors that can possess a Pawn or their child classes to control its actions. They receive notifications for events such as on key press and on key release. Each Controller can typically only control one Pawn, but this can be modified for games that need a player to control many entities. A good example would be war strategy games.

4.2.2 Custom Events

Custom Events[15] are similar to normal Unreal Engine Events[16]. The difference is that Custom Events are created and called by the user. They fulfill a goal similar to that of functions, but at the core they are different. For example, Custom Events can use Delay Nodes and Timelines, which manipulate when the code should execute by using threads, and unlike functions, Custom Events do not have a return variable, so knowing which one to use depends on the situation.

4.2.3 Interface

Similarly to Custom Events, Interface classes in Unreal Engine allow different Blueprint classes to use common functions, even if they execute different events, for example, having different items, such as a book, a gun, and a potion, each doing something else. It is possible to implement the "Use Item" function, where when it is called, the player can read a book, shoot a gun, or drink a potion.

4.2.4 Material

Material is an asset that is used to apply texture and visual look to objects. The appearance of the surface is affected by parameters such as color, roughness, and transparency. Based on these parameters and other data, this is then used to compute how the light interacts with the surface of the Material.

4.2.5 Widget

The Widget Blueprint is an asset used to create user interface by using a visual UI editing tool Unreal Motion Graphics(UMG).

4.2.6 Post Process Volume

Post Process Volume is a volume that can be added to the level and allows us to change post processing parameters and add post process effects. Multiple instances of this volume can exist in a level, so that we can, for example, have special effects for indoor area and other separate effects for outdoor area.

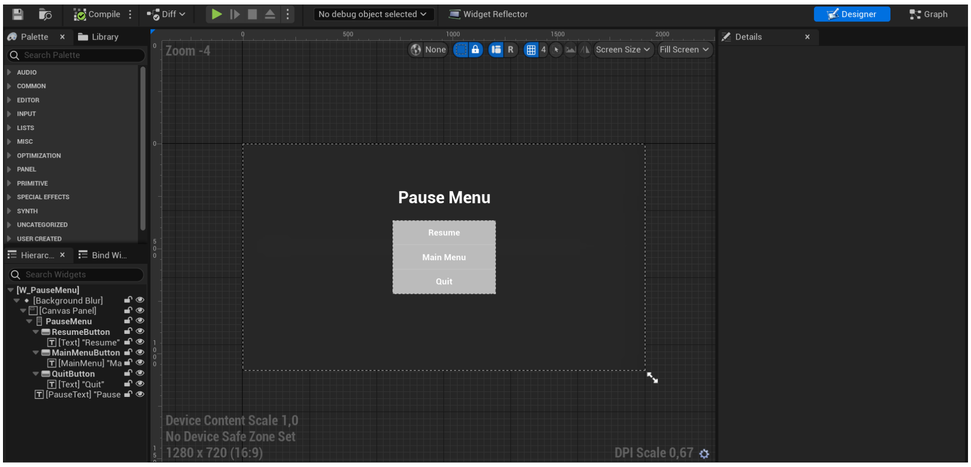


Figure 4.1: An image of the UMG UI designer

4.3 3D Modeling software

Blender was my primary choice to create models in. It is supported on various operating systems, such as Linux, Windows, and MacOS, which means that it is available to everyone and is also free.

I also have prior experience with it from courses taught at the university. Blender has many functions, such as allowing users to make models and 2D animations, and also allows for rendering or video editing. Thanks to the intuitive user interface, the program is relatively easy to use, even for beginners. It is open source, so it allows the user to import their own add-ons by using their Python API, which means that the user can create their own plugin through Python scripts to improve the functionalities of Blender.

For my game, I used the program to create simple models and basic materials that I then imported into Unreal Engine. Then, I assembled the models together to create a level; that includes some modular components and interactable objects. I did not model any of the furniture, as most of them were available online and of high quality free.

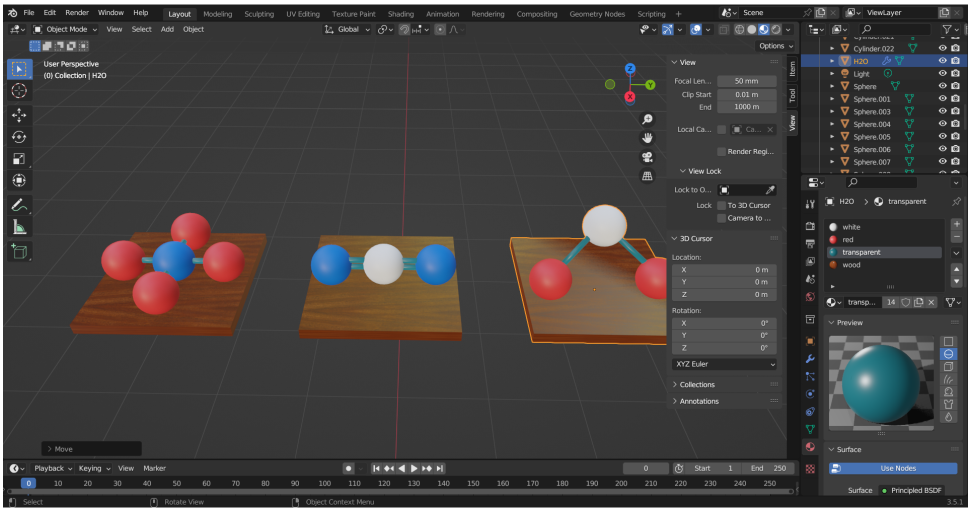
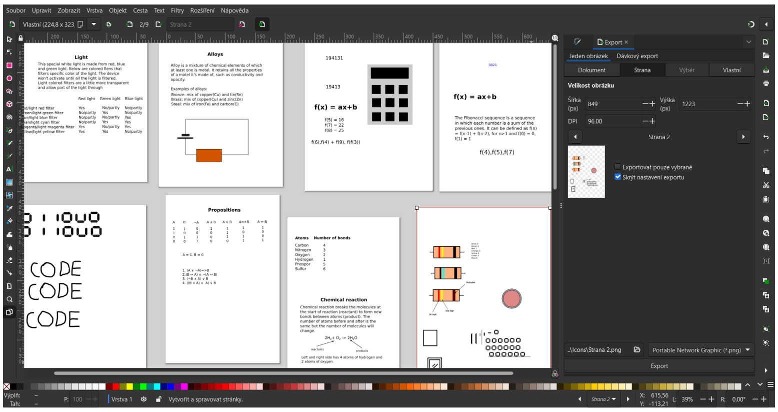


Figure 4.2: Models of molecules in blender

Chapter 5

Realization of the game

In this chapter, I will describe my workflow in creating modular components using Blender and Quixel Mixer[?] and show how I implemented the game design in Chapter 3 in Unreal Engine. Then, I will introduce a few Blueprint Classes that stand at the core of the game system and how some of the puzzles were made.



