**INTERNATIONAL UNIVERSITY**

**VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY**

**School of Computer Science and Engineering**

**-----\*\*\*-----**

Ảnh có chứa vòng tròn, biểu tượng, văn bản, Nhãn hiệu

Mô tả được tạo tự động

**PROJECT REPORT**

**Minesweeper**

**Adviser: Dr Vi Chi Thanh**

**Course: Data Structure & Algorithm**

**Semester 2 (2024-2025)**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Full Name | Student’s ID | Contribution |
| 1 | Nguyễn Đức Nguyên Phúc | ITDSIU21108 | 100% |

**TABLE OF CONTENTS**

**[LIST OF FIGURES](#_Toc155965767)** [2](#_Toc155965767)

**[ABSTRACT](#_Toc155965768)** [3](#_Toc155965768)

**[CHAPTER 1: INTRODUCTION](#_Toc155965769)** [4](#_Toc155965769)

**[1.](#_Toc155965770)****[Objectives](#_Toc155965770)** [4](#_Toc155965770)

**[2.](#_Toc155965771)****[The tools used](#_Toc155965771)** [5](#_Toc155965771)

**[CHAPTER 2: METHODOLOGY](#_Toc155965772)** [6](#_Toc155965772)

**[1.](#_Toc155965773)****[Rules](#_Toc155965773)** [6](#_Toc155965773)

**[2.](#_Toc155965774)****[Design](#_Toc155965774)** [7](#_Toc155965774)

**[3.](#_Toc155965775)****[UML Diagram](#_Toc155965775)** [12](#_Toc155965775)

**[CHAPTER 3: DEMO – RESULT](#_Toc155965776)** [20](#_Toc155965776)

**[CHAPTER 4: CONCLUSION AND FUTURE WORKS](#_Toc155965777)** [23](#_Toc155965777)

**[1.](#_Toc155965778)****[Conclusion](#_Toc155965778)** [23](#_Toc155965778)

**[2.](#_Toc155965779)****[Future works](#_Toc155965779)** [23](#_Toc155965779)

**[3.](#_Toc155965780)****[Acknowledgment](#_Toc155965780)** [24](#_Toc155965780)

**[REFERENCES](#_Toc155965781)** [25](#_Toc155965781)

# 

# **LIST OF FIGURES**

[Figure 1. Dragon Tale banner 5](#_Toc155996335)

[Figure 2. GitHub statistics 5](#_Toc155996336)

[Figure 3. Help screen 7](#_Toc155996337)

[Figure 4. Player 8](#_Toc155996338)

[Figure 5. Mini monsters 9](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996339)

[Figure 6. Boss 9](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996340)

[Figure 7. Health 9](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996341)

[Figure 8. Fireball 9](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996342)

[Figure 9. Coin 9](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996343)

[Figure 10. Venom 10](#_Toc155996344)

[Figure 11. Project structure 11](#_Toc155996345)

[Figure 12. GameState diagram 13](#_Toc155996346)

[Figure 13. Enemy diagram 14](#_Toc155996347)

[Figure 14. DieEnemies diagram 14](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996348)

[Figure 15. Player diagram 15](#_Toc155996349)

[Figure 16. Objects diagram 16](#_Toc155996350)

[Figure 17. Level diagram 17](#_Toc155996351)

[Figure 18. Ui diagram 18](#_Toc155996352)

[Figure 19. Main diagram 19](#_Toc155996353)

[Figure 20. Menu screen from the game 20](file:///C:\\Users\\Tran%20Phan\\Downloads\\REPORT%20-%20OOP.docx" \l "_Toc155996354)

[Figure 21. Help screen 20](#_Toc155996355)

[Figure 22. The start of the game 21](#_Toc155996356)

[Figure 23. During the game play 21](#_Toc155996357)

[Figure 24. End the game 22](#_Toc155996358)

# 

# **ABSTRACT**

Minesweeper is a logic puzzle game based on abstraction and simplification. The goal is to locate mines in a fixed-size grid by flagging or clicking cells without bombs. In case, a player clicks on a cell containing a bomb, all cells with bombs are revealed, resulting in a loss. Conversely, if a player successfully flags all bombs without clicking on any, they win.

Dragon Tale combines strategic thinking and prudent decision-making, offering players an immersive experience as they analyze and predict the position of mines. In this endeavor, I have developed the game "Minesweeper" to provide players with a version suitable for all ages. The game's theme is based on the concept of the original Minesweeper game.

Keywords: Minesweeper, logic, Data Structure & Algorithm.

# **CHAPTER 1: INTRODUCTION**

### **Objectives**

The project aims to develop a game that combines puzzle logic and strategy game concepts. It showcases core principles of object-oriented programming, data structures, and algorithms. As a standalone game, it requires analytical thinking and problem-solving skills. Despite its simplicity, the game is accessible to players of all ages, fostering critical thinking.

In Minesweeper, players navigate a pointer to clicking on cells to reveal safe spaces while avoiding bombs. The game also involves strategic flagging of suspected bomb locations.

In summary, the project's objectives are:

* Create an entertaining and engaging game experience.
* Implement object-oriented programming techniques and data structures/algorithms.
* Enhance the game's management and code efficiency.
* Explore potential features for future expansion.

