VIETNAM NATIONAL UNIVERSITY HCM
INTERNATIONAL UNIVERSITY

School of Computer Science & Engineering

PLANTS VS. ZOMBIES PROJECT

Course: OBJECT - ORIENTED PROGRAMMING

Group members:

No	Full name	Student ID
1	Dương Trọng Nghĩa	ITITIU21256
2	Lê Đăng Khoa	ITITIU21227
3	Ngô Thị Thương	ITCSIU21160
4	Nguyễn Phạm Kỳ Phương	ITITIU21287

Table of Contents

1. Abstract

The Plants vs Zombies game, developed and published by Popcap Game, is inspired by titles such as Magic: The Gathering and Warcraft III: Reign of Chaos, along with the Swiss Family movie Robinson Plants vs Zombies as the game.
In a tower defense video game, the player's goal is to prevent zombies from reaching the house. When a horde of zombies begins to attempt to approach the house along parallel lanes, the player must protect the house by planting trees in the lane to shoot bullets that kill the zombies or harm them. Players collect a currency called "sun" to

buy plants. If a zombie reaches the house by any lane, the level is considered failed and the player will have to restart that level.

In this undertaking, the team of "**Những kẻ khốn khổ**" created the game Plants vs Zombies to bring players a version of Plants vs Zombies with many changes and improvements. The theme of the game is recreated based on the idea of a tower defense game with plants fighting against zombies. Along with the new idea, the basic rules of Plants vs Zombies remain the same, but the game rules have been refreshed and added some innovative features. New characters, such as the House Owner, were introduced, along with innovative features to enhance the gameplay experience. The game encourages players to observe, strategize, and train the factory's defenses to overcome various challenges. This is the product of an effort to provide players with an entertaining game with lots of engaging content.

Keywords: defense, zombie, creativity, observation, entertainment, game, object-oriented programming.

2. Introduction

In today's rapidly advancing Software Technology industry, a higher level of programming skill is becoming increasingly essential. As a result, the traditional Procedural-Oriented Programming approach is no longer sufficient to meet all requirements. This has led to the development of a new method called "Object-Oriented Programming," following Alan Kay's principles, to address these challenges.

This project was specifically designed using the Java language, incorporating Object-Oriented Programming. This approach has effectively resolved several issues that commonly arise when using the traditional Procedural-Oriented Method:

- The code becomes more transparent, easily understandable, and concise.
- The project represents a cohesive logical system, achieved by combining numerous related classes.
- Each class contains multiple methods that perform distinct behaviors unique to that class.

- Resources can be efficiently reused, enhancing overall efficiency.

The purpose of this project is to design a basic game by using OOP (Object-Oriented Programming) method. Therefore, our team decide to recreate a game name ‘Plant vs Zombie’ followed the four pillars of this measure. This project will go through the game overview and design, describing how the game is implemented and the programming functions and libraries used in the design.

Beside of the requirement of the course, our team also wants to create this project on account of learning and practicing the techniques throughout the semester. On the other hand, we considered it as a good opportunity to have a special mark in progress of the future career as a programmer. Therefore, the final version is the effort and hard-working of all team members without much interference of others people or available source codes from the internet.

3. Game overview (game description, user manual)

Plants vs. Zombies is an action strategy game where players must prevent enemy zombies from crossing their front lawn and entering the house. Players strategically place plants throughout the yard in an effort to stop the forward progress of the zombies.

3.1. Game description

3.1.1. Character

- Plant

  Plants in Plants vs. Zombies are the primary defenders against the invading hordes of zombies (*3.1.1 Character - Zombie*). They possess a variety of abilities and characteristics that aid in the protection of the player's garden. These plants are strategically placed on the game's grid-like lawn to counteract the different types of zombies encountered.

  In order to facilitate the planting of plants in underground tiles, it is necessary to verify the status of the tiles to determine if they are empty or occupied. If a tile (*3.1.2 Environment - Tile*) is found to be empty, the specific properties of the plants, such as their ID, damage potential, and sun consumption, are considered by using the function

"setPlantParameter". Subsequently, the program continues to assess the conditions, specifically checking if the current amount of available sunlight (*3.1.2 Items - Sun*) is greater than or equal to the sun consumption of the selected plants.

Not enough sun	
Enough sun	

Fig. 3.1.1 Character - plant (1)

The plants (Peashooter and Shadow-shooter) are equipped with an alert system, which activates when a zombie appears within the same row as a plant. This alert is triggered through the function "alertPlant". When a plant receives an alert, it initiates an attack on the zombie, while if the plant successfully defeats all the zombies in its row, it returns to its idle state. The transition between alert and idle states is facilitated by the functions "setPlantDangered" and "setPlantIdle" respectively, allowing plants to perform their respective tasks based on the current situation.

Idle state	
Alert state	

Fig. 3.1.1 Character - plant (2)

- Zombie

Zombies in Plants vs. Zombies are the primary adversaries in the game. They are depicted as reanimated corpses with the objective of invading the player's garden and devouring the plants. Understanding the different types of zombies and their behaviors is essential for developing effective strategies to repel their attacks.

The game is structured into waves (*6.5 Wave/Level*), with each wave presenting a new set of challenges. A wave consists of a fixed number of zombies that players must defeat to progress to the next level. The number and types of zombies in each wave vary, becoming progressively more difficult as the game advances.

Zombies are randomly spawned across the lawn at the beginning of each wave. The specific locations where zombies appear are determined by the game's mechanics, ensuring a diverse and dynamic gameplay experience. This randomness adds an element of unpredictability, requiring players to stay alert and adapt their defensive strategies accordingly.

Fig. 3.1.1 Character - zombie (1)

Towards the end of each wave, a horde of zombies is generated. This horde comprises multiple zombie types and has a predefined total number of zombies. These zombies appear simultaneously across different locations on the lawn within a specific period of time. The

sudden influx of zombies during the horde adds intensity and challenges players to quickly and effectively defend their garden.