Light

This special white light is made from red, blue and green light. Below are colored flens that filters specific color of the light. The device won't activate until all the light is filtered. Light colored filters are a little more transparent and allow part of the light through

	Red light	Green light	Blue light
d/light red filter	Yes	No/partly	No/partly
een/light green filter	No/partly	Yes	No/partly
ue/light blue filter	No/partly	No/partly	Yes
an/light cyan filter	No/partly	Yes	Yes
agenta/light magenta filter	Yes	No/partly	Yes
llow/light yellow filter	Yes	Yes	No/partly

Alloys

Alloy is a mixture of chemical elements of which at least one is metal. It retains all the properties of a matel it's made of, such as conductivity and opacity.

Examples of alloys:
Bronze: mix of copper(Cu) and tin(Sn)
Brass: mix of copper(Cu) and zinc(Zn)
Steel: mix of iron(Fe) and carbon(C)

194131

19413

f(x) = ax+b

f(5) = 16
f(7) = 22
f(8) = 25

f(6),f(4) + f(9), f(f(3))

3821

f(x) = ax+b

The Fibonacci sequence is a sequence in which each number is a sum of the previous ones. It can be defined as f(n) = f(n-1) + f(n-2), for n>1 and f(0) = 0, f(1) = 1

f(4),f(5),f(7)

Propositions

A	B	¬A	A ∧ B	A ∨ B	A=>B	A ⇔ B
1	1	0	1	1	1	1
1	0	0	0	1	0	0
0	1	1	0	1	1	0
0	0	1	0	0	1	1

A = 1, B = 0

1. (A ∨ ¬A)=>B
2.(B ⇔ A) ∧ ¬(A ⇔ B)
3. (¬B ∧ A) ∨ B
4. ((B ∨ A) ∧ A) ∨ B

Atoms	Number of bonds
Carbon	4
Nitrogen	3
Oxygen	2
Hydrogen	1
Phospor	5
Sulfur	6

Chemical reaction

Chemical reaction breaks the molecules at the start of reaction (reactant) to form new bonds between atoms (product). The number of atoms before and after is the same but the number of molecules will change.

$2H_2 + O_2 \rightarrow 2H_2O$

reactants
products

Left and right side has 4 atoms of hydrogen and 2 atoms of oxygen.

Velikost obrázku
Šířka (px): 849
Výška (px): 1223
DPI: 96,00
Strana 2
☐ Exportovat pouze vybrané
☑ Skrýt nastavení exportu
..\Icons\Strana 2.png
Portable Network Graphic (*.png)

Výplň: –
Tah: –
P: 100
Vytvořit a spravovat stránky.
Strana 2
X: 615,56
Y: -113,21
L: 39%
R: 0,00°

Figure 5.1: Creating images in Inkscape

5.1 Prototype of the game

I made a prototype level where I tested some mechanics and functions. such as outline, Nanite and puzzles. The point of this is not to make a playable level, but to test things quickly and easily.

I mentioned using Nanite in my game, as it allows the rendering of high-quality models in real-time, with the downside of taking up a lot of memory. While creating an early prototype of my game, I ran into a few problems when using Nanite.

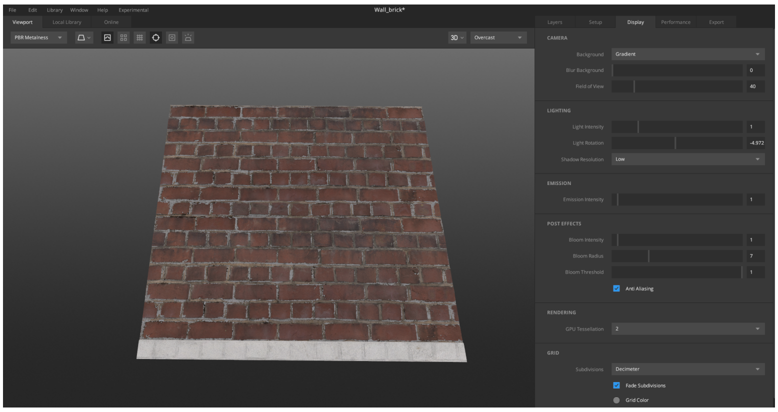
The first is that, since it is a relatively new technology, there is not much documentation on it. Therefore, figuring out what Nanite supports and how it can be used effectively required a lot of research.

The second reason is that it does not support some of the older functions that I need, such as highlighting an object and how Nanite meshes cannot be moved and have to be stationary. The downside of Nanite is too significant for it to be worth using over static meshes.

5.2 Use of modular components

As my game is an escape room simulation consisting of several rooms with similar appearance, the use of modular components is suitable for this situation. Modular walls were created in Blender, with the texture made and exported from Quixel Mixer to Blender to create materials.

In Figure 5.2 is a texture that I made in Quixel Mixer and in Figure 5.3 are the finished modular walls that I created in Blender for each level.