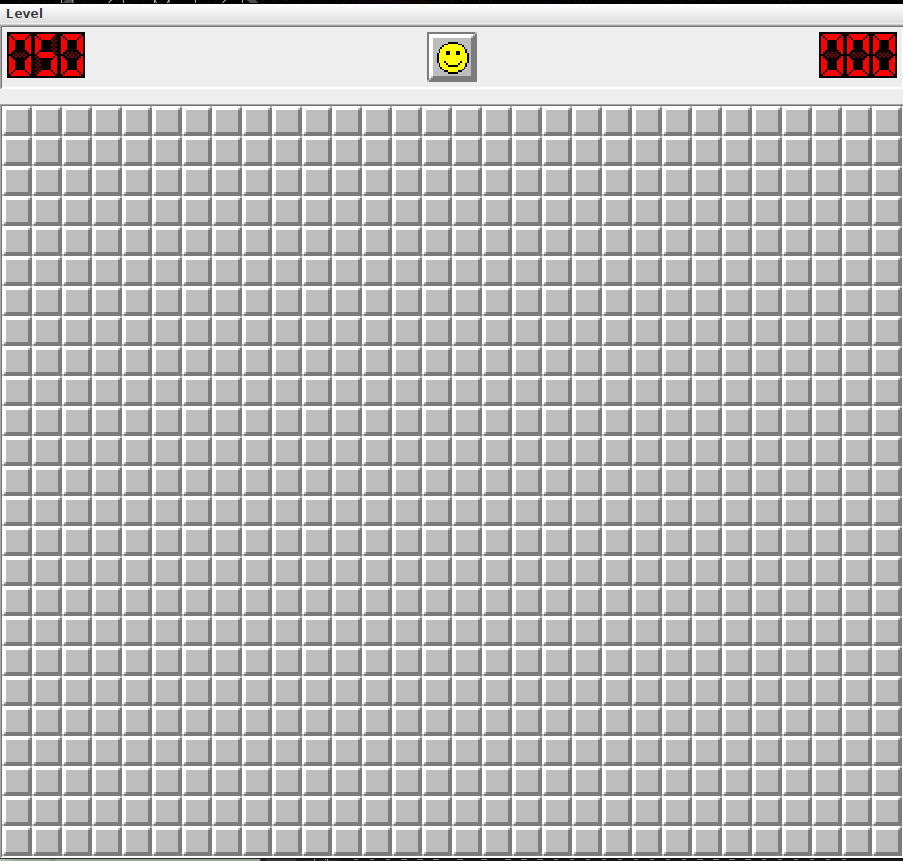


Figure 1. Minesweeper

### **The tools used**

* IDE for programming and debugging: IntelliJ IDEA, Eclipse
* Java Development Kit: 21.
* Mean of code version management: GitHub.

A graph with a green line

Description automatically generated

Figure 2. GitHub statistics

# **CHAPTER 2: METHODOLOGY**

### **Rules**

Key Rules of the Game:

When clicking on a cell:

* If it contains a bomb, you lose.
* If it's safe, a number appears, indicating the number of bombs surrounding it.



Figure 3. Example

* For instance: If you click on a cell with the value 4, it indicates that there are four randomly placed bombs in the adjacent cells.

### **Design**

* 1. *UI/UX*

Prioritizing UI/UX before our game's launch enabled us to gather valuable feedback from external sources, guiding us in establishing a consistent design pattern.

Minesweeper is a strategy and puzzle game that challenges players to locate hidden mines on a game board. The game requires a combination of strategic thinking and luck to succeed.

In this game, players click on unrevealed empty squares. When a square is clicked, several outcomes are possible:

* In case, no surrounding cells contain mines, the square is revealed as blank.
* In case, surrounding cells contain mines, the square displays a number indicating the number of mines adjacent to it.
* In case, a player clicks on a cell containing an unrevealed mine, it converts to a revealed mine, and all unrevealed mines on the board are shown. Consequently, the game is lost.
* In case, you suspect that a cell contains a hidden mine, you can flag it to prevent accidentally clicking on it.
* It is essential to avoid clicking on cells that contain unrevealed mines.

Crucial Components for Minesweeper UI/UX:

* Information to display:

Screenshot from 2024-03-19 11-41-32 

Figure 4. an unrevealed empty square Figure 5. an revealed as blank

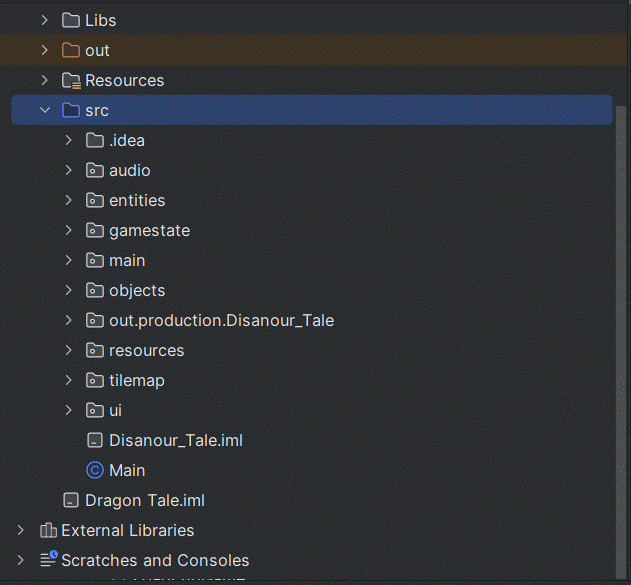
Figure 6: an revealed mine Figure 7: an unrevealed mine



*Figure 8: Digits (‘1’-’8’)*

* 1. *Game algorithm*

After much iteration and troubleshooting, we now have the project structure as seen in the image below.



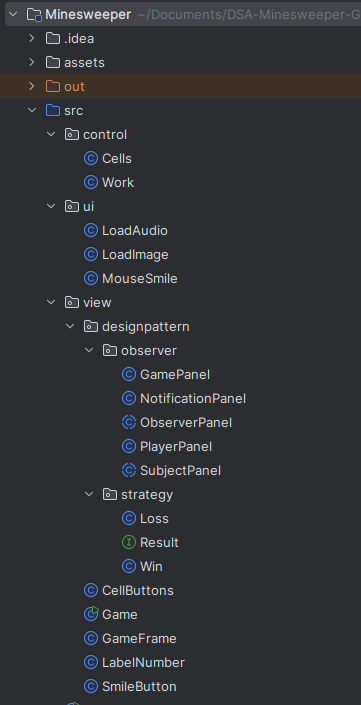


Figure 9. Project structure

The classes we teach can be arranged into distinct groups, like:

* Cells: is class to implement the cells in the board including coordinates (x,y) to identify the cells in board
* Work: is class to operate the game with some actions such as: create the board, fill the number and mine into the board and check win or loss
* UI is a package to load audio, image and control the mouse so it contains LoadAduio class, LoadImage class and MouseSimle class
* CellButtons: is class
* MapObject: is the superclass of the classes including the Player class to display the player's information (dragon tale), the Enemies class to describe various monsters in the road, the teleport class, the venom class to describe the enemies' venom to attack the player, and the collectable class to describe the coins in the journey.
* Enemies: is the superclass of all the classes, which includes the boss category or classes that are similar to mini monsters like Hero, Slugger, Arachnik, and Hat Monkey.
* Die Enemies is the superclass of all the classes, which includes DieBoss, DieHero, DieSlugger, DieArachnik, and DieHatMonkey.
* Animation: is used to describe the frame's actions of Entities.
* HUD: displays information including fireballs, health, coins, and scores.
* Background: displays the background of the game
* Tile and TileMap: are used to display deep holes and walls in the game.
* AudioPlayer: creates the music, even the sound of Dragon’s firing and scratching.
* Abstract class GameState displays the state of the game; its subclass HelpState demonstrates how to play; Level1State depicts the initial fight screen; MenuState shows several options; and WinnerState and GameOverState indicate when the player succeeds or fails. GameStateManager: is manager class various the state class inherits GameState class.
* Game and GamePanel: is initiated even controlling the programing.