Fig. 3.1.1 Character - zombie (2)

- Crazy Dave the house owner

Crazy Dave, a beloved character in the Plants vs. Zombies game series, is typically portrayed as a non-playable character (NPC) who guides players through the game. However, in this modified version, Crazy Dave takes on a unique role as a playable character with shooting abilities, adding a delightful twist to the gameplay experience.

Players actively participate by guiding Crazy Dave's movement through interactive clicks. By clicking on different locations within the game's environment, players can dictate Crazy Dave's path, enabling him to navigate the battlefield strategically. This interactive movement feature adds a layer of engagement and control, empowering players to position Crazy Dave optimally to maximize his shooting effectiveness.

Fig. 3.1.1 Character - Crazy Dave the house owner (1)

As players control Crazy Dave, his shooting abilities activate when zombies come within a specific range. This proximity-based mechanism ensures that Crazy Dave focuses his firepower (*3.1.2 Items - Projectile*) on the imminent threat, eliminating zombies in close proximity to him. This adds an element of timing and precision to the gameplay, as players must strategically position Crazy Dave to optimize his devastating attacks.

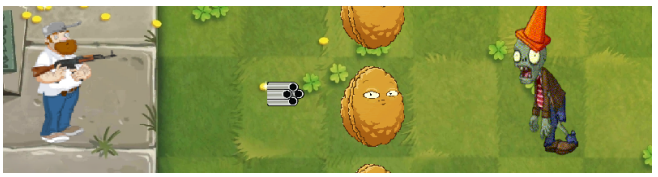
No zombie in range	
A zombie in range	

Fig. 3.1.1 Character - Crazy Dave the house owner (2)

3.1.2. Items

- Sun

The sun plays a crucial role as a resource that players must manage effectively to grow and sustain their plants. Sunlight is

represented by small sun icons that periodically fall from the sky or are generated by certain plants.

Sun acts as the primary resource used to purchase and deploy plants. Each plant has a designated sun cost associated with it, indicating the amount of sun required to plant it. Players must accumulate enough sun to strategically select and place plants in their garden to defend against the encroaching zombie horde.

Throughout the game, sun icons randomly appear on the game's lawn or are produced by specific sun-producing plants. These plants include Sunflowers, Twin Sunflowers, Sun-shrooms, and others. When players click on a sun icon, it is collected and added to their available sun count.

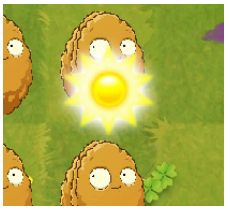
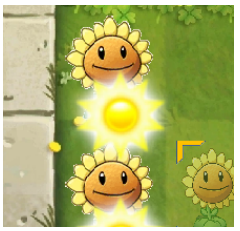
Sun from the lawn	
Sun from the sun-producing plant	

Fig. 3.1.2. Item - sun

- Projectile

The peashooter fires small green peas as its primary projectiles. These peas act as ammunition to damage and eliminate zombies. Each peashooter plant continuously produces and launches peas in a straight line towards the oncoming zombie wave.

Projectiles fired by shootable plants travel at a moderate speed, allowing them to reach their intended targets within a reasonable timeframe. The range of the plant's projectiles extends across a fixed distance in front of the shooter. This range determines the maximum distance the peas can travel before disappearing.

Fig. 3.1.2 Item - projectile

Similar to other shooting plants in the game, Crazy Dave's projectiles typically travel in a straight line (*Fig. 3.1.1 Characters - Crazy Dave the house owner (2)*). They are launched from his weapon and move forward until they hit a target or reach a certain range.

3.1.3. Environment

- Tile

The tile refers to the individual units or squares on the game's playing field or lawn where players can plant their defensive plants. The playing field in game is divided into a grid system, with each grid representing a tile. The grid is typically organized in rows and columns, creating a matrix of tiles where players can strategically position their plants.

Players can place their plants on the available tiles within the grid. Each tile can accommodate a single plant, and the placement of plants is subject to the player's strategic decisions. Proper placement is

essential for maximizing plant effectiveness, creating defensive formations, and countering the approaching zombie horde.

Similar to the plants, Crazy Dave moves across the playing field by occupying and traversing the individual tiles on the grid. Each tile represents a distinct position that Crazy Dave can occupy at any given time. He moves from one tile to another, allowing players to strategically position him in different areas of the lawn.

Fig. 3.1.3 Environment - tile

- Selection frame

The game interface includes a selection frame for tiles and a plant bar. Both can be used be the mouse controller or keyboard controller (*3.2 User manual*).

The selection frame of tiles refers to the highlighted area or border that appears around a tile when it is selected or targeted. This frame helps players identify and interact with specific tiles on the grid.

Bar's selection frame	
Tile's selection frame	

Fig 3.1.3 Environment - selection frame

- Plant bar

The plant bar is a user interface element that displays the available plant options for players to choose from during gameplay.

The primary function of the plant bar is to allow players to select and choose plants for placement on the grid. It typically appears at the top or side of the screen and consists of a row or column of plant icons representing different plant types.

Fig. 3.1.3 Environment - plant bar

3.2 User manual

3.2.1 Mouse controller

Playing Plants vs. Zombies with a mouse is relatively straightforward. The general overview of how player can control the game using a mouse is listed below:

- Menu Navigation: Player can navigate through the game's menus using the mouse. Move the cursor over the desired option and click to select it.
- Plant Selection: To select a plant, move the cursor over the plant icon located on the right side of the screen. Click on the desired plant to pick it up.
- Plant Placement: Once player has selected a plant, move the cursor to the desired location on the grid. Left-click to place the selected plant in that spot.
- Sun Collection: Sunlight, which is the in-game currency, falls from the sky during the day. Move the cursor over the sun to collect it. The collected sunlight can be used to purchase and place plants.

- Crazy Dave movement: To move the houseowner at a desirable position, the player can perform by clicking at the corresponding row.

3.2.2 Keyboard controller

Similar to Mouse controller, keyboard controller can help the player control the game.