CAMERA

Background: Gradient
Blur Background: 0
Field of View: 40

LIGHTING

Light Intensity: 1
Light Rotation: -4.972
Shadow Resolution: Low

EMISSION

Emission Intensity: 1

POST EFFECTS

Bloom Intensity: 1
Bloom Radius: 7
Bloom Threshold: 1
☑ Anti Aliasing

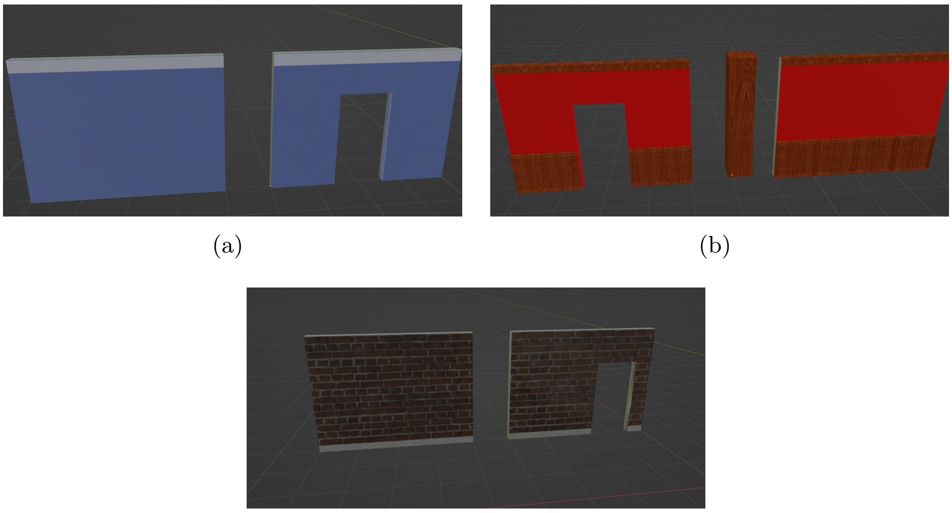
RENDERING

GPU Tessellation: 2

GRID

Subdivisions: Decimeter
☑ Fade Subdivisions
● Grid Color

Figure 5.2: Creating texture for modular wall in Quixel Mixer



(a)

(b)

(c)

Figure 5.3: Modular walls for the game levels created in Blender. (a) Math level, (b) chemistry level, and (c) physics level.

5.3 Mechanics

These are the basic mechanics and the core of the game. It is heavily inspired by possible actions that an individual can take in escape rooms and the game Escape Simulator.

- **Interaction:** enabling the player to interact with particular objects in the environment when the player looks at them. Interactable items will be highlighted. This includes picking up items, opening drawers, etc.
- **Selected item:** showing the player the currently selected item in the inventory. The selected item will be outlined in green color.
- **Use item:** allowing the player to use or examine the item in the inventory. The slot that contains the item will be outlined in yellow instead of green.
- **Drop item:** allowing the player to discard items that are not needed or helpful at the moment.
- **Crouch:** enabling the player to reach or see objects that would otherwise be difficult.

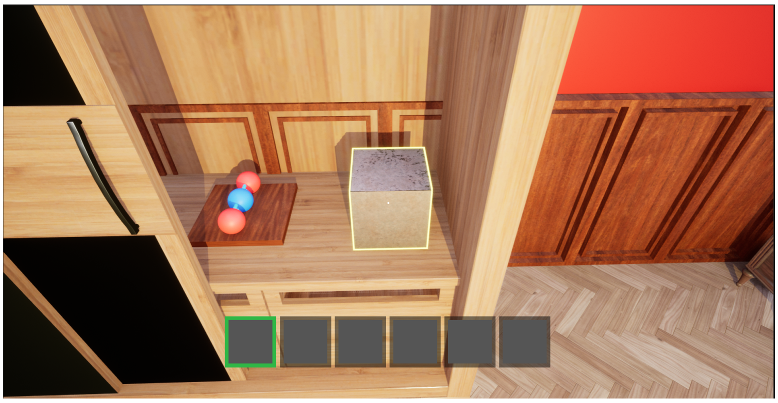
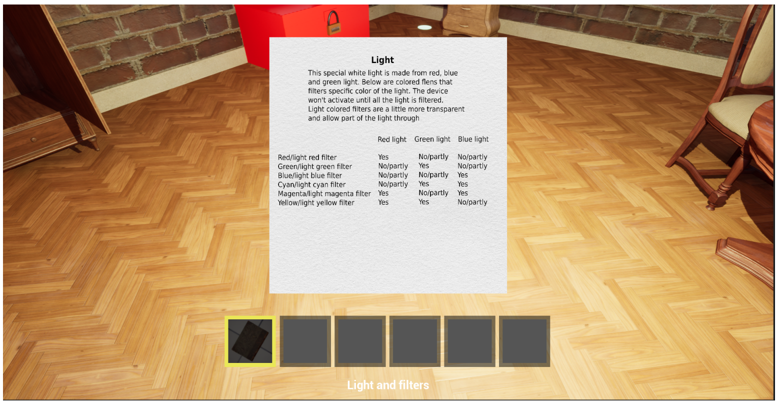


Figure 5.4: An item highlighted when interacted with



Light

This special white light is made from red, blue
and green light. Below are colored flens that
filters specific color of the light. The device
won't activate until all the light is filtered.
Light colored filters are a little more transparent
and allow part of the light through

	Red light	Green light	Blue light
Red/light red filter	Yes	No/partly	No/partly
Green/light green filter	No/partly	Yes	No/partly
Blue/light blue filter	No/partly	No/partly	Yes
Cyan/light cyan filter	No/partly	Yes	Yes
Magenta/light magenta filter	Yes	No/partly	Yes
Yellow/light yellow filter	Yes	Yes	No/partly

Figure 5.5: Using item outlines the item slot in yellow

5.4 Blueprints implementation

I will expand on all the important mechanics in the game and show the corresponding blueprints.

5.4.1 General Naming Conventions

In the following part, I will describe my code by using the names of various functions, Blueprints, Materials, and Widgets. So, to make the reading comprehensible, I will be providing some naming conventions that I used.

- **BP_:** prefix used for all Blueprint Classes.

- **M_:** prefix used for all Materials.
- **BPI_:** prefix used for all Blueprint Interfaces.
- **W_:** prefix used for all Widgets.

5.4.2 Player Mechanics

All of the player mechanics will be implemented in **BP_FirstPersonCharacter**. Basic mechanics, such as movement and jumping, were provided by a template from Unreal Engine. I built upon this Blueprint and added more functionalities, such as interaction, crouch, and the inventory system.

