

# TABLE OF CONTENTS

|   |     |
|---|-----|
| DECLARATION .....                           | i   |
| ACKNOWLEDGEMENT .....                       | ii  |
| TABLE OF CONTENTS.....                      | iii |
| LIST OF FIGURES .....                       | iv  |
| ABSTRACT.....                               | v   |
| Chapter 01 - INTRODUCTION .....             | 1   |
| 1. INTRODUCTION TO GAMES .....              | 1   |
| 1.1 A brief history of computer games ..... | 2   |
| 1.2 2D games and 3D games.....              | 2   |
| Chapter 02 – WORKING METHODOLOGY .....      | 4   |
| 2. WORKING PROCESS .....                    | 4   |
| 2.1 Ideation .....                          | 4   |
| 2.2 Story And Narrative Development .....   | 5   |
| 2.4 Player-Testing And Prototyping .....    | 5   |
| Chapter 03- SYSTEM DESIGN .....             | 7   |
| 3. SYSTEM DESIGNING .....                   | 7   |
| 3.1 About Platform.....                     | 7   |
| 3.2 Programming Language.....               | 7   |
| 3.3 Actor Designing .....                   | 8   |
| 3.4 Level Designing .....                   | 9   |
| 3.5 User Interface(UI) .....                | 10  |
| Chapter 04 - RESULT AND CONCLUSION .....    | 12  |
| 4. PROJECT RESULT & CONCLUSION.....         | 12  |
| 4.1 Project Result.....                     | 12  |
| 4.2 Project Conclusion .....                | 13  |
| REFERENCES .....                            | 14  |

## LIST OF FIGURES

|   |    |
|---|----|
| Figure 1: Basic Idea .....                      | 4  |
| Figure 2: Player Movements .....                | 5  |
| Figure 3: Player Testing Phase Screenshot ..... | 6  |
| Figure 4: Godot Icon .....                      | 7  |
| Figure 5 : Player Designing Process .....       | 8  |
| Figure 6: Enemy Designing Process .....         | 9  |
| Figure 7: Tilemaping in Level Designing .....   | 9  |
| Figure 8: Level 2 Design.....                   | 10 |
| Figure 9: MainScreen.....                       | 10 |
| Figure 10: AudioPlayerBlock .....               | 11 |

## **ABSTRACT**

The creation of video games involves multidisciplinary processes that are not accessible to the general public. Currently, video game development environments are very powerful tools, but they also require an advanced technical level to even start using them. This article presents a 2D game development environment to propose an alternative model to reduce the technical complexity existing in these systems, presenting a data model and a game editor that allows fulfilling this goal. In order to test its capabilities, several games have been successfully implemented in the proposed environment. With this achievement, it can be stated that it is possible to create video games simply and adorably for the general public without giving up its potential and remarking that there is still a long way to go to reach democratization in the creation of video games and the need to continue working in this field.

In this project, we are building a 2D game using modern assets rather than old assets. We are using GODOT Game Engine as our workspace. In this workspace, we have all the essential provided tools to build and implement our game theme. We created this as a single-player game that only has 3 axes to move around and has different levels to complete and in all these levels we have random numbers of coins to collect by the player. It gives a sense of accomplishment to the player which will encourage them to move forward to the next level.

Our player will move forward to the end of the level with the maximum coins it can gain in the level and at the end, it will collide with the teleporter which will teleport it to the next level. After they complete all the levels we give them a congratulations message and the option to play again.

# **Chapter 01 - INTRODUCTION**

Gaming is becoming more and more popular over the decades and the game business is booming. People play games to relax, to be better at competitive games or simply just to kill time. Therefore, the game industry has become one of the most sizeable (and interesting) businesses in the world. New games are introduced to players every day by indie game companies and corporations, and news about game release dates pop up in every social media and advertisements. As a result, more programmers and developers are taking an interest in game development and design, but not everyone knows where to start as the skills needed are vast and varied. Most find game development intimidating because of the number of skills and knowledge needed to make a game: programming, animation, sound design, environment art... this is the fundamental knowledge to make a game. Not everyone has all these skills on their hands, and that is why triple-A games are developed by big teams (or game studios) such as Ubisoft, Naughty Dog, Square Enix, CD Projekt Red... but that does not mean an individual cannot make good games by themselves. There are many indie game developers that create astonishing games like Minecraft by Markus “Notch” Persson, Stardew Valley by Eric Barone, Undertale by Toby Fox, ... Thus, in my thesis project, I want to help newcomers to game development start their first game as I guide them step by step. I will introduce the basic skills of game development with the programming language C# and one of the most widely used game engine out there: Unity. Since this is an introduction to game development, I only cover a 2D game as it is the basic. My goal is that by the end of my thesis, the programmers and developers get a fundamental knowledge on game development and can start to make games independently. Through my thesis, I introduce thoroughly each of the components I mention above (C#, Unity, 2D games...). During the introduction of each component, I give examples so that readers can easily understand and relate since not every newcomer is acquainted with the words or expressions.

