

**Report**

Last Survivor: Nakhon Si Thammarat

**Prepared by**

Yanason Saksritrakun 6504062663095

**Present to**

Asst.Prof. Sathit Prasomphan

**Object Oriented Programming subject**

Term 1/2023

**Introduction**

Project name: Last Survivor: Nakhon Si Thammarat

Prepared by: Yanason Saksritrakun

Instructor: Asst.Prof. Sathit Prasomphan

**Chapter 1: Introduction and Significance**

This project aimed to assess proficiency in Object-Oriented Programming through the application of acquired concepts to develop a 2D game.

Project type:

Game

Project Benefits:

1. Enjoyment and Entertainment
2. Enhances Quick Decision-Making Skills

Table of Work Plan (October-November)

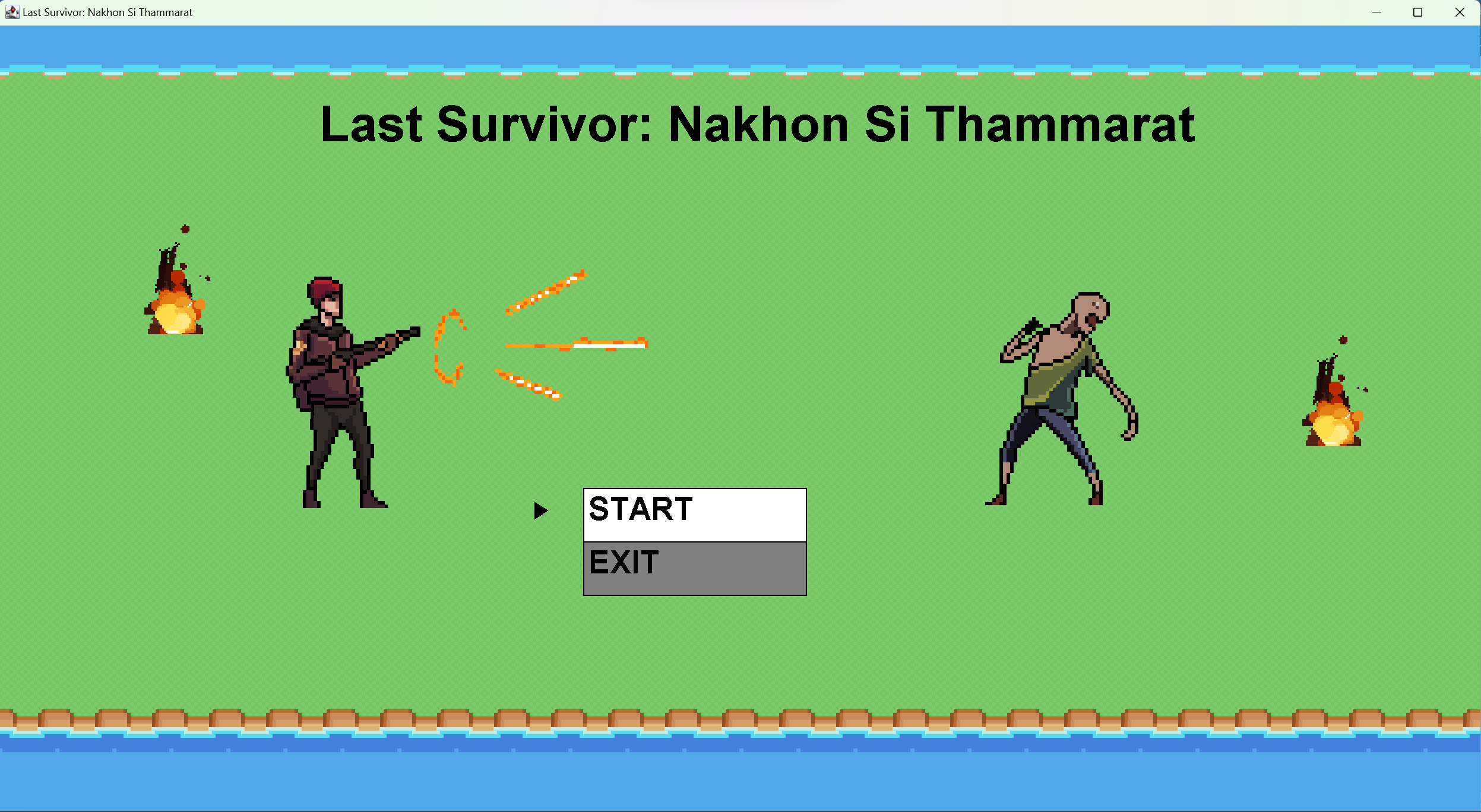
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Activity | 17-24 | 25-1 | 2-8 |
| 1 | Find Pictures, Graphics, or  Designs in the Game |  |  |  |
| 2 | Study Various Relevant  Information |  |  |  |
| 3 | Start Writing a Program |  |  |  |
| 4 | Check for Errors |  |  |  |
| 5 | Prepare Documents |  |  |  |