### **UML Diagram**

We provided the UML diagrams for the entire project and each group that was addressed to help you better understand the structure and methods.

* **Whole UML Diagram:**

[OOP-DragonTale-Game/UML/UML.pdf at main · nguyenducnguyenphuc2002/OOP-DragonTale-Game (github.com)](https://github.com/nguyenducnguyenphuc2002/OOP-DragonTale-Game/blob/main/UML/UML.pdf?fbclid=IwAR1Z7bxsTsBwutZSRkIEHaIkuRv-JppvztfkOhtS8LlmG2UlxeCfxrhrzJw), **access** to view the whole UML Diagram.

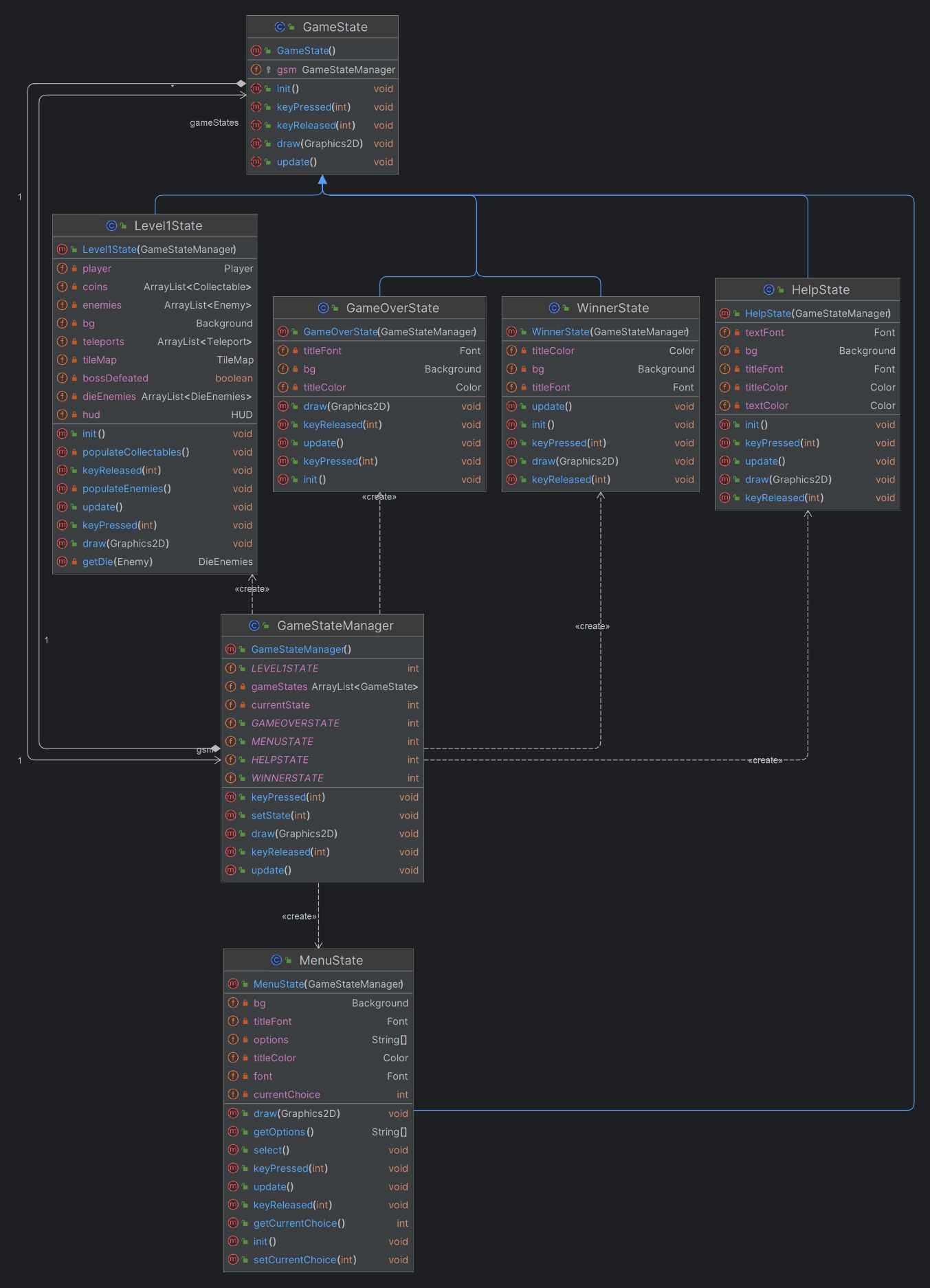


Figure 12. GameState diagram

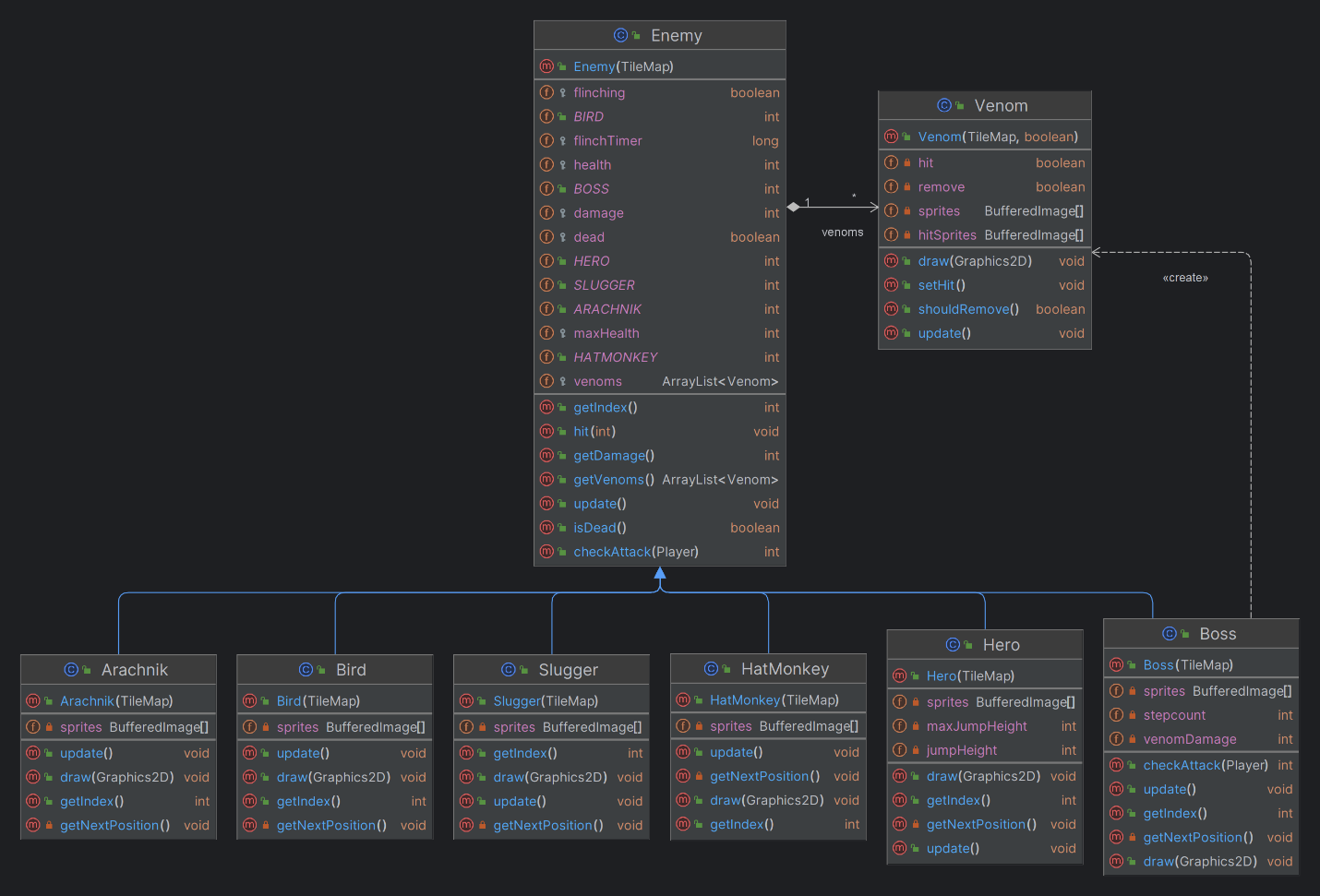


Figure 13. Enemy diagram

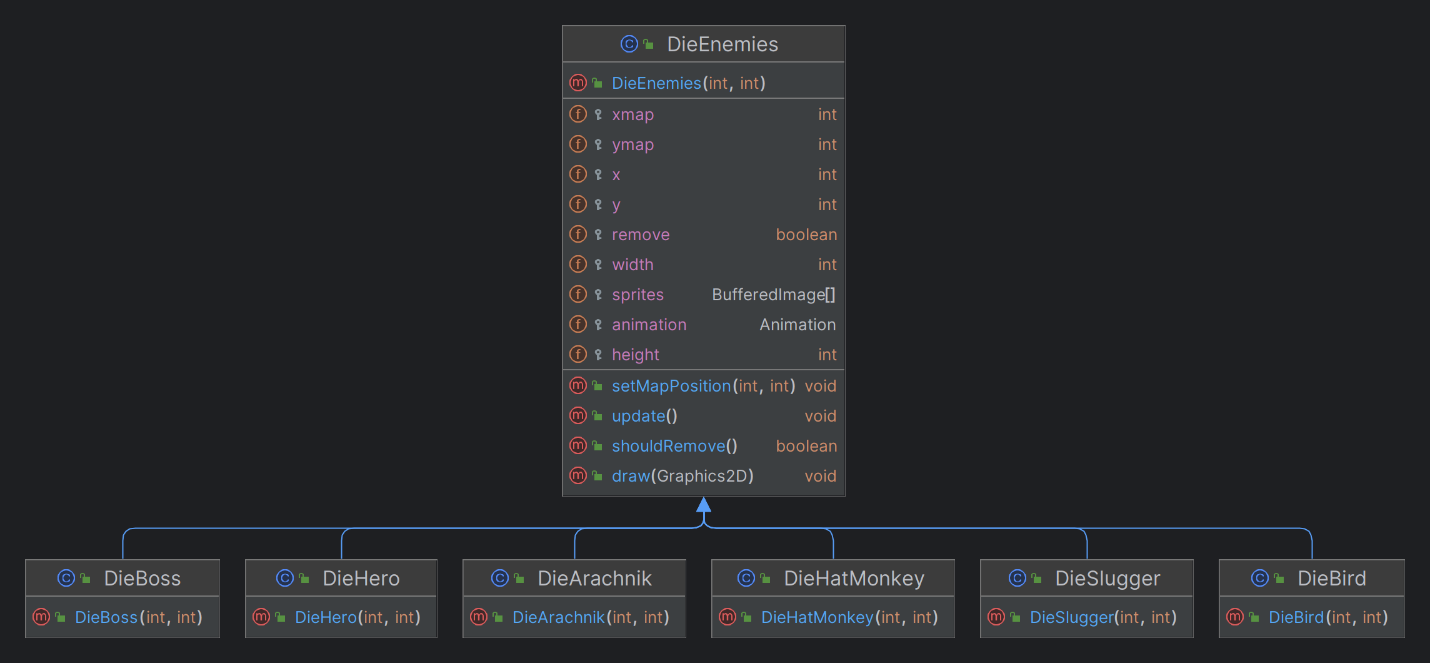


Figure 14. DieEnemies diagram

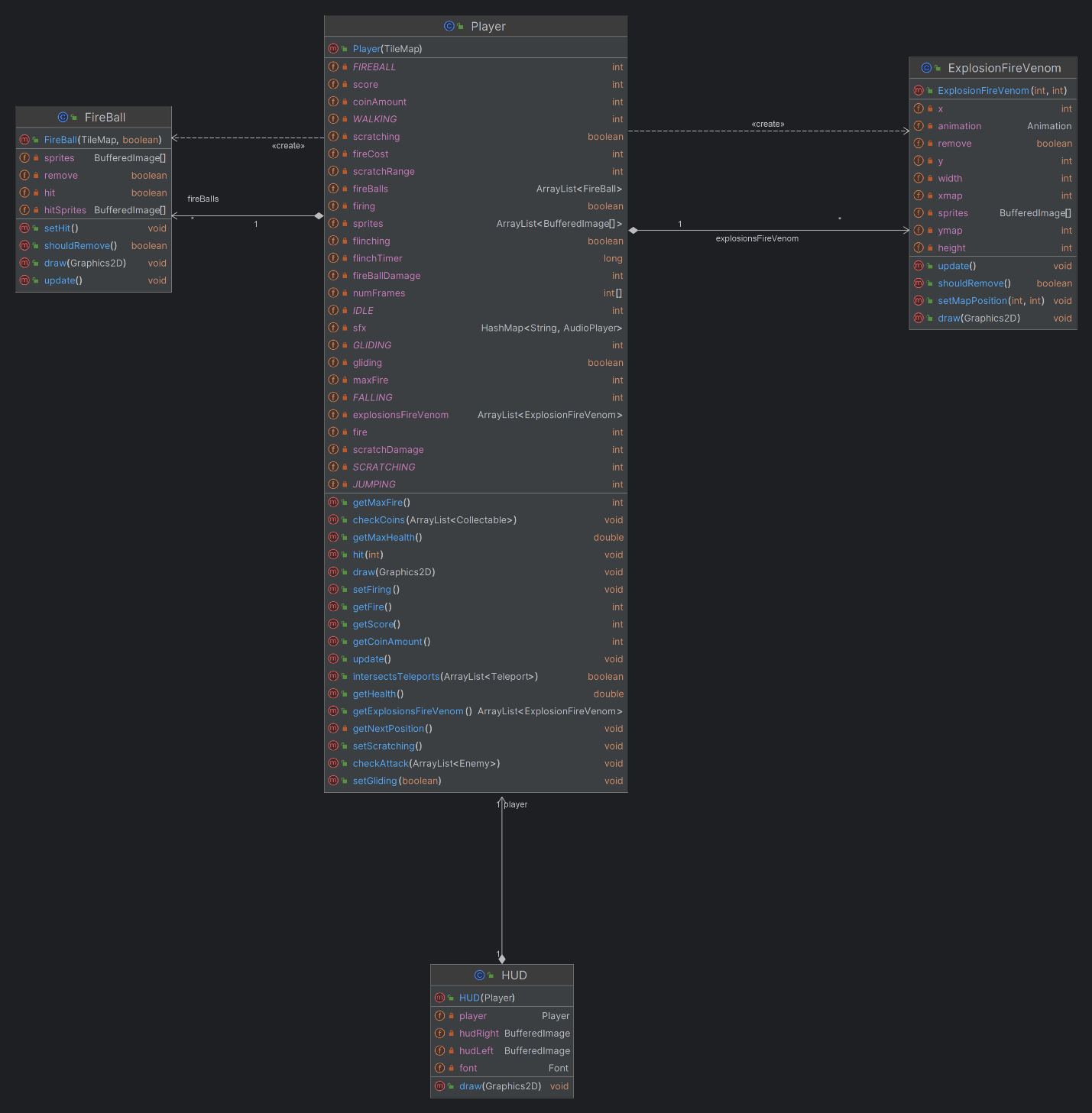


Figure 15. Player diagram

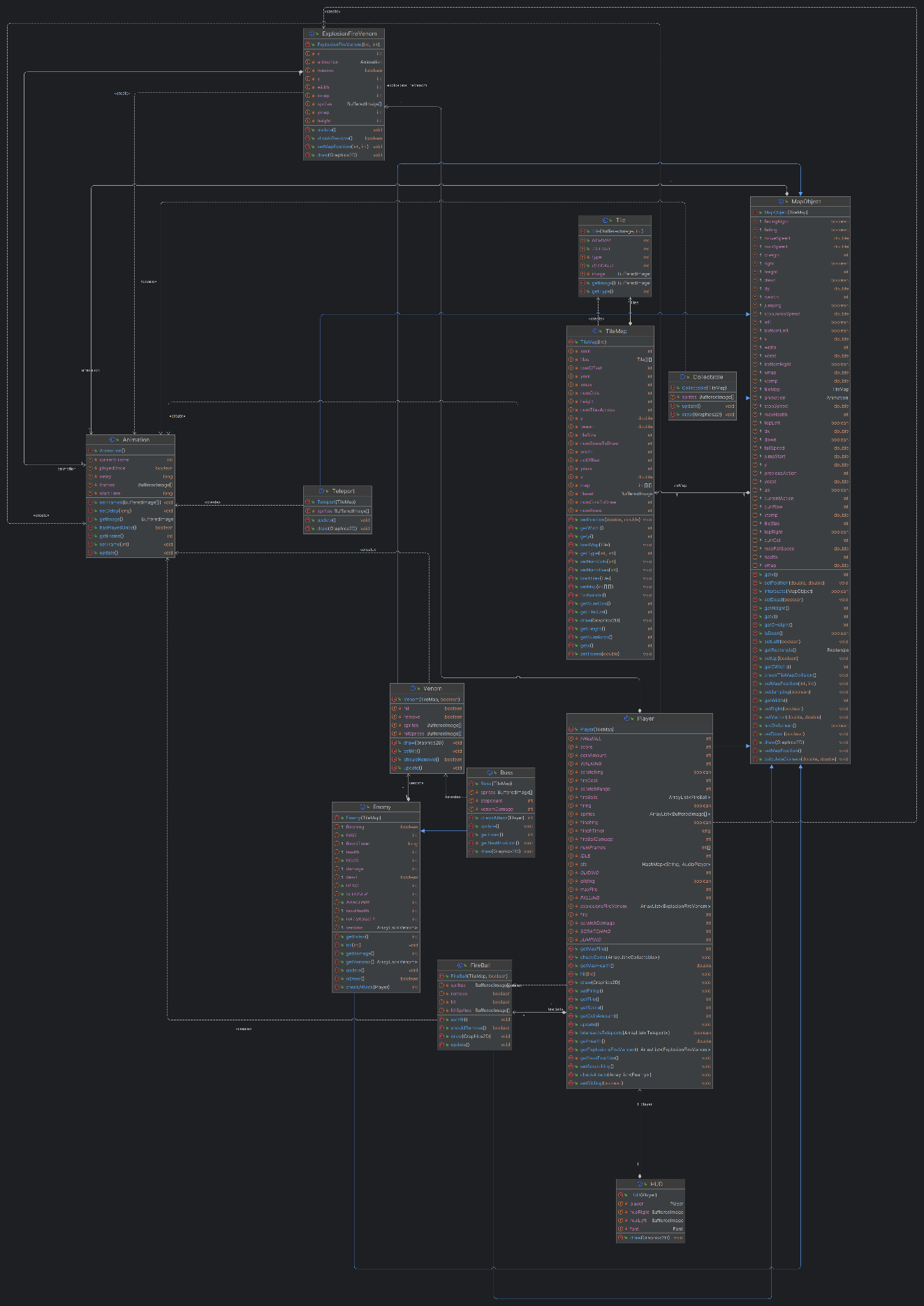


Figure 16. Objects diagram

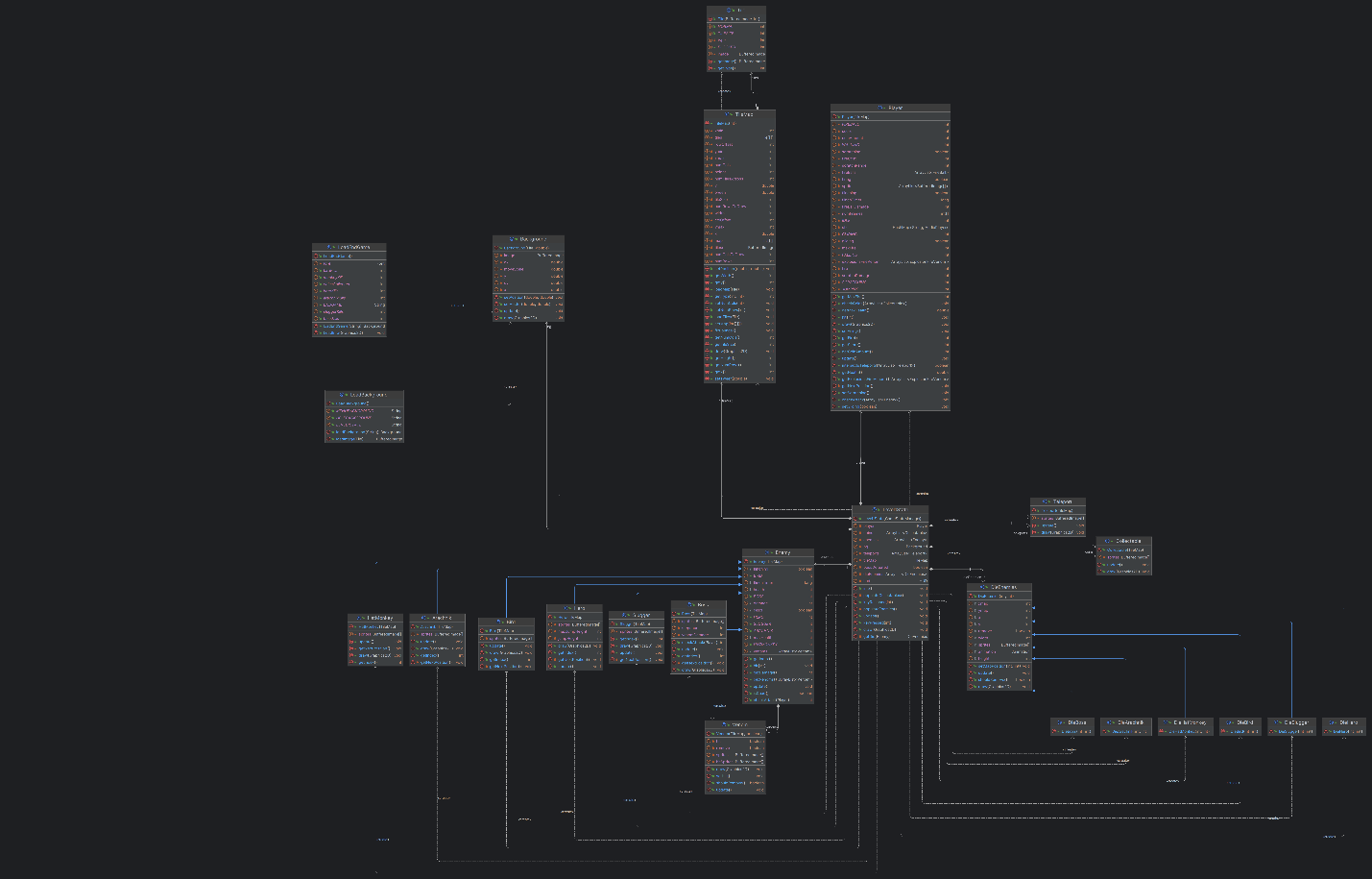


Figure 17. Level diagram

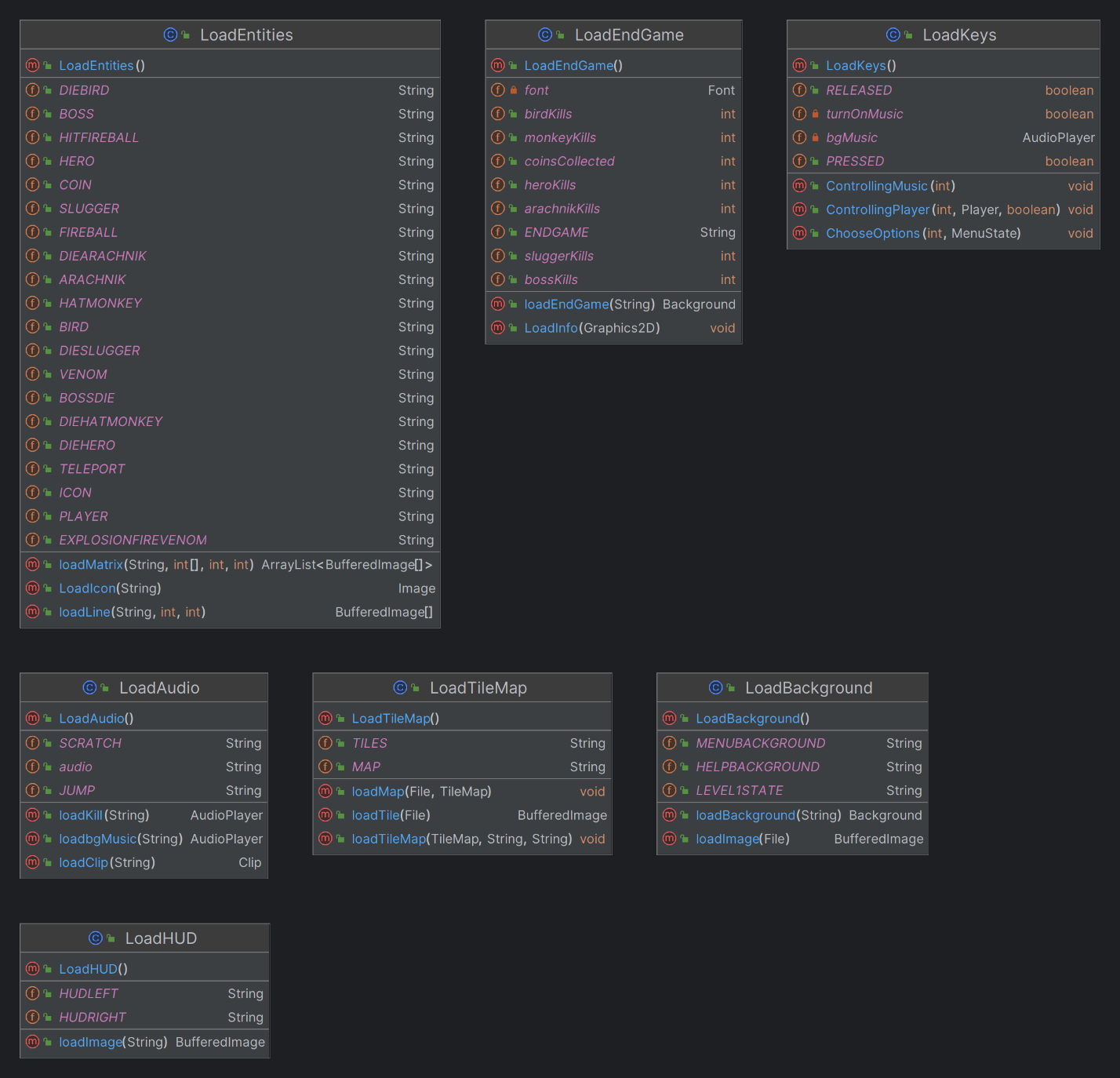


Figure 18. Ui diagram

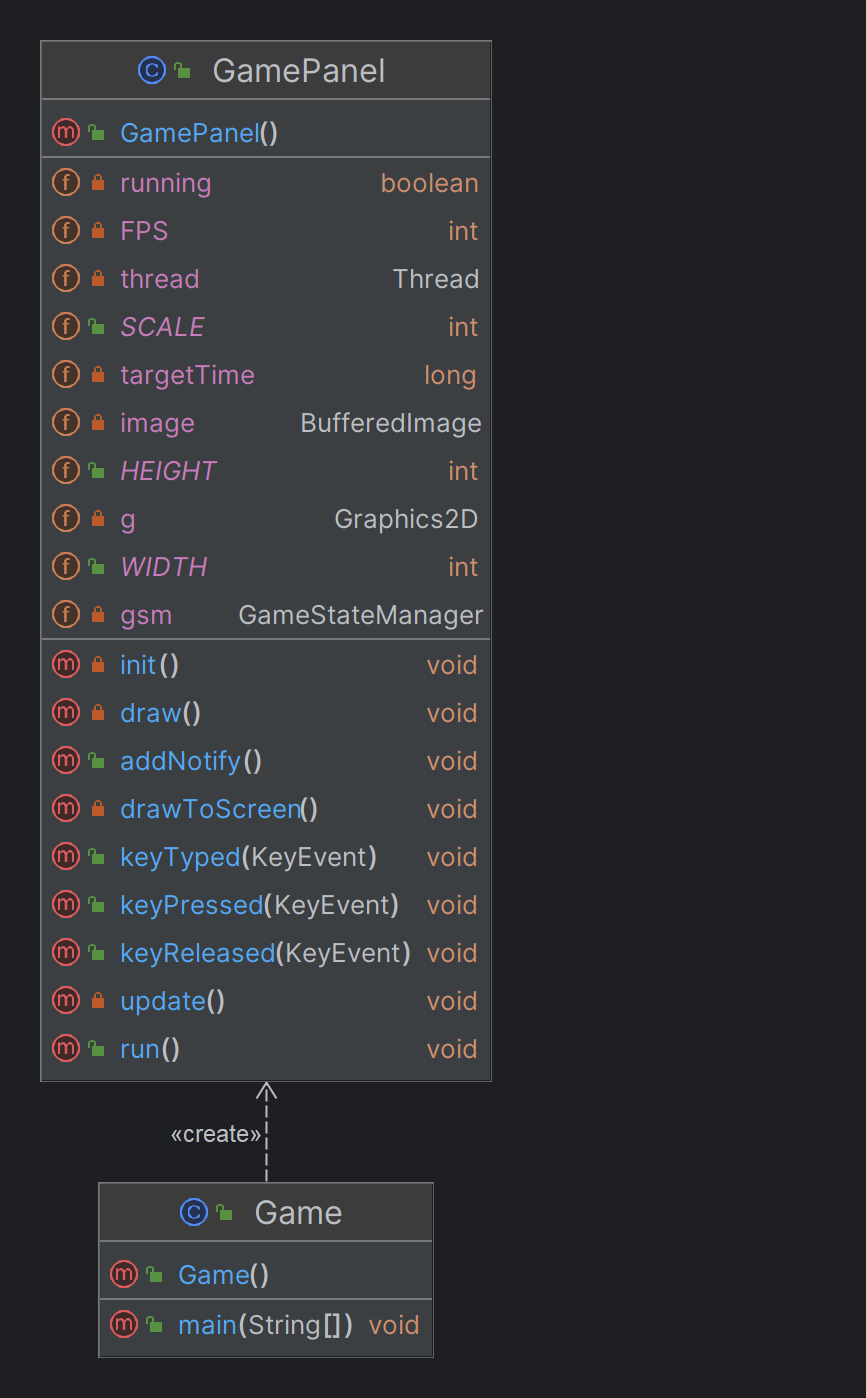


Figure 19. Main diagram