## **1. INTRODUCTION TO GAMES**

Computer games, are electronic games that involve interaction with a user interface or input device – such as a joystick, controller, keyboard, or motion sensing device – to generate visual feedback. This feedback mostly commonly is shown on a video display device, such as a TV set, monitor, touchscreen, or virtual reality headset. Some computer games do not always depend on a graphics display, for example text adventure games and computer chess can be played through teletype printers. Video games are defined based on their platform, which

include arcade video games, console games, and personal computer (PC) games. More recently, the industry has expanded onto mobile gaming through smartphones and tablet computers, virtual and augmented reality systems, and remote cloud gaming. Video games are classified into a wide range of genres based on their type of gameplay and purpose..

### **1.1 A brief history of computer games**

Alan Turing – the one that inspired modern computer science and artificial intelligence – was the first person to make a computer game, a computer Chess program in 1947. It was not his purpose to make it playable for people but to test out artificial intelligence. It was not until 1952 that he tested the program with a colleague, and he pretended to be the computer (Tristan Donovan, 2010). Turing's first game inspired so many mathematicians and computer scientists back in those days that they decided to continue his work. Later in 1972, Pong was released into the arcade. It became so popular back then that the arcade machine was jammed because it had so many arcade coins in it. July 1980, a missing slice of pizza inspired Toru Iwatani to create Pac-Man, the first arcade game hit that people can play at home with their console. Fast-forward to early 21st century, we have news channels about games, streaming services for gamers, online game stores, dedicated merchandises based on popular games, with many companies dedicated to making games like Electronic Arts, Ubisoft, CD Project Red... With all that listed above, it is without a doubt that game industry has become one of the fastest growing and most profitable industries in the world.

### **1.2 2D games and 3D games**

The first generation of computer games was 2-dimensional (2D) since technology back then was limited for 3-dimensional (3D) simulation. But nowadays, the growth of technology allows us to make 3D simulations with the right tools. 3D printing is now possible, 3D simulations allows us to experience things that are normally out of our reach or too risky to try (learn to drive, learn to fly a plane, simulate the solar system,...). So, what are 2D and 3D games? Figure 1. 2D and 3D comparison. In 2D games, the world inside it is only two-dimensional, which means that it only consists of 2 axes: X and Y. 2D games use flat graphics, called sprites, and don't have three-dimensional geometry. They're drawn to the screen as flat images, and the camera (orthographic camera) has no perspective (Unity, 2021). normally in 2D games, the player can only move left and right, up and down, according to the X-Y axis since the environment in 2D games is visually flat. Some of the popular 2D games examples: Super Mario Bros, Pac-Man, Donkey Kong, ... The limitation on movement however does not mean

that 2D games are less enjoyable than 3D games, this depends on the players' preference. Moreover, for players who prefer casual gaming and lightweight games, 2D games is the more suitable choice because 3D games nowadays are often more demanding and take more space on the hard drive. 8 3D games, however, offer a more immersive experience. 3D games usually make use of three-dimensional geometry, with Materials and Textures rendered on the surface of GameObjects to make them appear as solid environments, characters and objects that make up the game world (Unity 2021). In a 3D game, the player can move and see its world in 3D, which means it is a simulation of the real world. Objects in a 3D environment have depth and volume because the game world operates on an X-Y-Z axis. Players can move up and down, left and right, forward and backward. 3D games have more possibilities and goals than 2D games. In 2D games, the character only moves in two dimensions, which is easy to control. While 3D games make it more challenging for the players because they have to get used to the 3D movement of the game, for example WASD keys to move, space to jump and mouse to look around. The reason why I want to write for 2D game development only is because it is the most basic and easy to learn. After learning to make 2D games, readers have the knowledge and foundation in making games so that it is an easy transition to 3D game development should they choose to.

## Chapter 02 – WORKING METHODOLOGY

Development methodology refers to a series of techniques and/or processes by which a video game is developed. While it is possible to develop a video game by following various general software methodologies (e.g. the waterfall model, the incremental or the agile method, etc.), game development generally consists of three phases: pre-production, production and post-production based on the film's life cycle

### 2. WORKING PROCESS

Before designing the story and challenges of the game it is necessary to determine a series of game characteristics that may affect subsequent design decisions. These features include gender, avatar control, platform, future users, narrative level, area of application and interactivity. For example, the classification of things could be used to determine the video game genre (e.g. action, adventure, fight, logic, simulation, sport, strategy, etc.).