**Chapter** 2 **: Development**

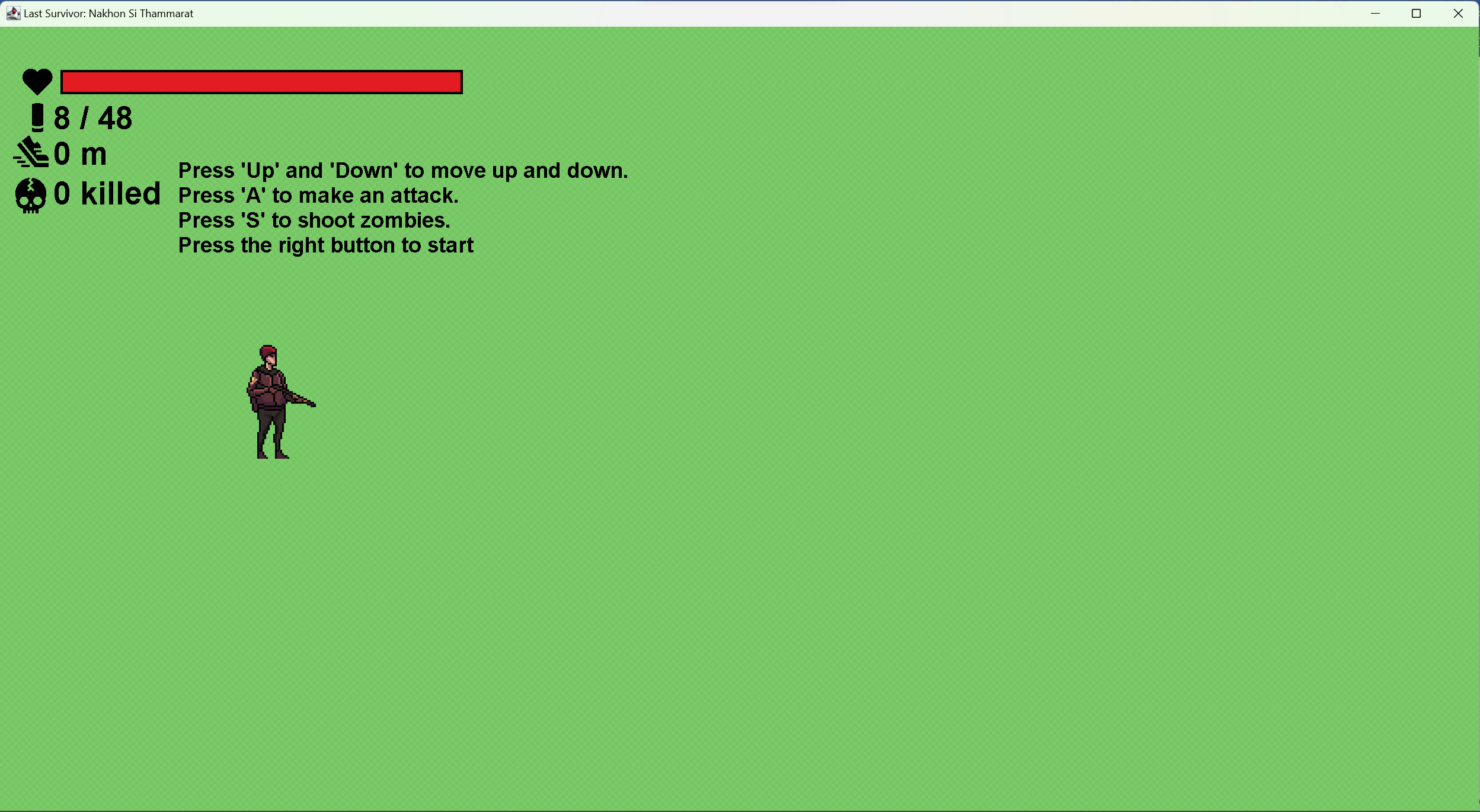
Story:

In a desolate world, "Film" stands as the lone survivor on Earth. Prepared to navigate the chaotic horde of zombies in the heart of Nakhon Si Thammarat, survival is the only goal.

How to play?



Select “START” using up&down button and press spacebar to play



Press “Up” and “Down” to move.

Press “A” to attack zombies.

Press “S” to shoot zombies.

Press the “Right” button to start running.

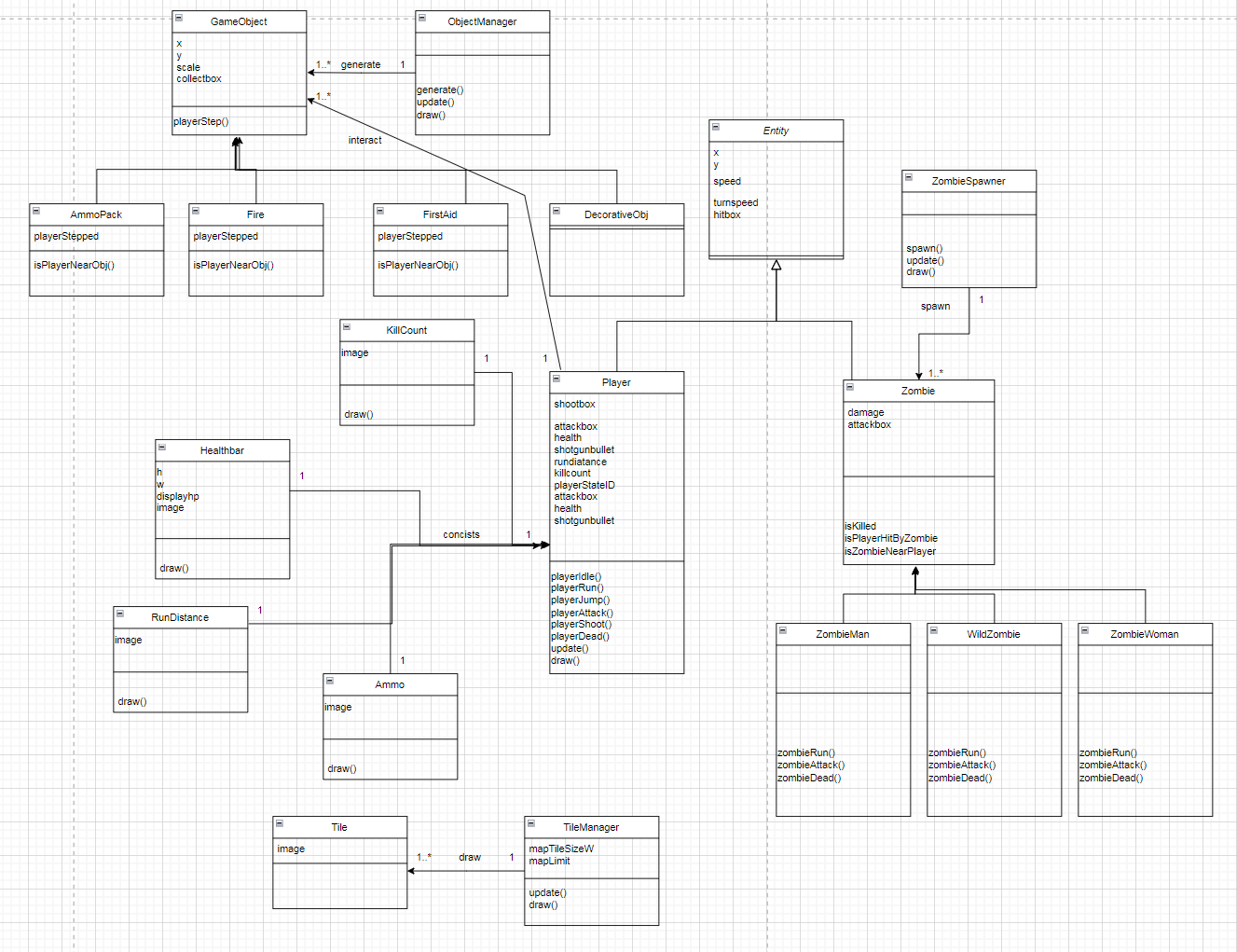


* Try to avoid the zombies and fire or you will be attacked.　If you can't avoid it, shoot them.
* Each zombie has a different speed, be careful.
* Run as far as you can.



After game over, choose to restart or exit.