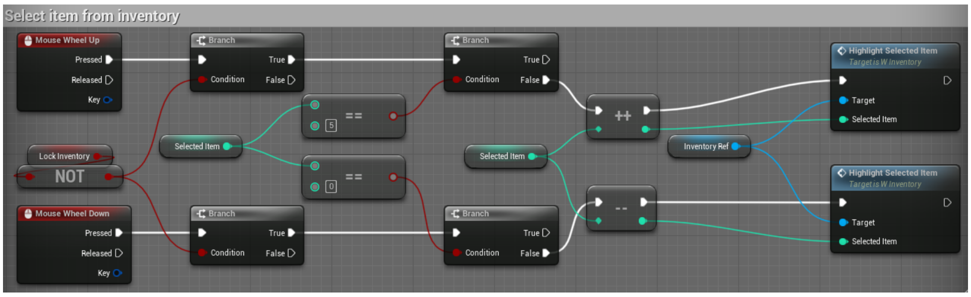


Figure 5.6: Select item in BP_FirstPersonCharacter

5.4.3 Interaction

The interaction is implemented by using a Line Trace with an event tick **Event tick** that is called every frame. This Line Trace registers everything in line of sight and returns Hit Result. This Hit Result is passed to a function called **Check Interactable**, which checks if the hit component it received contains the tag **Interactable**. This function verifies whether the object is valid for interaction. If it is, the function **Highlight Object** receives the object and sets its Custom Depth to true, which will allow the Custom Depth Buffer to outline the material. Finally, it saves the Hit Result of the Line Trace as a variable for the event **InputAction Interact**. This event is triggered when the player presses the E key and takes the Hit Result stored and calls an **Interact Function** in the blueprint interface **BPI_Interact**. For Blueprints that do not implement this, it will do nothing.

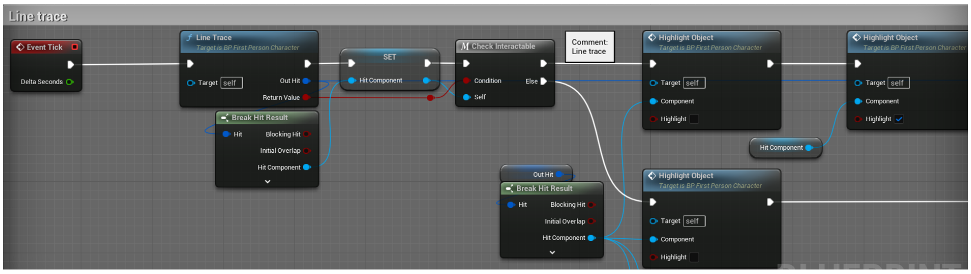


Figure 5.7: Line trace in BP_FirstPersonCharacter

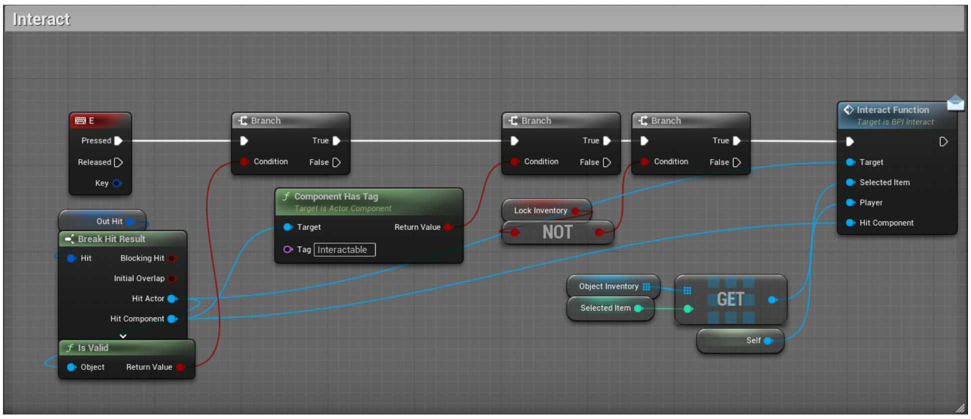


Figure 5.8: Interact with an object in BP_FirstPersonCharacter

5.4.4 Highlighting interactable objects

Highlighting objects was done by outlining the interactable objects in yellow. The outline was made by using an asset called Material in Unreal Engine and by using post-process effects that are enabled by adding Post Process volume to the level and setting the Material Domain of the Material to Post Process. With these settings, there was a need to tell the Material which part of the object to outline. To find out which part of the object to outline, we use shaders to find out. This was possible by checking the distance between the camera and the pixels of the object. The implemented outline is in Material called**M_YellowOutline**

After implementing this, the outline was shaky, and to get rid of it, I set the Blendable Location to Before Tonemapping, which stopped the shaking of the outline. This setting enabled the outline to not work properly when multiple translucent objects were next to each other.

5.4.5 Interactable

Interactable objects are divided into two categories: items and environmental objects. Both implement the interface **BPI_Interact**. The **Interact Function** will change something about environmental objects, be it movement, properties, or behavior. For items, it will remove them from the scene and add them to the inventory. The details of an item are defined in **ItemStruct** inside of **BP_ItemMaster**, where the **Interact Function** is implemented. Each item then inherits from **BP_ItemMaster**.