We divide our design process into 4 parts: -

1. Ideation
2. Story and Narrative Development
3. Character and World Design
4. Player Testing and Prototyping

#### 2.1 Ideation

The idea of the game was simple as to keep avoiding obstruction and move forward. So here our basic idea of the game which used to look like this :

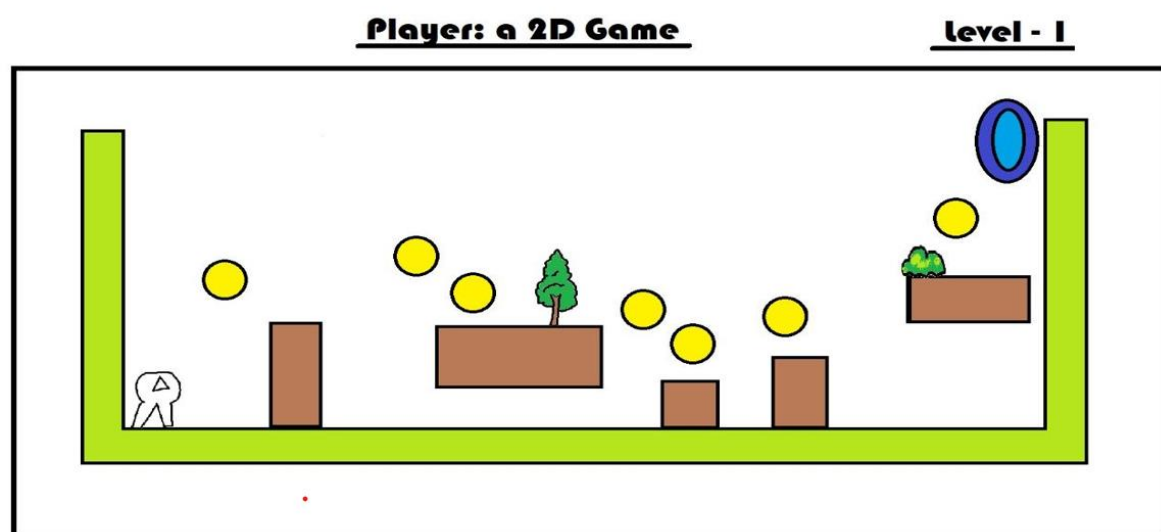


Figure 1: Basic Idea

## 2.2 Story And Narrative Development

This is where our actual work started. As we were given something like that image and we were to make it at least interesting for the player so even if they look at it once they tried to play it. With this motive we started working on the story and believe us the more inspirations you think of the easy it gets for you to search a story. The inspirations for you can come from various comics, movies, novels, cartoons or anything you like in your dream world anything like that. The game design is a place where you can make people see things from your perspective and this is the coolest things. Many of your ideas for story may come from some dreams. In our case we choose a storyline of simple but huge successful game 'Bounce'.

## 2.3 Character And World Design

Here we work on designing the characters as per the requirement of the story and the kind of game we are making like in 2-D game and considering it moving in one direction. So first we start with our player and enemy designing after adding some functionalities using coding and asset manipulation to our player and enemy, we move forward to our level designing. Here where we implement all the objects i.e. player, enemy, tree.

Here how we design our player and enemy movements:

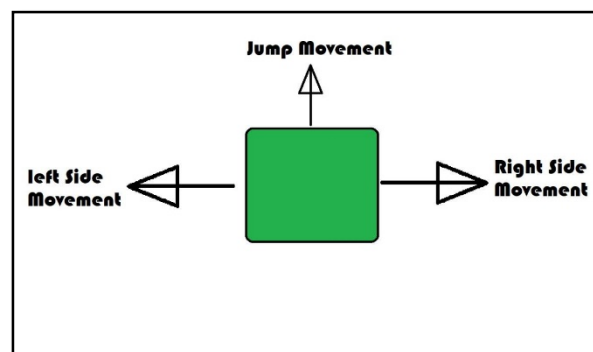


Figure 2: Player Movements

The world design was also thought out to be of some different because my world was supposed to look like some other planet and give a different feeling. I went for inspirations for the world design from various online sites like Pinterest. Thus, I worked on the background mainly on quality.