Class diagram



A solitary player embarks on the journey, armed with a health rating, attack range, and capacity to withstand damage. The arsenal includes shotgun shells for interacting with objects—

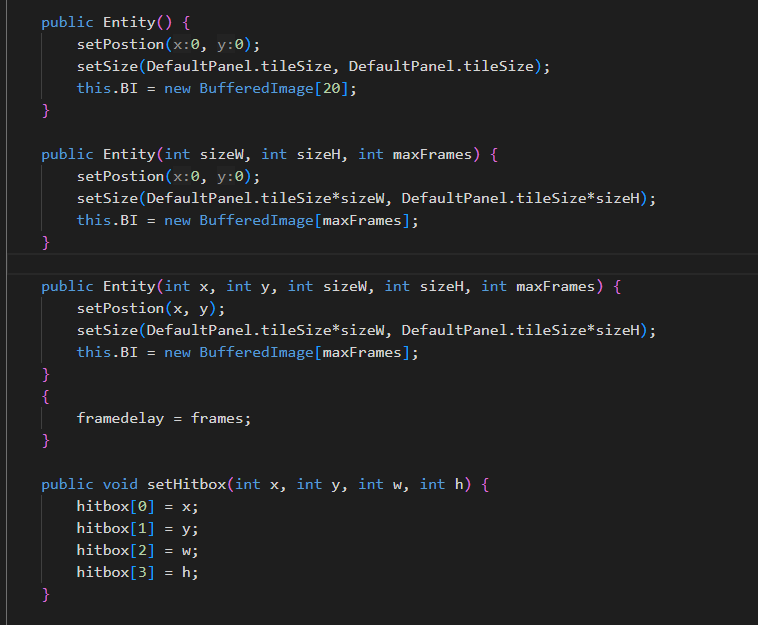
taking damage from fire, collecting items. Among the diverse objects are decorations, lights, first aid kits, and ammunition. The player faces various zombie adversaries, including zombies men, wild zombies, and zombie women

OOP

Constructor

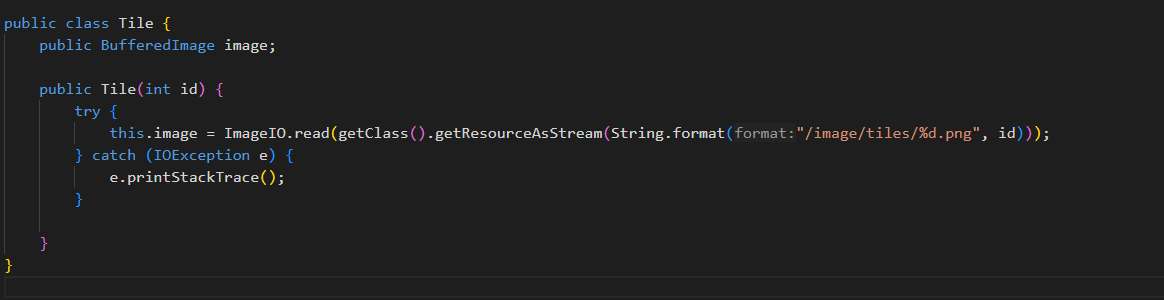
Entity class (for player, zombie)

Set size and frames (sprite sheet)



Tile (map image)