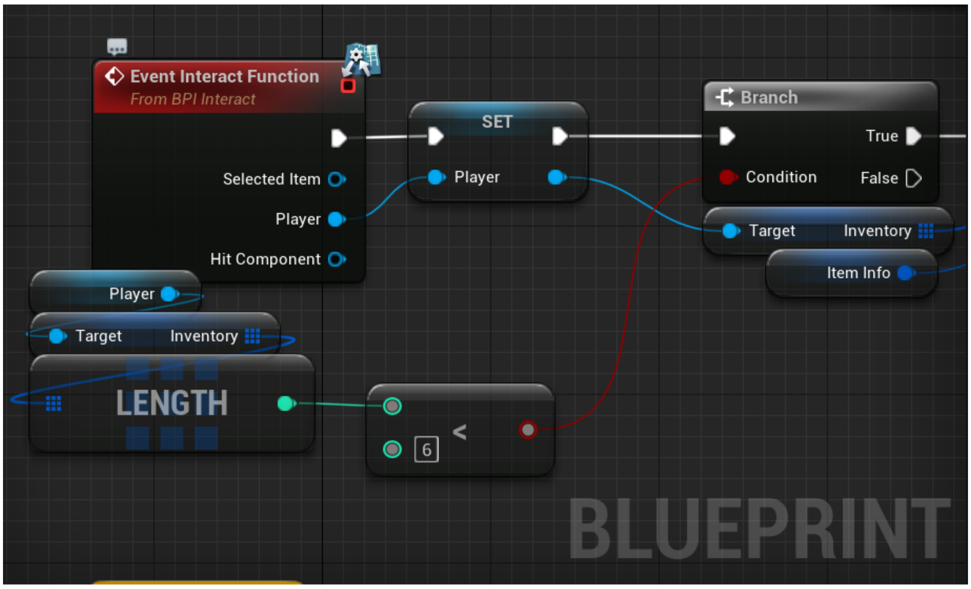


Figure 5.9: Implemented interface BPI_Interact in BP_ItemMaster

5.4.6 Inventory

Inventory is an array of structures called **ItemStruct**, and is then passed to the widget **W_Inventory** to display to the user. Using or dropping the items can be done by calling the event **Left Mouse Button** or **Right Mouse Button**, respectively. The Blueprint Interface can only be used on Actors, so it cannot be used for items in the inventory. The reason why I implemented a temporary array of objects containing the Actors removed from the scene is to be able to call **Use Function** from the interface **BPI_Use**.

5.4.7 Narrator

Each level contains an Actor in the level that inherits from **BP_DialogueSystem**. The **BP_DialogueSystem** sole job is to only display the dialogue and contains level-specific dialogue and **Call Event** function. This function is used by other Actors. The Actors in the level will hold a reference to **BP_DialogueSystem**, and whenever a certain event that requires a narrator is triggered, for example, solving a puzzle, then the object will call the function **Call Dialogue** and pass the dialogue text as an input. The **Call Event** which will trigger a dialogue.

Chapter 6

Results

The game consists of three rooms for each level. Each level implements at least three puzzles with varying difficulties and different types of puzzles. The game contains at least 15 simple models that I created, and the rest of the models were found in Unreal Marketplace[17], Quixel Bridge[11] and online from other sites. There is also a narrator who keeps the player company in every level of the game.

The following images are visual representations of the game. Figures 6.1 and 6.2 show the look of the math level of the game. Figures 6.3 to 6.5 show the puzzles and parts of the chemistry level. The appearance of the physics level and its puzzles are shown in Figures 6.6 to 6.8.