## 2.4 Player-Testing And Prototyping

It was the most amazing part because this is where you test your own made game for the first time. We continuously testing and improving our game according to our needs we add some

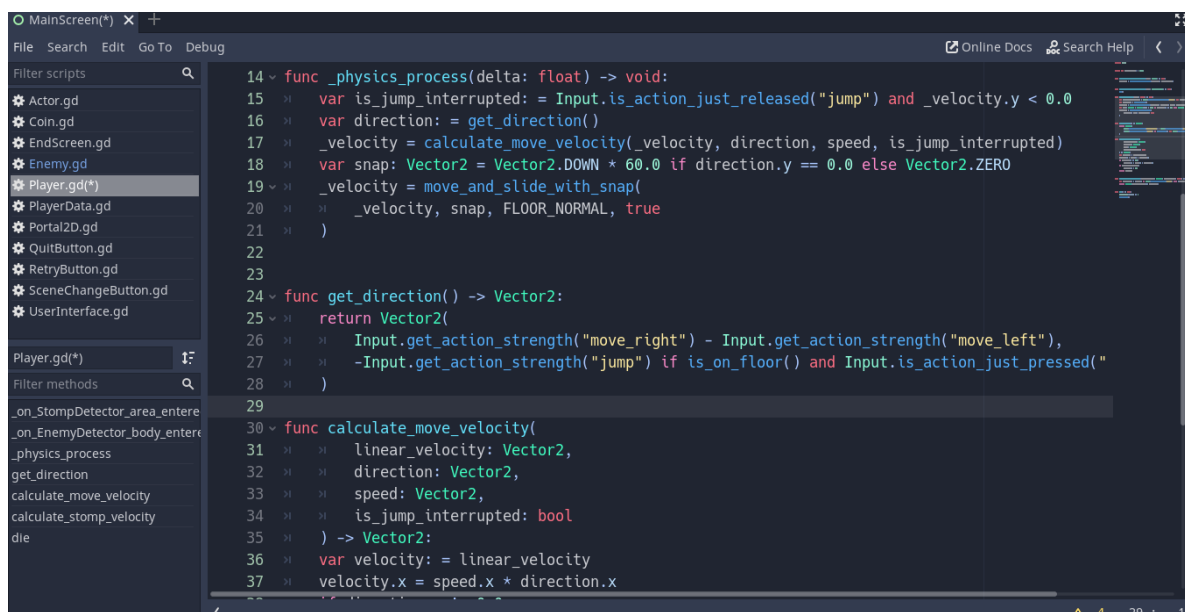


sound effects on every button pick up some matching colours so it does not look odd. We even add some effects on button's whenever they pressed.

After our first prototype testing we move forward for upgrades in our UI. We placed some minor changes in our game, like change or tone colour texture so it can be more interactive for user.

Testing process started with the testing of individual program units such as functions or objects.

These were then integrated into sub-systems and systems, and interactions of these units were tested.



```
14 func _physics_process(delta: float) -> void:
15     var is_jump_interrupted: = Input.is_action_just_released("jump") and _velocity.y < 0.0
16     var direction: = get_direction()
17     _velocity = calculate_move_velocity(_velocity, direction, speed, is_jump_interrupted)
18     var snap: Vector2 = Vector2.DOWN * 60.0 if direction.y == 0.0 else Vector2.ZERO
19     _velocity = move_and_slide_with_snap(
20         _velocity, snap, FLOOR_NORMAL, true
21     )
22
23
24 func get_direction() -> Vector2:
25     return Vector2(
26         Input.get_action_strength("move_right") - Input.get_action_strength("move_left"),
27         -Input.get_action_strength("jump") if is_on_floor() and Input.is_action_just_pressed("
28     )
29
30 func calculate_move_velocity(
31     linear_velocity: Vector2,
32     direction: Vector2,
33     speed: Vector2,
34     is_jump_interrupted: bool
35 ) -> Vector2:
36     var velocity: = linear_velocity
37     velocity.x = speed.x * direction.x
```

Figure 3: Player Testing Phase Screenshot

## Chapter 03- SYSTEM DESIGN

As games grow more complex and gamers expectations soar, the discipline of game systems design becomes ever more important. Game systems designers plan a games rules and balance, its character's attributes, most of its data, and how its AI, weapons, and objects work and interact. Let's discuss our system designing of out 2d game.

### 3. SYSTEM DESIGNING

System Designing become even more important to complete any game. For our project we divide it into 5 key factors: -

1. About Platform
2. Programming Language
3. Actor Designing
4. Level Designing
5. User Interface

#### 3.1 About Platform



*Figure 4: Godot Icon*

The Godot game engine is rather new; it became available in 2014. Only until recently it became popular. It is free and open-source, very suitable for beginner developers. It is suitable for 2D and 3D game development but is lightweight. In comparison with Unity or Unreal Engine, the quality of product is not as good. It does not have the graphical power of Unreal Engine, nor the support community of Unity, but it is easy to learn to use if you want to start your game development journey. In terms of platforms, the majority of games made with Godot are mobile games. Popular games: Sigil Breakers, Kip, The Kingdom of Avalae.

#### 3.2 Programming Language

GScript is a high-level, dynamically typed programming language used to create content. It uses a syntax similar to Python (blocks are indent-based and many keywords are similar). Its

goal is to be optimized for and tightly integrated with Godot Engine, allowing great flexibility for content creation and integration.