Pass id for setting image



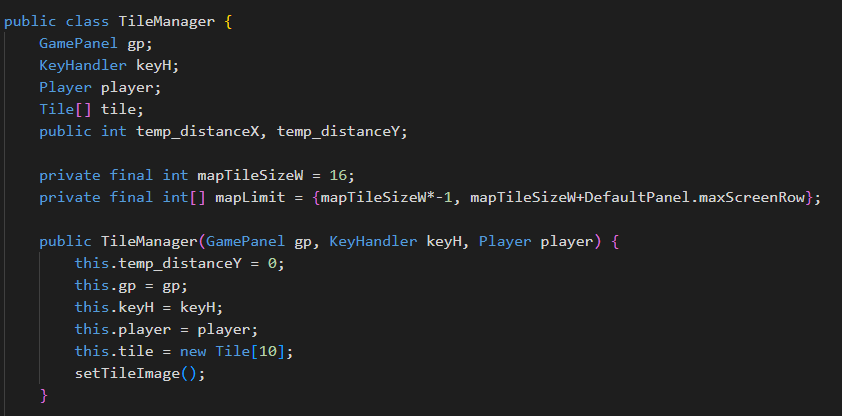
TileManager (Draw map)

Pass object Player, GamePanel, KeyHandler

Get gameStateID from class GamePanel.

KeyHandler uses for move player by camera view.

And get player speed.



ObjectManager (generate object such as fire, items)

Pass object Player, TileManager

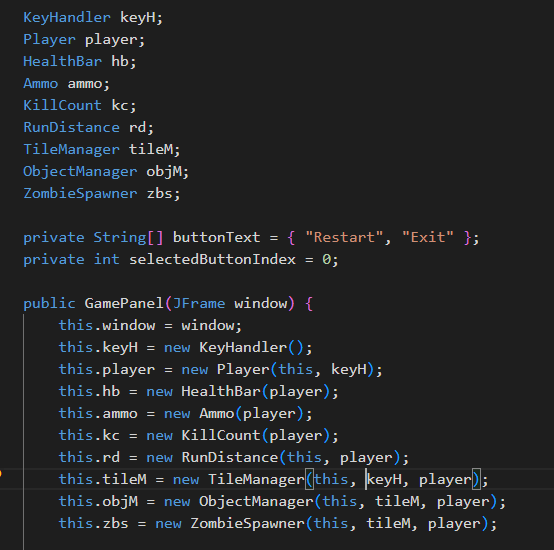
Get player coordinates from TileManager (coordinates by camera view)

And get player x, y to check if item is collected by player.



GamePanel (Draw all objects)

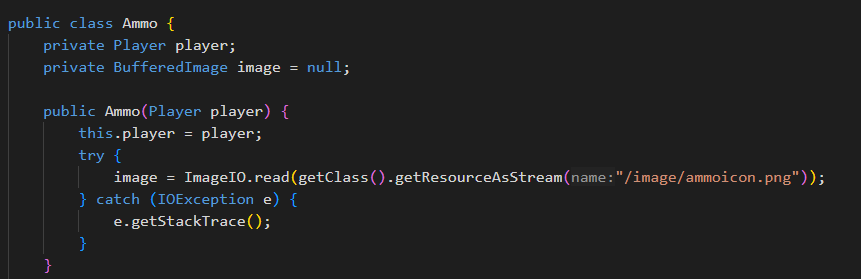
Pass JFrame to swap a panel to another panel.