# **CHAPTER 3: DEMO – RESULT**

Ảnh có chứa văn bản, ảnh chụp màn hình, Phim hoạt hình, phim hoạt hình

Mô tả được tạo tự độngTo test our game on a machine that had an IDE and Java Development Kit 21 installed. We pulled our git repository and ran the Main class to launch the game. The following screenshots show some examples of the game states in the current build.

Figure 20. Menu screen from the game

Ảnh có chứa văn bản, Phim hoạt hình, hình mẫu, ảnh chụp màn hình

Mô tả được tạo tự động

Figure 21. Help screen

Ảnh có chứa ảnh chụp màn hình, phim hoạt hình, Phim hoạt hình, hình mẫu

Mô tả được tạo tự động

Figure 22. The start of the game

Ảnh có chứa ảnh chụp màn hình, Phim hoạt hình, Hoạt hình, hình mẫu

Mô tả được tạo tự động Ảnh có chứa ảnh chụp màn hình, phim hoạt hình, Phim hoạt hình, Hoạt hình

Mô tả được tạo tự động

Ảnh có chứa phim hoạt hình, màu xanh lá cây

Mô tả được tạo tự động

Figure 23. During the game play

*Ảnh có chứa văn bản, ảnh chụp màn hình, phim hoạt hình

Mô tả được tạo tự động*