- Plant Selection: using arrow buttons to move in plantBar (left arrow to move left and right arrow to move right); possible object can be selected in plantBar: 5 plants and shovel, navigate to the desired plant or shovel.
- Plant Placement: using W A S D for moving up, left, down and right between tiles, respectively; there are total of 45 tiles that can be selected by this way. Navigate the tile selection to the desired tile. Once the plant or shovel in plantBar and tile are selected, press enter to perform action.
- Sun Collection: player can use W A S D to move the selection frame. The sun will be collected automatically whenever the selection frame moves to the tile containing the sun.

4. UML diagram

4.1 Overview

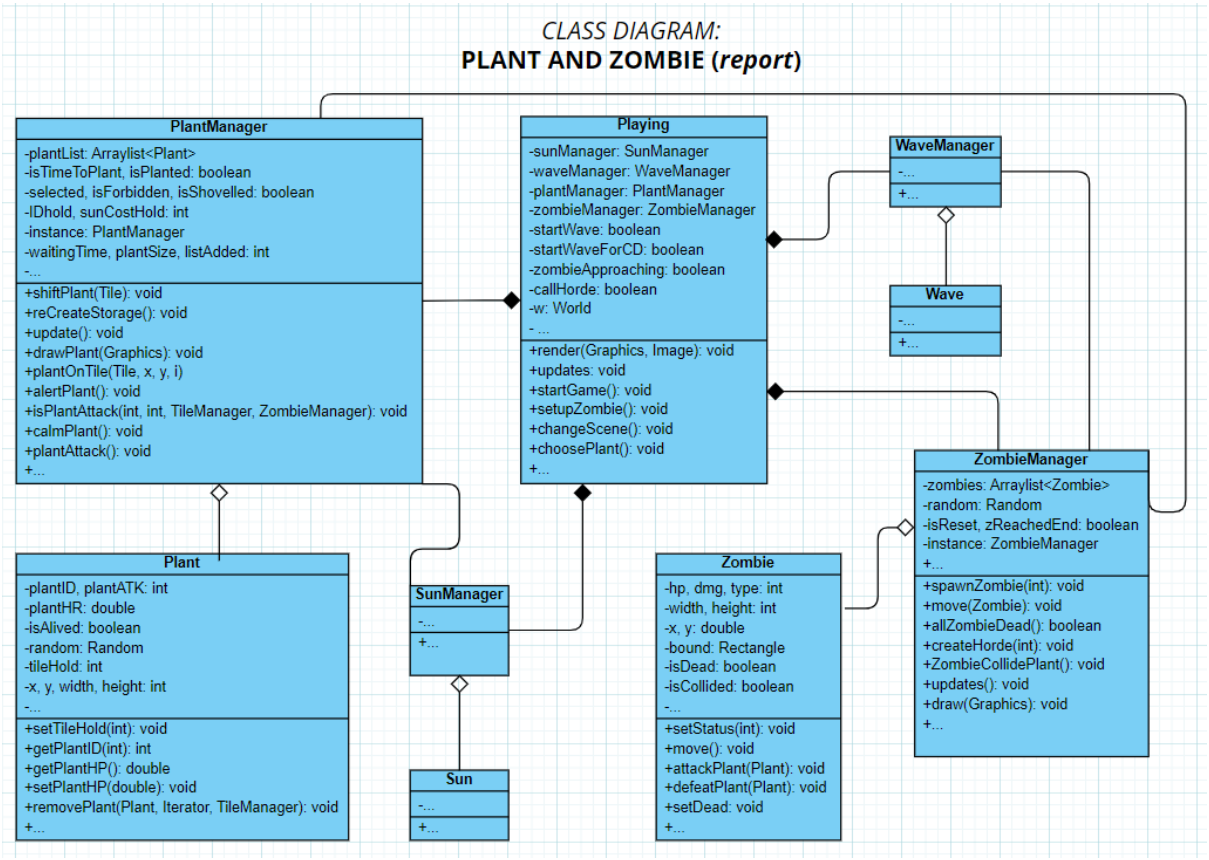


Fig. 4.1 Plant and zombie class diagram (report)