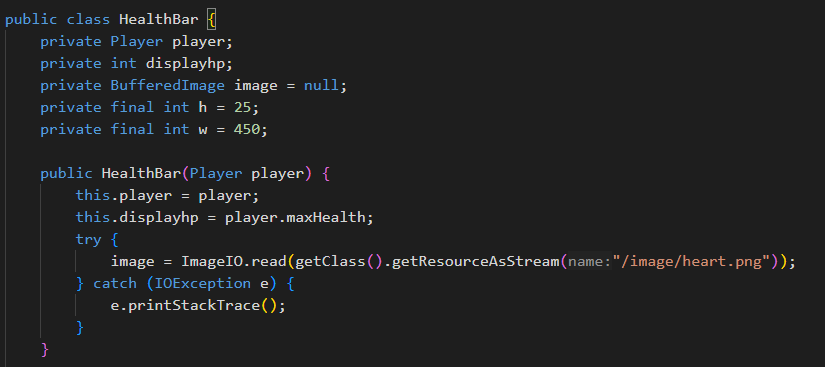


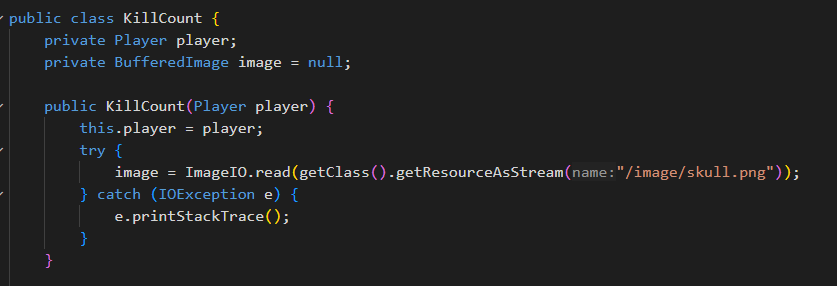
Player stat class

Get some attribute from player, set image and draw.





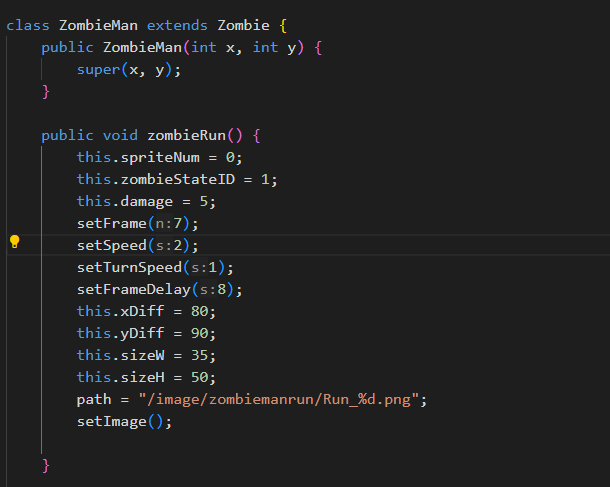


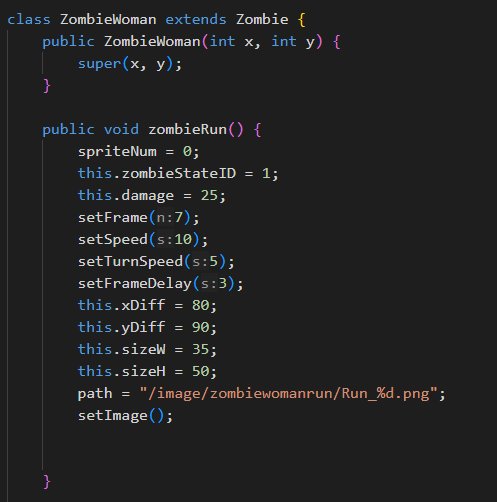


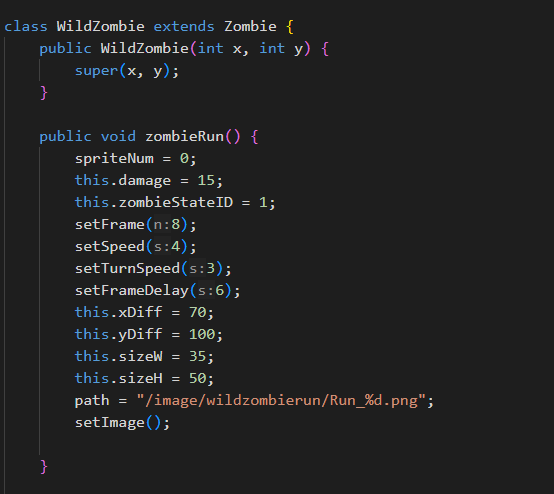
Abstract zombie class that inherits from Entity class