GScript is a Dynamically Typed language. As such, its main advantages are that:

- The language is simple and easy to learn.
- Most code can be written and changed quickly and without hassle.
- Less code written means less errors & mistakes to fix.
- Easier to read the code (less clutter).
- No compilation is required to test.
- Runtime is tiny.
- Duck-typing and polymorphism by nature.

This, translated to reality, means that Godot+GScript are a combination designed to create games quickly and efficiently. For games that are very computationally intensive and can't benefit from the engine built-in tools (such as the Vector types, Physics Engine, Math library, etc.), the possibility of using C++ is present too. This allows you to still create most of the game in GScript and add small bits of C++ in the areas that need a performance boost.

### 3.3 Actor Designing

Firstly, we design our actors. We use two actors in this game one is Player and second one is Enemy. Because they almost have same functionalities we put it under one node that is actor. In designing of actors we add moments in it and after that we integrated it to the coding so we can control our actors using keyboard keys (i.e. W, S, A, D)

Here are some screenshots of designing our actors in Godot platform: -

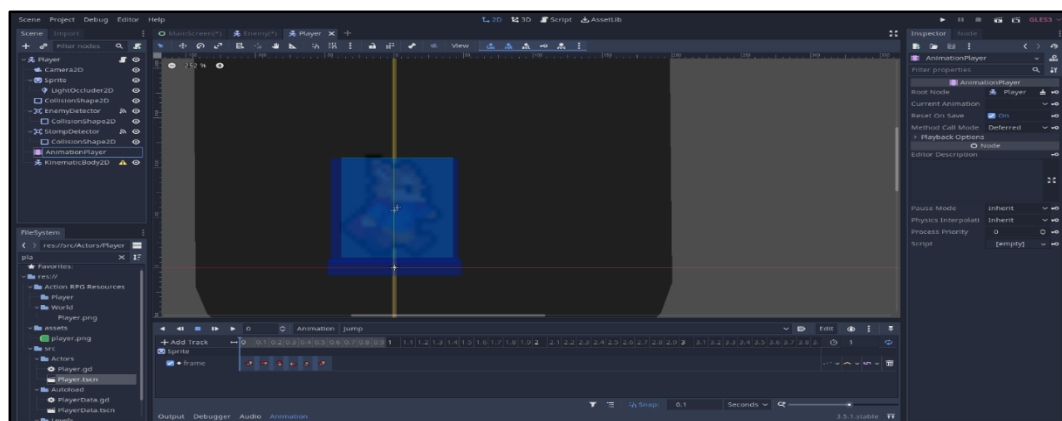


Figure 5 : Player Designing Process

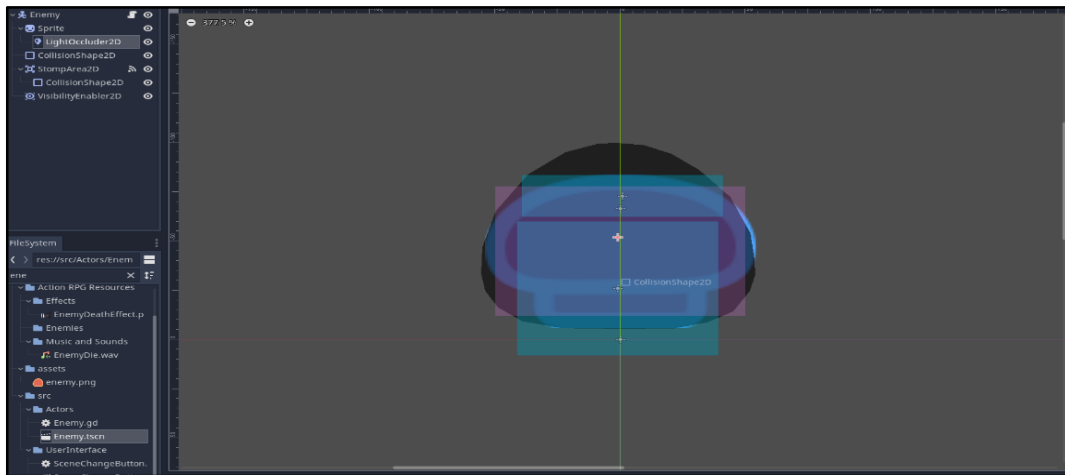


Figure 6: Enemy Designing Process

### 3.4 Level Designing

Godot give us an environment to create multiple levels and give us multiple functionalities to upgrade our levels. Put it simply we can say we just need to duplicate this scene tree and attach the same script to it. The easiest way to do this is to use Scene | Save As and save the level as Level2.tscn. Then, you can use some of the existing tiles or draw a whole new level layout. Feel free to do this with as many levels as you like, making sure to save them all in the levels folder. After that we link them together so that each level will lead to the next. Even we arrange them incorrectly we will be able to put them in whatever order we like. For creating levels in Godot we are using Tilemap. Before using Tilemap we need to give Tileset to Tilemap. After we successfully created tilemap we just need to drag our cursor wherever we need to implement our Tilemap.