4.2 Plant

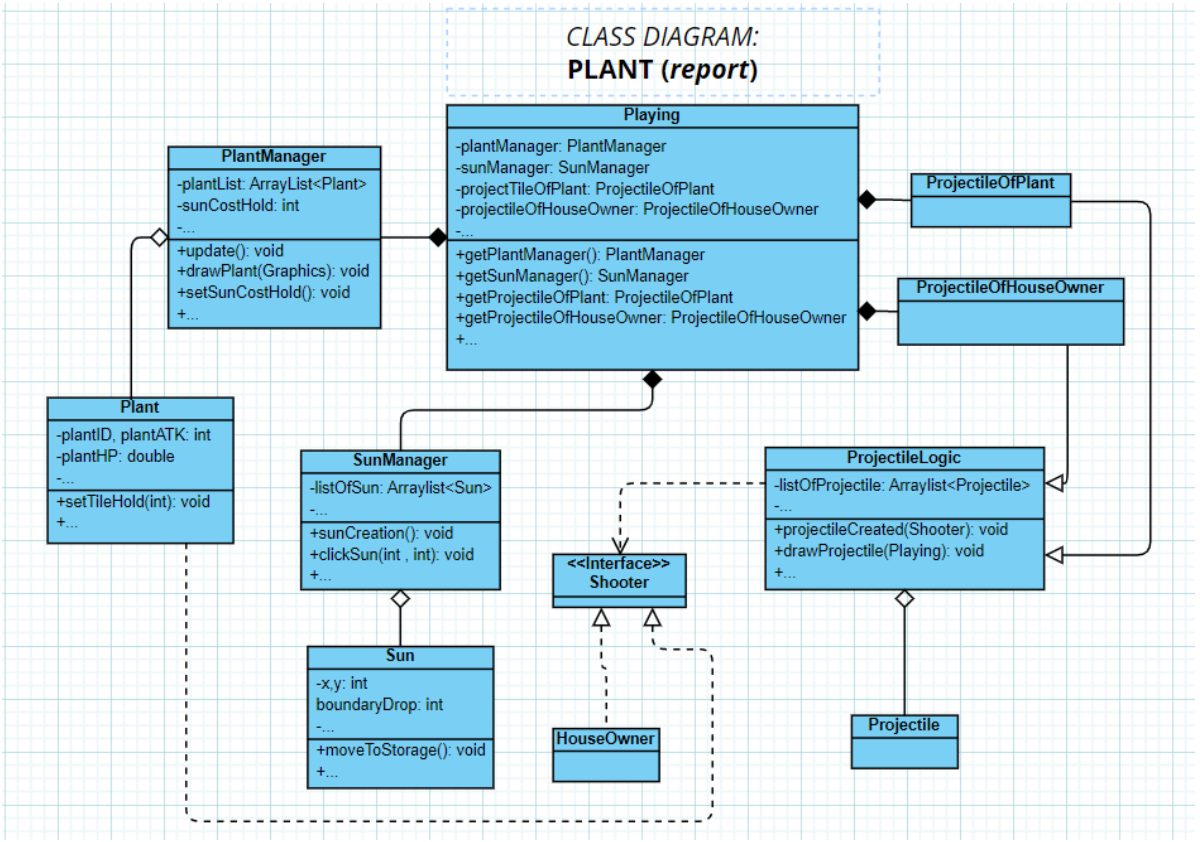


Fig. 4.2 Plant class diagram (report)

4.3 Zombie

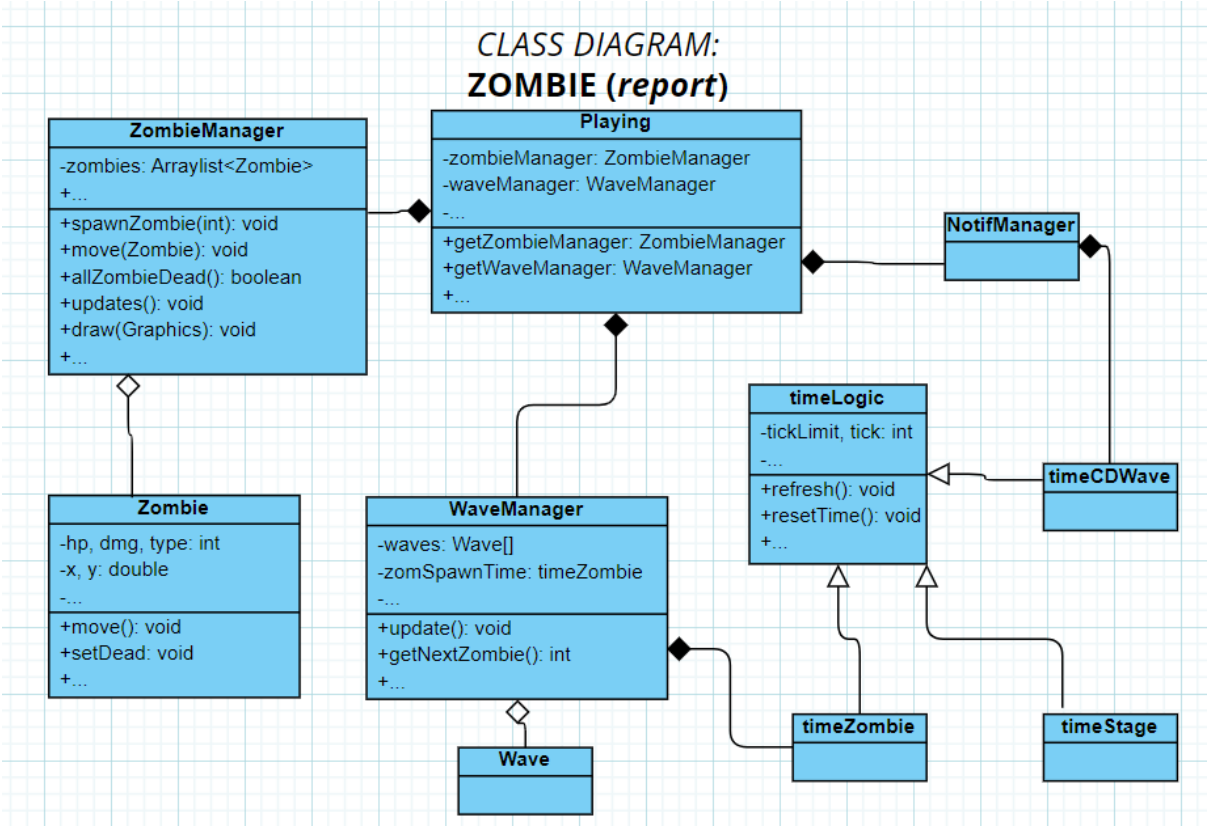


Fig. 4.3 Zombie class diagram (report)

5. Methodology

Before participating in the recent project, our team conducted research on the Object-Oriented Programming (OOP) methodology and its basic properties. However, we acknowledged that a thorough understanding of OOP alone would not be sufficient.

Our team also needed to address the challenge of collaborative coding. As a solution, we turned to Github, one of the most valuable tools for software developer teams. We primarily coded the game using Intellij IDEA and utilized Github as a code hosting platform for version control and collaboration. This enabled us to organize and manage the project more efficiently.

One month prior to starting the project, we conducted research on related games and followed how-to tutorials from various sources. Simultaneously, we engaged in practical exercises focused on OOP concepts and applied our knowledge using the Github platform.