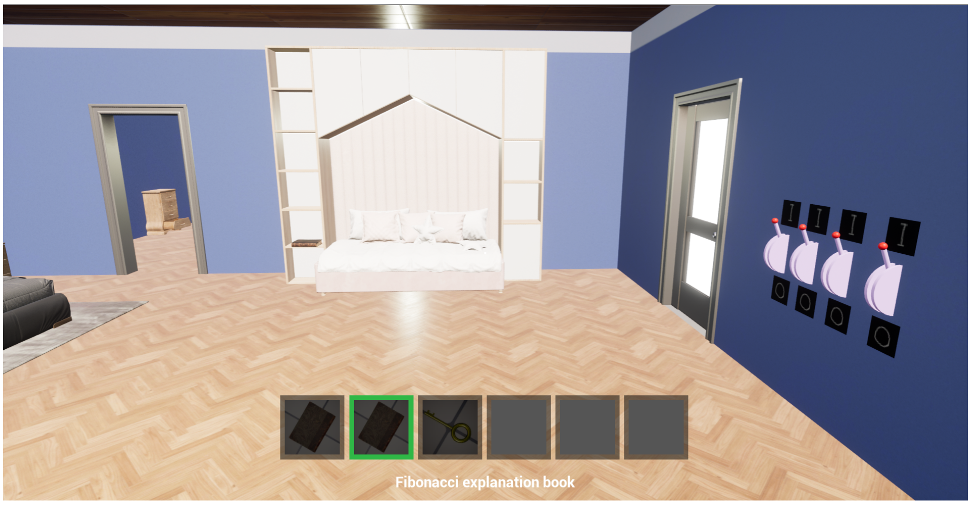


Figure 6.1: Screenshot from the math level

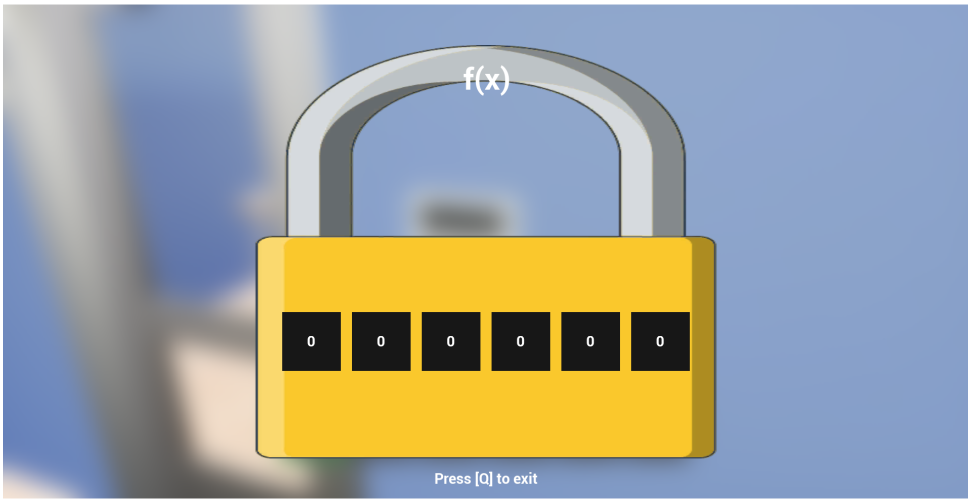


Figure 6.2: Function puzzle in the math level

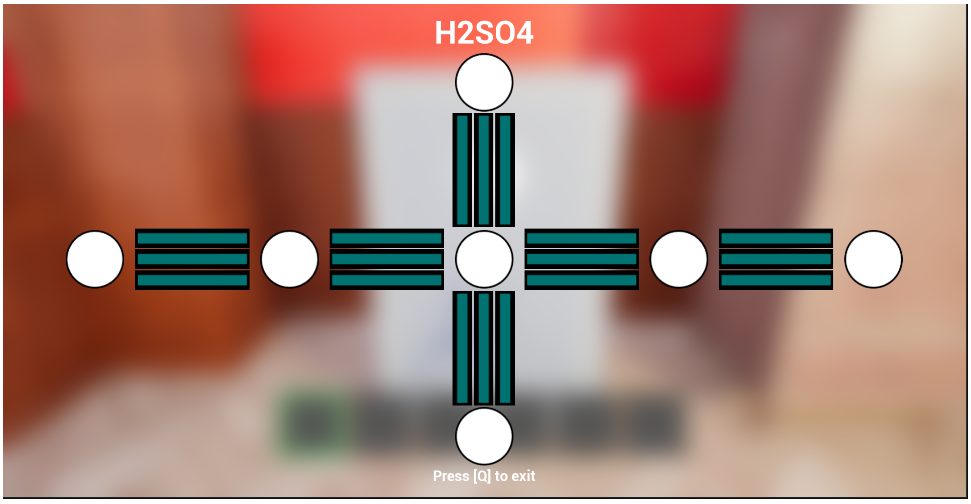


Figure 6.3: Chemical bond puzzle in chemistry level

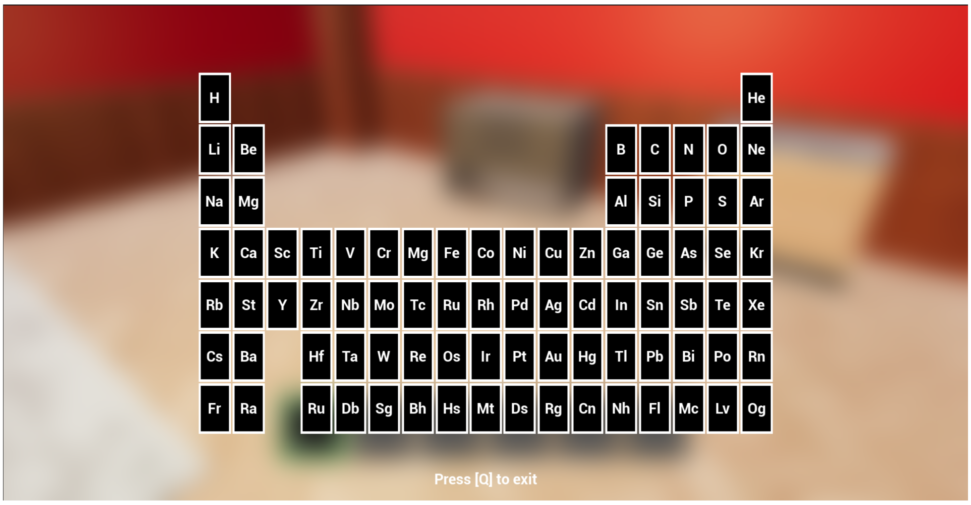

H He
Li Be B C N O Ne
Na Mg Al Si P S Ar
K Ca Sc Ti V Cr Mg Fe Co Ni Cu Zn Ga Ge As Se Kr
Rb St Y Zr Nb Mo Tc Ru Rh Pd Ag Cd In Sn Sb Te Xe
Cs Ba Hf Ta W Re Os Ir Pt Au Hg Tl Pb Bi Po Rn
Fr Ra Ru Db Sg Bh Hs Mt Ds Rg Cn Nh Fl Mc Lv Og
Press [Q] to exit