While implementing Tilemap into level environment it looks like this:

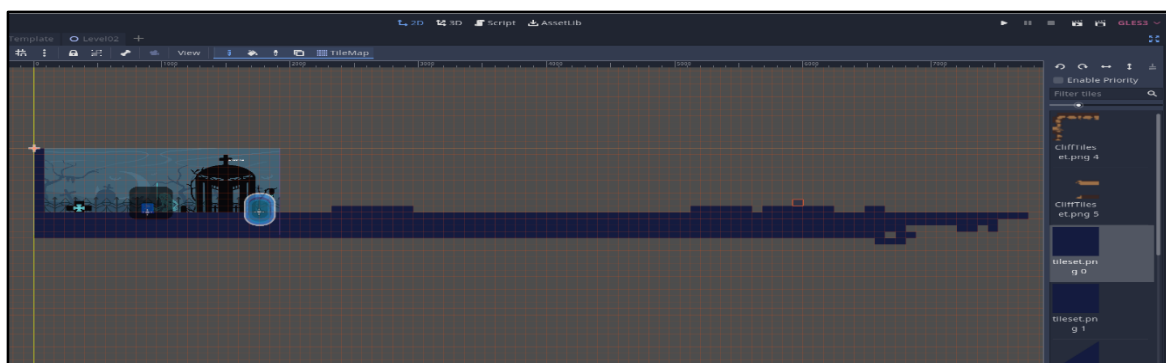


Figure 7: Tilemapping in Level Designing

After successfully implement Tilemapping we start creating environment for our game for that we put assets & designed actors into this level and arrange all of these in a satisfactory way. After all the things done in level environment different levels look like this:

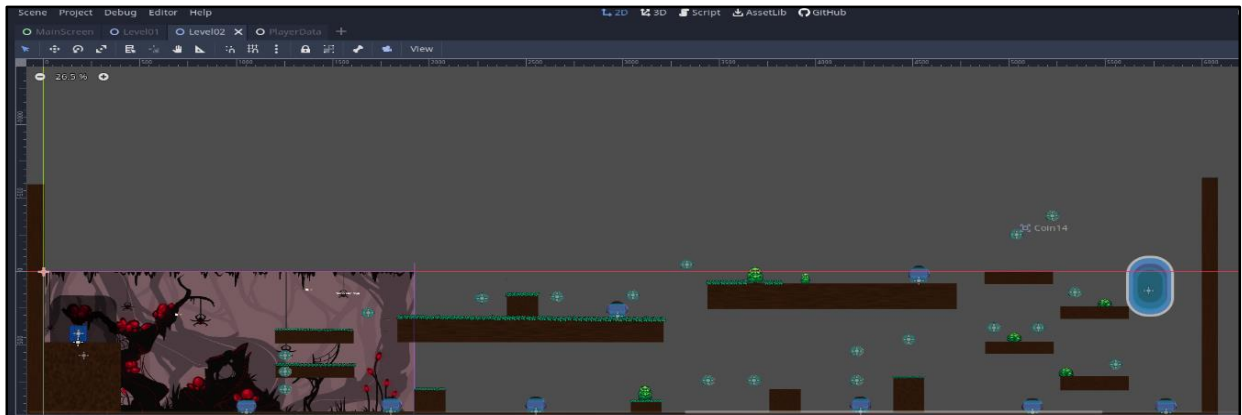


Figure 8: Level 2 Design

### 3.5 User Interface(UI)

In User Interface we have many functionalities like MainScreen, EndScreen, Buttons, Button Effects, Sound Effects, Background Music.

#### (i) MainScreen and EndScreen

MainScreen will give us options to start or quit our game. So we can control our Menu using buttons which connected via coding to Level 1 screen. EndScreen will appear when player will complete all the levels. In this screen we will give user options to play again or quit. And also all the buttons are connected via coding to Level 1 and Quit button.