6. Game design

6.1 Objects

Objects play the most important role, bringing life and soul to this game. There are various types of objects in this game, and each type of object play a specific role. Various objects in this game are used to create UI, thus making it interact-able by players. Therefore, players can use this opportunity to make many special formations, strategies or even performing experiments between objects

In addition, many objects have the ability to interact with other objects as well, making the game more consistent in terms of game logic. Plants, zombies, projectiles, bar, and buttons are some typical objects of the game. Each of them will have different interactions with each other or with players in the game:

- Plants will use projectile to attack zombies
- Zombies will continuously move and attack plant if needed
- Bar and Buttons help players to interact with the game

Fig. 6.1 Plants using projectiles to attack zombies (upper image) and zombies move and attack plants (lower image)

6.2 Game scenes

Scenes are the collection of atmospheres in the game

Similar to other games, scenes in this game are also very important, as they can navigate players to make correct decisions based on each scene. Not only that, they can also make the environment of the game become more colorful and lively.

Scenes also provide User Interface, allowing players to interact with them. Many UI are useful since they help players have a better experience of the game. There are total of 5 scenes in this game:

- Menu: initial scene when players turn on the game. It contains 2 buttons which allow players to play or quit the game.
- Playing: The main scenes of the game where players will interacts with every main objects and contents.
- Setting: the scene can be found when players press “Setting” button. Not only it can temporarily stop the game, it also provides various functions such as Exit the game.

- Winning, Losing: the scenes which tell players the game has finished. When the conditions are met, either winning or losing scene will appear, indicating that players have won (or lost) the game.

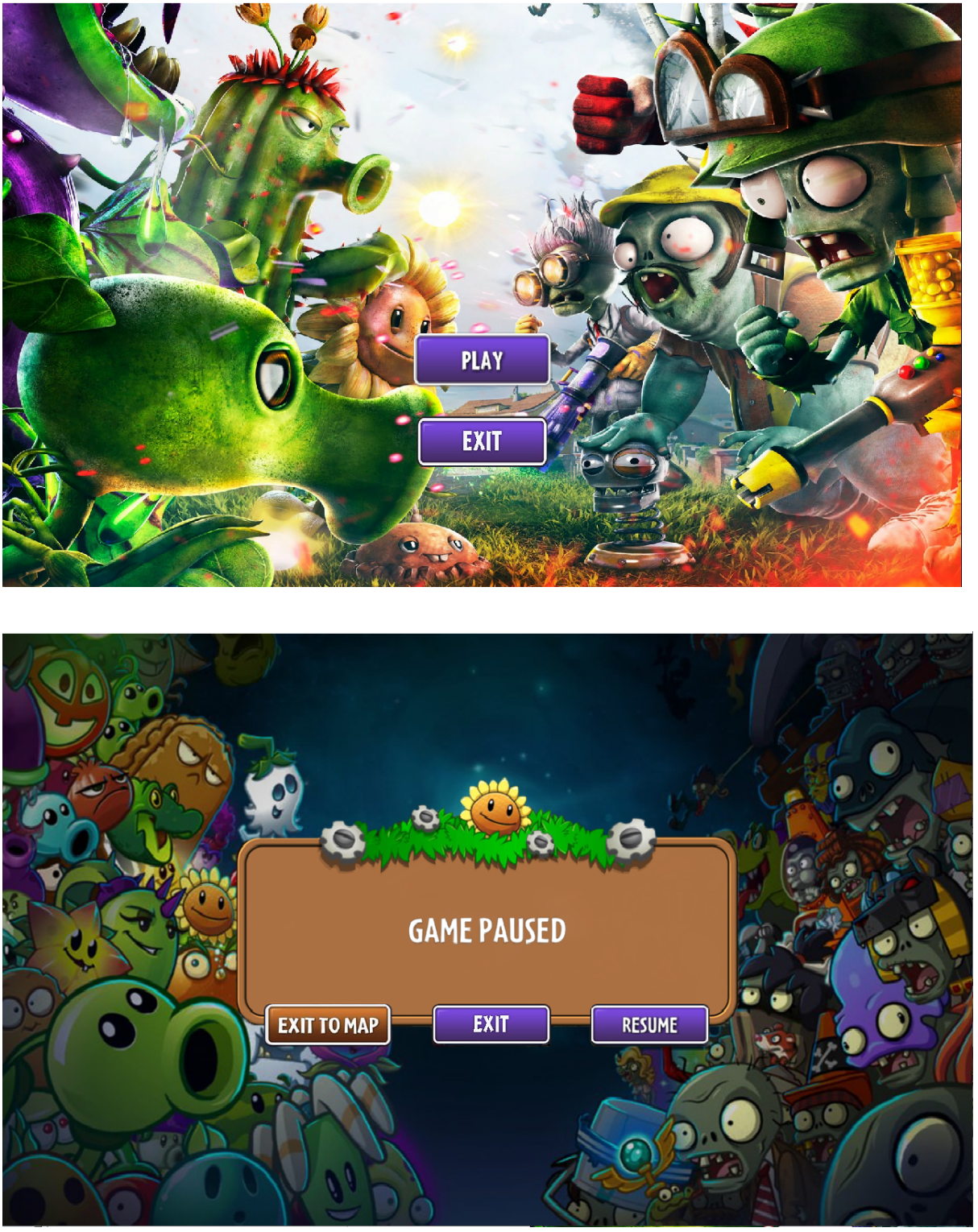


Fig. 6.2 Menu scene (Upper image) and Setting scene (Lower image)

6.3 Animation

At first, when making the game, the objects themselves did not have any animation, which makes the game look boring and unattractive. Therefore, in order to fix it, animation has come to live.

Using the definition of animation, the animations in this game are created in one of the most traditional yet effective ways: by fast-forwarding many picture frames of an animation, the objects are given a view that looks extremely like an animation.