Figure 6.4: Periodic table puzzle in chemistry level

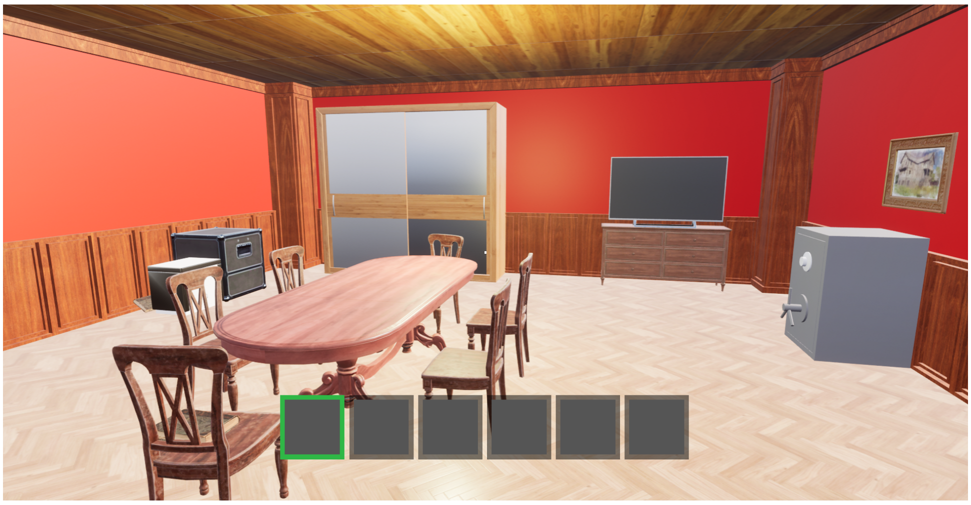


Figure 6.5: Screenshot from the chemistry level

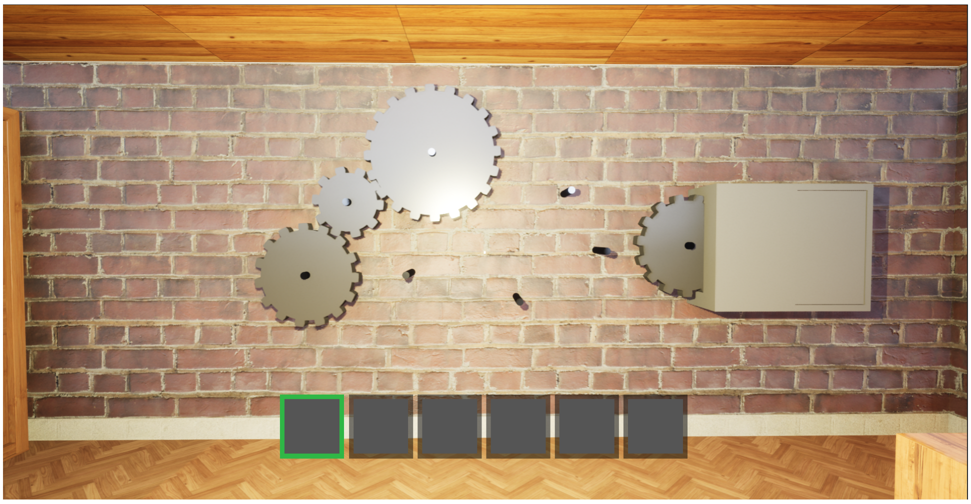


Figure 6.6: Example of a puzzle in the physics level

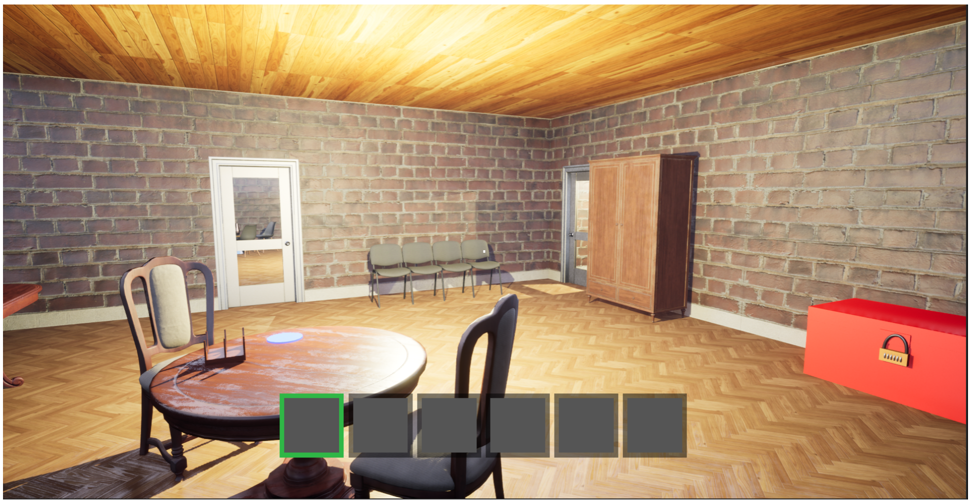


Figure 6.7: Screenshot from the physics level

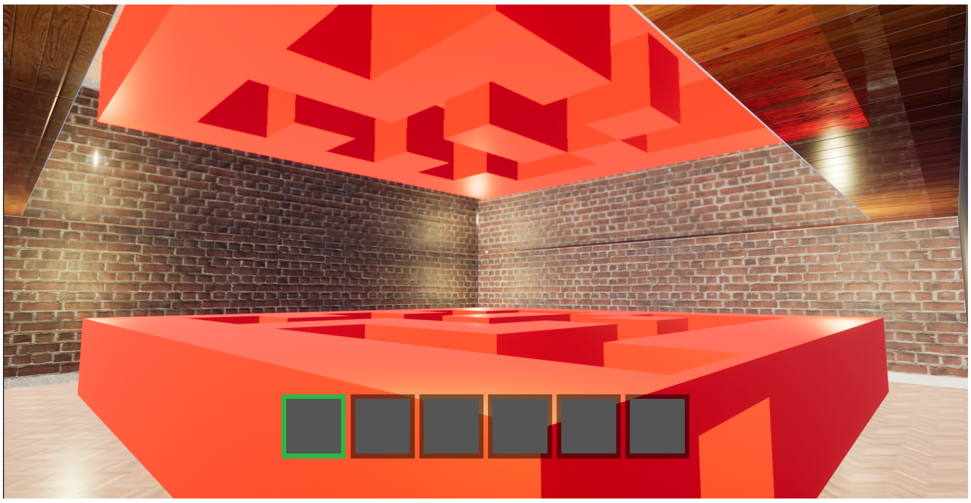


Figure 6.8: Maze in physics level

Chapter 7

Testing

Testing is one of the most essential steps in creating the game, since this may reveal various complications and bugs that would otherwise go unnoticed. When it comes to testing, unexpected things always occur in the hands of the user, as they do things differently from the developer. It also helps the developer understand whether they implemented the game correctly, depending on the user feedback.

The game was thoroughly tested with six people. When they were playing the game, I observed their playthrough and helped them progress if they got stuck. During testing, many testers encountered bugs and other problems. Some of the bugs were so awful that the player could not continue and had to restart the game.

An example of a bug the player encountered was when the player could move and interact with the environment while using a book, which displays a different UI. When the player interacts with a puzzle while having a book open, it would overlap another UI on top of the previous one, not allowing the game to register some inputs and with that preventing the player from doing anything and essentially locking them in suspended state. This is solved by not allowing the player to interact with any object in the environment while keeping their movement intact.

Other bugs include puzzles that do not work when the player tries to solve the puzzle in a certain way. For example, a pipe puzzle in the physics level would still be solved, even if the solution was wrong, because one specific position of the pipes did not update correctly.

7.0.1 Questions

I gave the testers a Likert scale questionnaire, which is a survey consisting of questions with commonly five or seven answer statements ranging from positive to negative, with the respondent choosing one of them. I have also asked them an open-ended question. Each user gave me valuable feedback and gave me insight into how I could improve the game in the future.

The questions in the questionnaire were as follows:

1. How satisfied were you with the number of rooms?
2. What do you think about the variety of puzzles?
3. How do you feel about the narrator in the game?
4. Did you feel like you learned new knowledge or were reminded of something you knew?
5. How difficult was the game for you?
6. Was the game fun for you?

As for the open-ended question, it was:

- Is there anything you want to add outside of the questions from questionnaire?