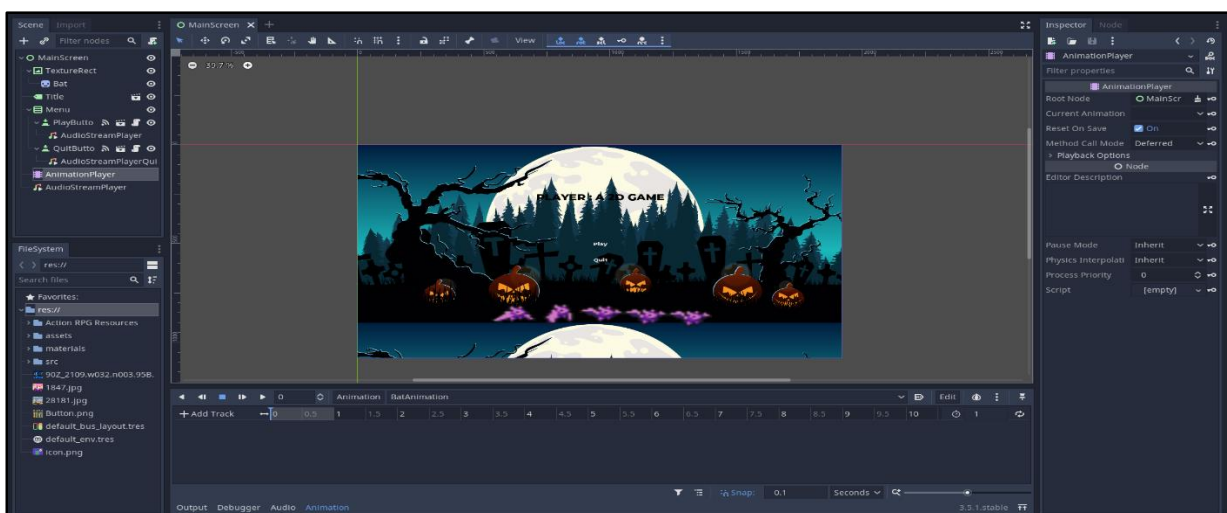
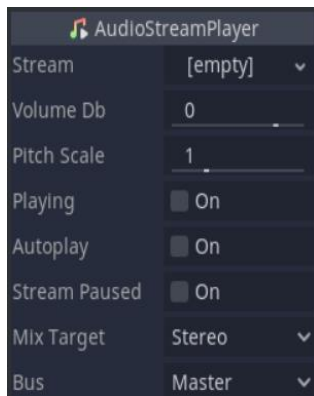


Figure 9: MainScreen

## (ii) **AudioStreamPlayer**

In Godot for adding sound effects we use AudioStreamPlayer. Basically an audio stream is an abstract object that emits sound. The sound can come from many places, but is most commonly loaded from the file system. Audio files can be loaded as AudioStreams and placed inside an AudioStreamPlayer. You can find information on supported formats and differences in Importing audio samples. where we put our downloaded audio so it can be played whenever user playing the game.



*Figure 10: AudioPlayerBlock*

## Chapter 04 - RESULT AND CONCLUSION

### 4. PROJECT RESULT & CONCLUSION

We learned a lot from this project about 2D game development. How things actually get done and how sometime game making can be easy and how it can be so much difficult to build games.