Animations in this game originally come from many sources, but most of it comes from wiki-fandom or from the original game. The 2 types of objects use animation are plants and zombies:

- Plant: attack and idle animations
- Zombie: attack and move animation

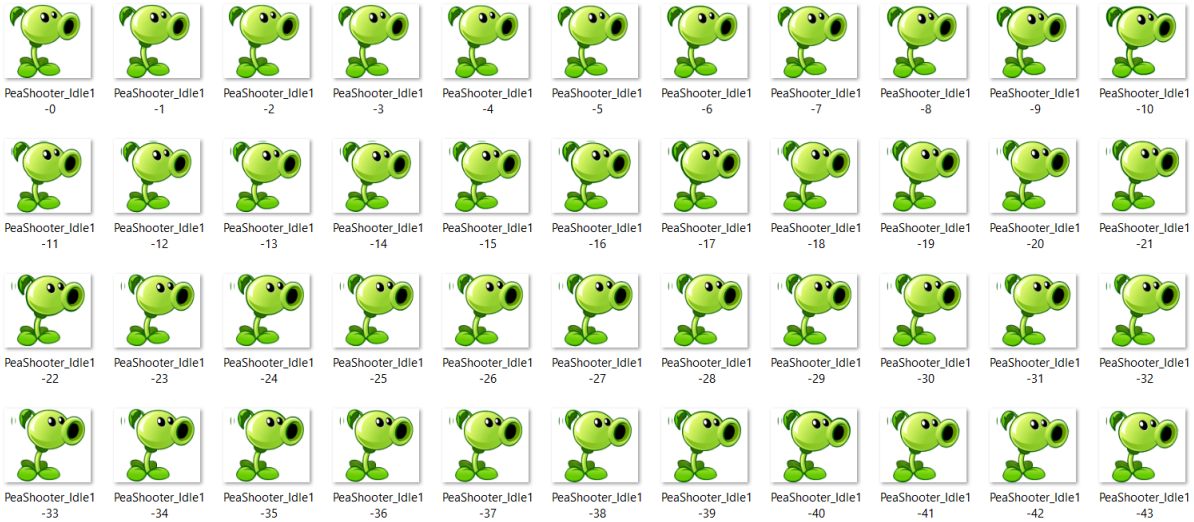
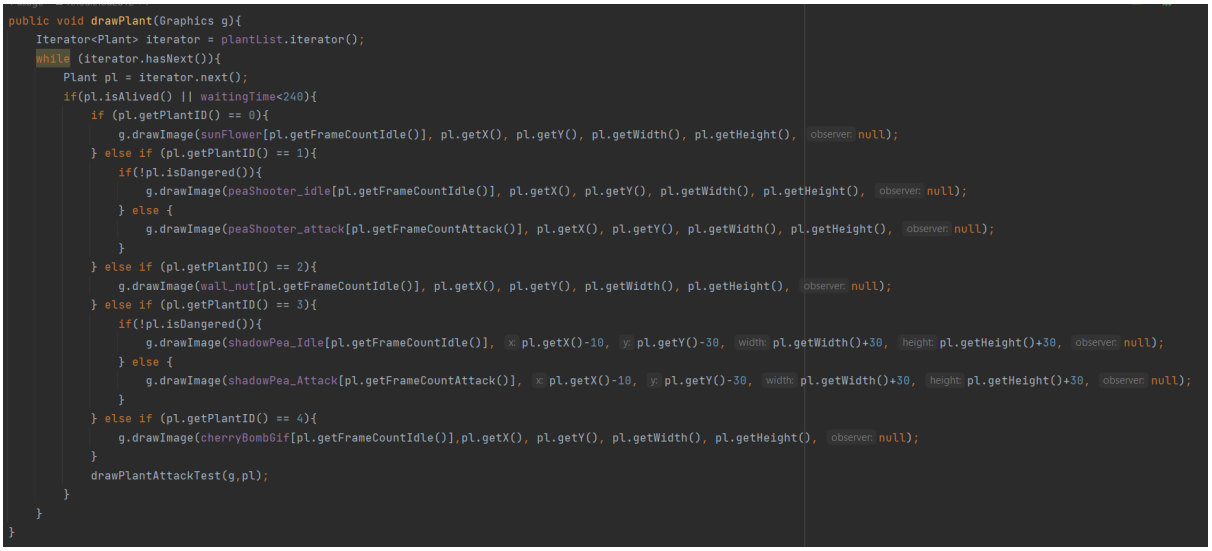


Fig. 6.3.1 An example for a list of frame pictures to create animation for peashooter idle state

```
public void drawPlantAttackTest(Graphics g, Plant pl){
    if(pl.getPlantID() == 1 && waitingTime <240){
        g.drawImage(peaShooter_attack[pl.getFrameCountAttack()], pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(), observer: null);
    } else if(pl.getPlantID() == 3 && waitingTime<240){
        g.drawImage(shadowPea_Attack[pl.getFrameCountAttack()], x: pl.getX()-10, y: pl.getY()-30, width: pl.getWidth()+30, height: pl.getHeight()+30, observer: null);
    }
}
```



```
public void drawPlant(Graphics g){
    Iterator<Plant> iterator = plantList.iterator();
    while (iterator.hasNext()){
        Plant pl = iterator.next();
        if(pl.isAlived() || waitingTime<240){
            if (pl.getPlantID() == 0){
                g.drawImage(sunFlower[pl.getFrameCountIdle()], pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(),  observer: null);
            } else if (pl.getPlantID() == 1){
                if(!pl.isDangered()){
                    g.drawImage(peaShooter_idle[pl.getFrameCountIdle()], pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(),  observer: null);
                } else {
                    g.drawImage(peaShooter_attack[pl.getFrameCountAttack()], pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(),  observer: null);
                }
            } else if (pl.getPlantID() == 2){
                g.drawImage(wall_nut[pl.getFrameCountIdle()], pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(),  observer: null);
            } else if (pl.getPlantID() == 3){
                if(!pl.isDangered()){
                    g.drawImage(shadowPea_Idle[pl.getFrameCountIdle()],  x: pl.getX()-10,  y: pl.getY()-30,  width: pl.getWidth()+30,  height: pl.getHeight()+30,  observer: null);
                } else {
                    g.drawImage(shadowPea_Attack[pl.getFrameCountAttack()],  x: pl.getX()-10,  y: pl.getY()-30,  width: pl.getWidth()+30,  height: pl.getHeight()+30,  observer: null);
                }
            } else if (pl.getPlantID() == 4){
                g.drawImage(cherryBombGif[pl.getFrameCountIdle()],pl.getX(), pl.getY(), pl.getWidth(), pl.getHeight(),  observer: null);
            }
            drawPlantAttackTest(g,pl);
        }
    }
}
```