7.0.2 Results

From the information I have gathered when asking the testers questions, they were generally satisfied with the number of rooms in a level, and while they thought there was enough variety in puzzles, many of them did not consider them enjoyable, particularly the ones in the chemistry level, as the puzzles were too tedious. Many testers found the puzzles difficult, as they had to relearn something they already forgot to solve the puzzle. One of the testers mentioned that it would take them a lot longer without my guidance to clear

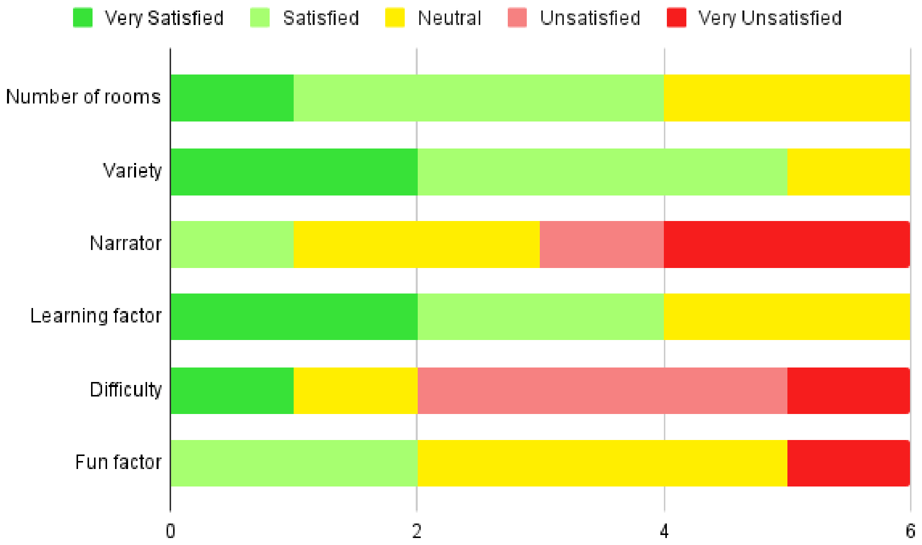


Figure 7.1: Likert scale questionnaire

the game. Some puzzles were hard enough that the testers had to use external tools, such as pen and paper. However, most of the testers were satisfied and enjoyed the physics level, as the puzzles were more intuitive compared to those in the previous levels. None of the testers paid any attention to the narrator, as he was not very noticeable with the dialogue fading into the screen and having no voice-over. The feedback I also got from one tester was that the puzzles were too reliant on books and that swapping between the puzzle and the book was annoying.

Overall, most of them liked the game, even though the game had its shortcomings, and this allowed me to consider a different approach to creating puzzles for the next levels.

Chapter 8

Conclusions

This thesis aimed to create a 3D escape room using modular components and test it with at least six players. I started by studying the principles of game design and analyzing similar logic games. Subsequently, I created a simple concept with various ideas of game mechanics that I wanted to implement. This served as a foundation for my game. The next step was to decide on the genre and type of game. Once I decided, I started experimenting with various types of puzzles, that fit the theme of the level in the escape room game, then verified if it was possible to implement them. With the logic behind the puzzles ready, I started to make models and with simple materials added, because part of the puzzles required them. The puzzles and clues were then placed with other furniture in the level. Once everything was ready, the last step was to let the others test the game for bugs and ask for their opinion about the implemented game.

My next goal is to add more variety of puzzles to the game as well as having a voiced narrator to improve the game. I will also focus more on puzzles, where the solution is more on the intuitive side than the ones relying on knowledge. I also plan to add a system that gives hints if a player gets stuck because of a puzzle and cannot progress.

Appendix A

Bibliography

[1] Blender. https://www.blender.org/, last accessed on 23/12/2023.

[2] Blockout. https://book.leveldesignbook.com/process/blockout, last accessed on 23/12/2023.

[3] Celeste mechanics. https://celeste.ink/wiki/Mechanics#Badeline, last accessed on 23/12/2023.

[4] Game design. https://en.wikipedia.org/wiki/Game_design, last accessed on 23/12/2023.

[5] Game engine. https://en.wikipedia.org/wiki/Game_engine, last accessed on 23/12/2023.

[6] Inkscape. https://inkscape.org/, last accessed on 23/12/2023.

[7] Kahoot. https://kahoot.com/, last accessed on 23/12/2023.

[8] Krita. https://krita.org/, last accessed on 23/12/2023.

[9] Modular approach in programming. https://www.geeksforgeeks.org/modular-approach-in-programming/, last accessed on 23/12/2023.

[10] Modular programming. https://en.wikipedia.org/wiki/Modular_programming/, last accessed on 23/12/2023.

[11] Quixel bridge. https://quixel.com/bridge, last accessed on 23/12/2023.

[12] Unity. https://unity.com/, last accessed on 23/12/2023.

[13] Unreal engine. https://www.unrealengine.com/en-US, last accessed on 23/12/2023.

[14] Unreal engine blueprints. https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/Blueprints/GettingStarted/, last accessed on 23/12/2023.

[15] Unreal engine custom events. https://docs.unrealengine.com/4.26/en-US/ProgrammingAndScripting/Blueprints/UserGuide/Events/Custom/, last accessed on 23/12/2023.

[16] Unreal engine events. https://docs.unrealengine.com/4.26/en-US/ProgrammingAndScripting/Blueprints/UserGuide/Events/, last accessed on 23/12/2023.

[17] Unreal engine marketplace. https://www.unrealengine.com/marketplace/en-US/store, last accessed on 23/12/2023.

[18] Unreal engine nanite. https://docs.unrealengine.com/5.0/en-US/nanite-virtualized-geometry-in-unreal-engine/, last accessed on 23/12/2023.

[19] Unreal engine pawns. https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/Framework/Pawn/, last accessed on 23/12/2023.

[20] Extremely OK Games, Ltd. Celeste. https://store.steampowered.com/app/504230/Celeste/, last accessed on 23/12/2023.

[21] Total Mayhem Games. We were here. https://store.steampowered.com/app/582500/We_Were_Here/, last accessed on 23/12/2023.

[22] Adam Kramarzewski and Ennio De Nucci. *Practical Game Design : Learn the Art of Game Design Through Applicable Skills and Cutting-Edge Insights*. Packt Publishing, Limited, 2018.

[23] Briar Lee Mitchell. *Game Design Essentials*. John Wiley & Sons, Incorporated, 2012.

[24] Draw Me A Pixel. There is no game. https://store.steampowered.com/app/1240210/There_Is_No_Game_Wrong_Dimension/, last accessed on 23/12/2023.

[25] Samantha Stahlke and Pejman Mirza-Babaei. *The Game Designer's Playbook : An Introduction to Game Interaction Design*. Oxford University Press, Incorporated, 2022.

[26] Nataska Statham, João Jacob, and Mikael Fridenfalk. Game environment art with modular architecture. *Entertainment Computing*, 41:100476, 2022. , last accessed on 23/12/2023.

[27] Pine Studio. Escape simulator. https://store.steampowered.com/app/1435790/Escape_Simulator/, last accessed on 23/12/2023.