#### 4.1 Project Result

Here is the final result of our 2D game making:-

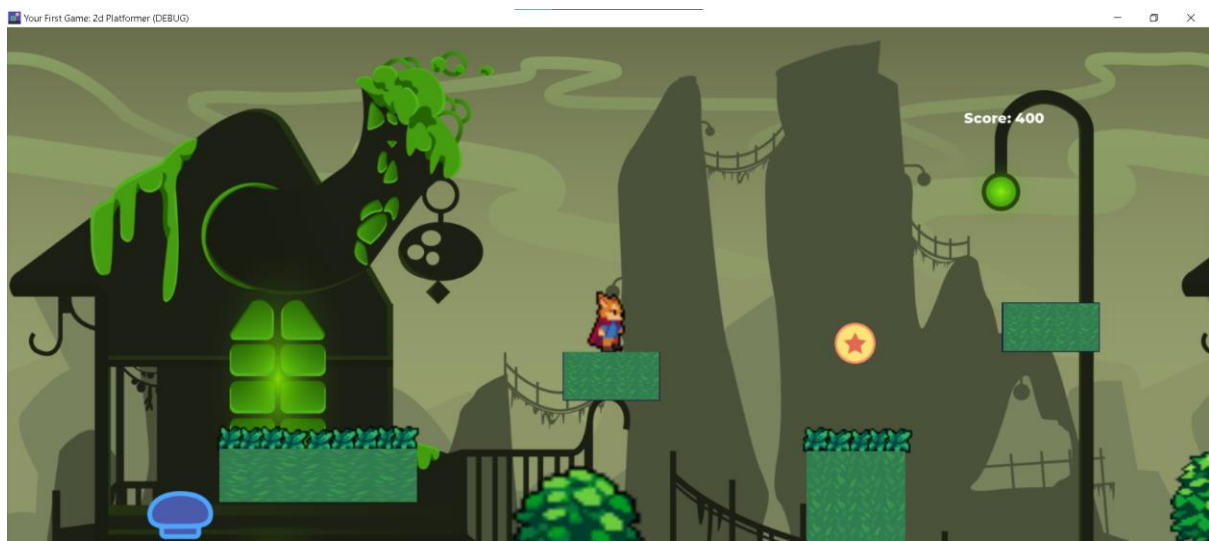
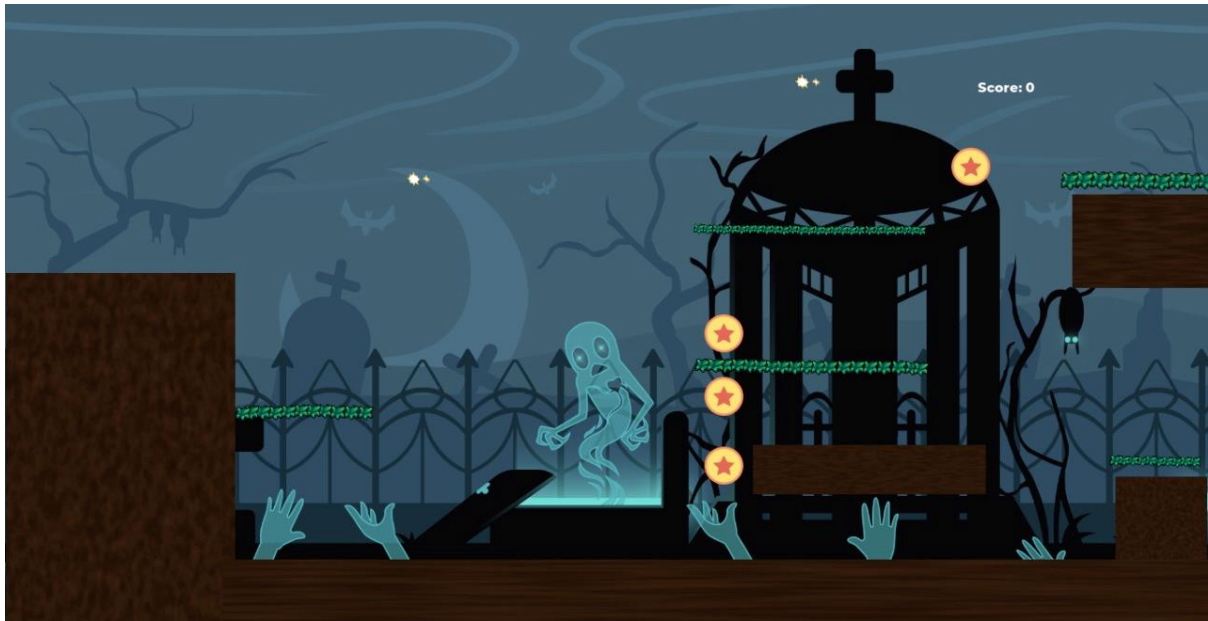


Figure 11: Level 1 Screenshot



Figure 12: Level 2 Screenshot





*Figure 13: Level 3 Screenshot*

## 4.2 Project Conclusion

The project was started with modest aim with no prior experience in any programming projects as this, but ended up in learning many things, fine tuning the programming skills and getting into the real world of software development with an exposure to corporate environment. During the development of any software of significant utility, we are forced with the trade-off between speed of execution and amount of memory consumed. This is simple interactive application. It is extremely user friendly and has the features, which makes simple graphics project. It has an open source and no security features has been included. The user is free to alter the code for feature enhancement. Checking and verification of all possible types of the functions are taken care. Care was taken to avoid bugs. Bugs may be reported to creator as the need.

Further this project can be enhanced by adding few more options i.e menu in game with more options in it, Better animation of player movements and enemy movements. Using all these we have a idea how the things are done. So we can move forward for a 3D game, more interactive and with a better story and concepts.



## REFERENCES

1. [https://docs.godotengine.org/en/stable/tutorials/2d/using\\_tilemaps.html](https://docs.godotengine.org/en/stable/tutorials/2d/using_tilemaps.html)
2. [https://www.youtube.com/watch?v=c2mkyW\\_TymY](https://www.youtube.com/watch?v=c2mkyW_TymY)
3. [https://docs.godotengine.org/en/stable/tutorials/scripting/gdscript/gdscript\\_basics.html](https://docs.godotengine.org/en/stable/tutorials/scripting/gdscript/gdscript_basics.html)
4. <https://docs.godotengine.org/en/stable/tutorials/ui/index.html>
5. [https://docs.godotengine.org/en/stable/tutorials/audio/audio\\_streams.html](https://docs.godotengine.org/en/stable/tutorials/audio/audio_streams.html)
6. <https://docs.godotengine.org/en/stable/tutorials/physics/index.html>