Fig 6.3.2 Methods to import every frame pictures to an array (upper image) and draw plants based on plant's current frame position (lower image)

In the code example, we first import all frame pictures of an action of an object into an array with the length equals to the number of pictures. Then for each frame, the draw method will draw a frame picture of each plant, and by the end of the game loop, the frame position will increase by 1, and in the next loop the draw method will draw the next frame position

6.4 Audio

Even if audio is not a must in this game, this feature still has a very significant role in order to bring to players a better experience. Without audio or sound, the game will become boring and players no longer have the interest to play the game.

Many objects of this game have their own audio, thus each audio will have its own unique properties. Each audio also brings to players different emotions and feelings, such as: calm, happy or intense feeling.

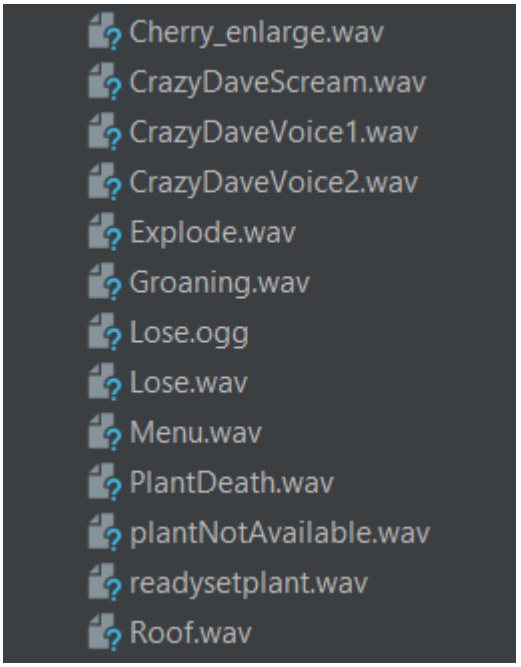


Fig. 6.4 Several audios used in this game

6.5 Notification

Notifications in the game serve as important communication tools to inform players about various events, updates, and game-related information. The generation and display of notifications primarily rely on the functionality provided by the Time Logic system and its subsequent descendants. By leveraging this system. Here are some key aspects of notifications in a game like Plants vs. Zombies.

Wave Announcements: Notifications can alert players when a new wave of zombies is approaching or when a particularly challenging wave is about to begin. This helps players prepare and strategize their defense accordingly.

Objective Updates: Notifications can provide updates on the current objectives or goals of the player. For example, they may remind players to protect the garden from a certain number of waves or to collect a specific number of sun points.

Current wave notification	

Countdown wave notification	
State clear notification	

Fig. 6.5 Notifications

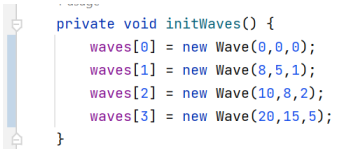
6.6 Wave/Level

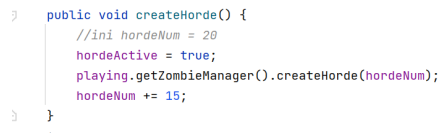
Waves or levels represent distinct stages of gameplay where players face increasing challenges and progress through the game's storyline. Here is an overview of waves/levels in the game:

The wave itself, along with the assistance of the 'WaveManager', governs the composition and quantity of zombies that will appear in the current level or wave. The 'WaveManager' system is responsible for determining the specific types and numbers of zombies that will be spawned during gameplay.

In the game, waves of zombies consist of two distinct moments. The first moment involves individual zombies walking towards their target, typically the player's defenses. These zombies move independently, advancing at a steady pace to reach their destination.

The second moment occurs when a zombie horde is unleashed. During this intense phase, a large group of zombies appears simultaneously, presenting a more formidable challenge for the player. The horde of zombies advances together, creating a heightened sense of urgency and requiring swift action from the player to defend against the onslaught.

Wave inital		Wave 1: 8 normal zombies, 6 conehead zombies, 1 special zombie. Wave 2: 10 normal zombie,

		8 conehead zombies, 2 special zombies. Wave 3: 20 normal zombie, 15 conehead zombies, 5 special zombies.
Horde (appear at the final moment of each wave)		Wave 1: 20 random zombies Wave 2: 35 random zombies Wave 3: 50 random zombies

7. Conclusion

The game is still being developed. After completing a game with some novel features compared to the original version, the team has a more thorough understanding of the four features of OOP, the SOLID principle, and the Design pattern principle, which helps the team be fluent in OOP of game development also in the programming process. Thus, Plants vs Zombies is developed rigorously using the basic idea of OOP, and the game code contains all four major OOP features (encapsulation, inheritance, abstraction, and polymorphism), the SOLID principle and a design pattern learned from the classroom.

8. Future works

Unfortunately, the team was hoping to develop even more levels with different levels and some other plants for players. However, due to a lack of time and the need to spend more time resolving some bugs, this group has not yet concluded the initial wish for development. Accordingly, in the future, some new plants and natural conditions such as day or night will be included to add more resources to the player and evaluate the player's management skills. Furthermore, some of the player interactions with HouseOwner using the keyboard will also be implemented in future updates. Therefore, any new commits are appreciated.