Figure 24. End the game

# **CHAPTER 4: CONCLUSION AND FUTURE WORKS**

### **Conclusion**

The game's development has not yet ended. In the final term, the team has acquired a deeper understanding of the four fundamental features of Object-Oriented Programming (OOP) and the SOLID principles. This knowledge has been instrumental in enhancing our proficiency in OOP within the realm of game development as well as in the programming process after finishing a game with some novel features compared to the original version. The project's classes have discussed the concept of encapsulation. Objects belong to enemies’ package and player package is where inheritance, abstraction, and polymorphism have been used the most frequently. Because of this, Dragon Tale was strictly developed using the fundamental idea of OOP, and the game code contains all four key OOP features and a design pattern learned from class. As we approach the conclusion of the project, it is crucial to acknowledge that these team members will not be afforded a second opportunity to work on this. The wealth of knowledge gained in this endeavor is not only a testament to our collective expertise but also a testament to the innovative spirit that has driven us to push the boundaries of game development.

### **Future works**

Unfortunately, the team was hoping to develop power-up items that players can collect during gameplay. These power-ups could enhance the player's abilities, offer temporary invincibility, or provide special attacks, adding an extra layer of strategy to the game. In addition, expand the gaming experience by introducing additional levels with distinct challenges, environments, and enemy types, design levels with a mix of terrains, including platforms, cliffs, and water bodies. Varying terrains will require players to adapt their movement strategies, adding an element of exploration and agility to the gameplay. Furthermore, incorporating a feature that allows players to customize their characters with various skins, outfits, and accessories will also be applied in future updates, players