9. Acknowledgment

We want to express our sincerest thanks to our lecturers and the people who have helped us to achieve this project's goals:

Dr. Tran Thanh Tung

MSc. Pham Quoc Son Lam

10. References

About Github:

https://www.youtube.com/watch?v=3uq3VCT5V2g&t=334s&pp=ygUPZ2l0aHViIGludGVsbGlq

https://www.youtube.com/watch?v=mM_drNdss4c&t=1s

About Code:

[Tower defense: Wave mechanism - Game loop]
https://www.youtube.com/playlist?list=PL4rzdwizLaxb0-TajNIp5DOoT_PAxhx0T

[Chess: Tile]

https://www.youtube.com/watch?v=h8fSdSUKttk&list=PLOJzCFLZdG4zk5d-1_ah2B4kqZSeIlWtt&index=1

[Game loop]

https://www.youtube.com/watch?v=lW6ZtvQVzyg&t=191s&pp=ygUQZ2FtZWxvb3AgaW4gamF2YQ%3D%3D

[GUI]

https://www.youtube.com/watch?v=7GaAW-DdPuI&pp=ygUOamZyYW1lIGluIGphdmE%3D

[Mouse listener]

https://www.youtube.com/watch?v=jptf1Wd_omw&t=197s&pp=ygUdbW91c2VsaXN0ZW5lciBhbmQga2V5bGlzdGVuZXI%3D

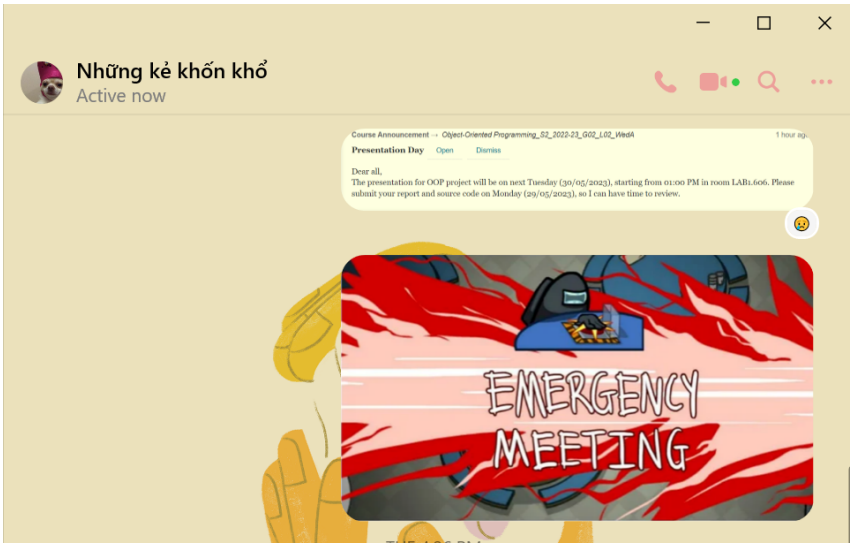
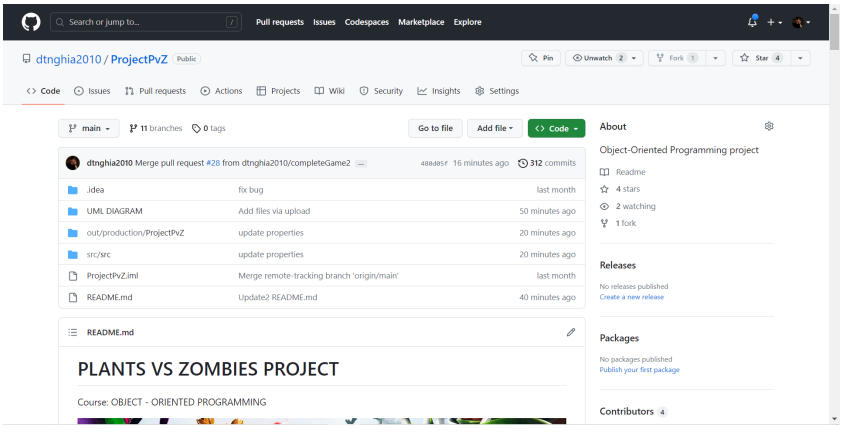
[SOLID principles]

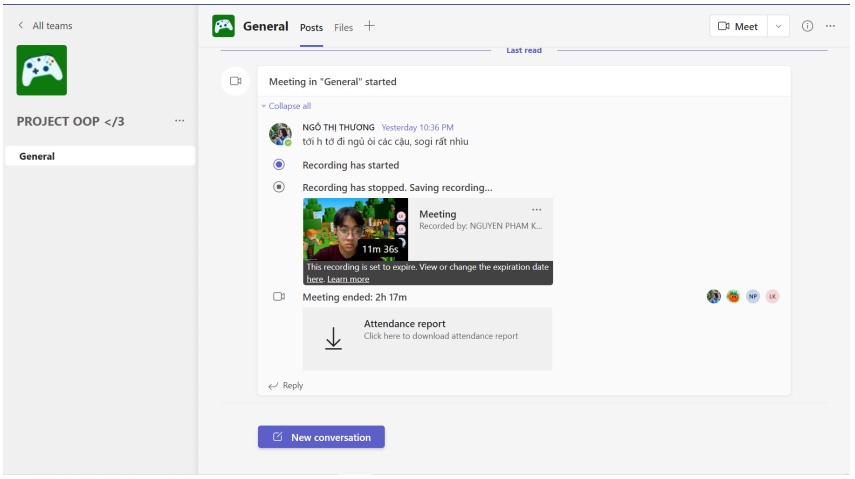
https://www.youtube.com/watch?v=_jDNAf3CzeY&pp=ygURY29kZSBnYW1lIGluIGphdmE%3D

11. Our Github project

https://github.com/dtnghia2010/ProjectPvZ

12. Workplace

Group chat	
Github repository	

MS Teams	 All teams PROJECT OOP </3 General General Posts Files Meet Last read Meeting in "General" started Collapse all NGÔ THỊ THƯƠNG Yesterday 10:36 PM tới h tớ đi ngủ ói các cậu, sogi rất nhiu Recording has started Recording has stopped. Saving recording... 11m 36s Meeting Recorded by: NGUYEN PHAM K... This recording is set to expire. View or change the expiration date here. Learn more Meeting ended: 2h 17m Attendance report Click here to download attendance report Reply New conversation