can tailor their characters to reflect their individual style preferences, creating a more immersive and enjoyable gameplay journey. Therefore, any new commitments are highly appreciated.

1. **Acknowledgment**

We would like to convey our deepest appreciation to our instructor and individuals who assisted us in reaching the goals of this project:

* Dr. Tran Thanh Tung
* MSc. Nguyen Quang Phu
* Original code from foreignguymike (Foreignguymike, 2018)
* The sites Geeksforgeeks, Javapoints, and so on
* The README.md template from othneildrew (Drew, 2018/2022)

# **REFERENCES**

*[1] ForeignGuyMike. (2018). foreignguymike/dragontale [Java]. Retrieved*

*December 10, 2023, from*

*https://github.com/foreignguymike/legacyYTtutorials/tree/master/dragontale*

*[2] Java Swing Tutorial. Javatpoint. (n.d.). Retrieved December 11, 2023, from*

*https://www.javatpoint.com/java-swing*

*[3] UML Diagrams Tutorial. Javatpoint. (n.d.). Retrieved January 4, 2023, from*

*https://www.javatpoint.com/uml-diagrams*

*[4] Drew, O. (2022). Othneildrew/Best-README-Template. Retrieved January 4,*

*2023, from*

*https://github.com/othneildrew/Best-README-Template*

*[5] SOLID Principle in Programming: Understand With Real Life Examples.*

*GeeksforGeeks. Retrieved December 28, 2023, from*

*<https://www.geeksforgeeks.org/solid-principle-in-programming-understand-with-real-life-examples/>*

*[6] [Playing music in a Java](https://stackoverflow.com/questions/27854171/playing-music-in-a-java-game" \t "_blank) game. stackoverflow. Retrieved December 16, 2023,*

*from*

*https://stackoverflow.com/questions/27854171/playing-music-in-a-java-game*

*[7] [Ileriayo](https://github.com/Ileriayo" \t "_blank) Adebiyi. (2020). [Ileriayo](https://github.com/Ileriayo" \t "_blank)/ [markdown-](https://github.com/Ileriayo/markdown-badges" \t "_blank)badges. Retrieved January 4,*

*2023, from https://github.com/Ileriayo/markdown-badges*

*[8] The Java™ Tutorials: Coordinates. Oracle. Retrieved December 10, 2023, from*

*https://docs.oracle.com/javase/tutorial/2d/overview/coordinate.html*