Type of zombie

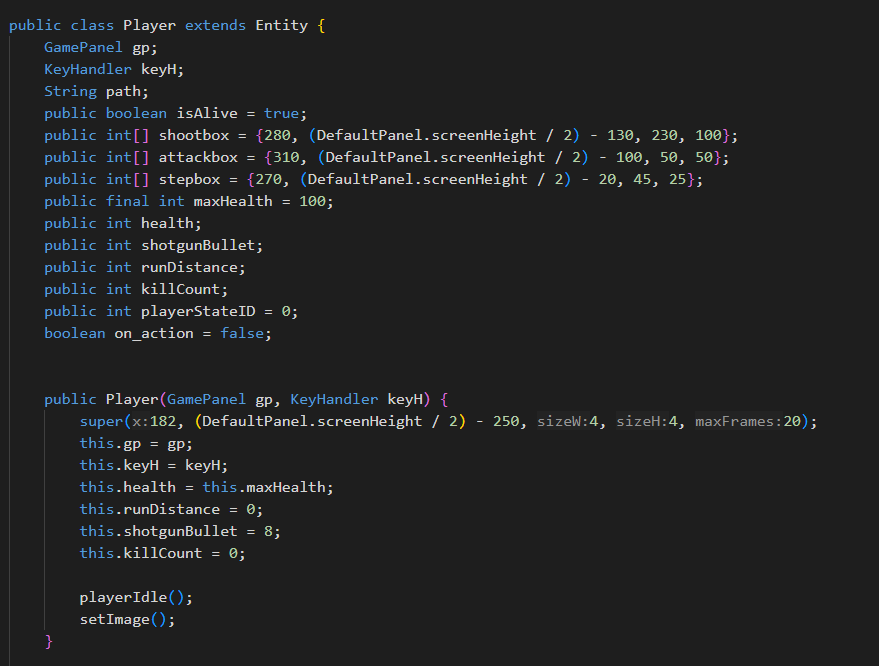




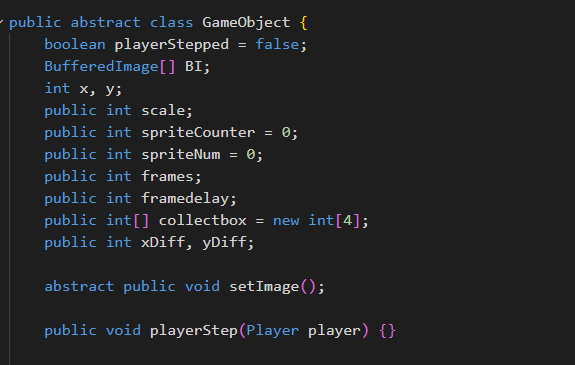


Each zombie has a different speed, turn speed, damage, and frames (for animation)

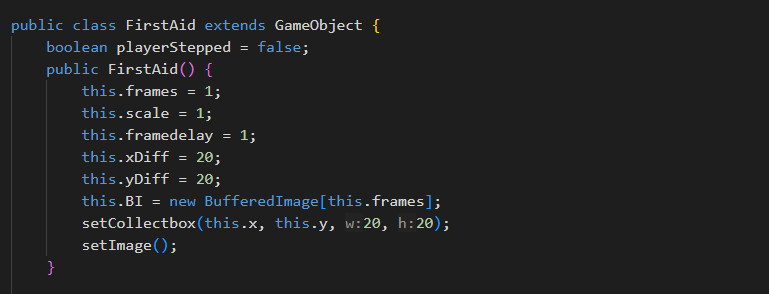
Player is a type of Entity too

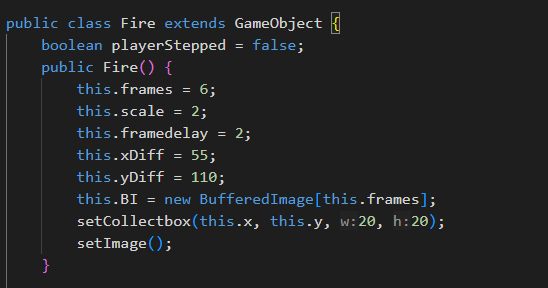


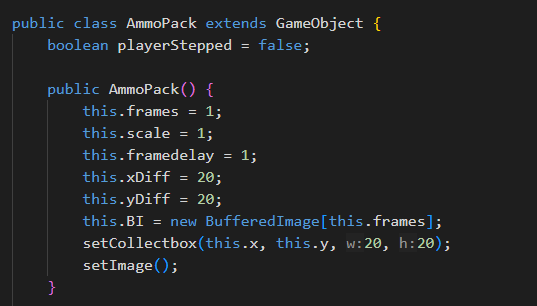
GameObject (Super abstract class)

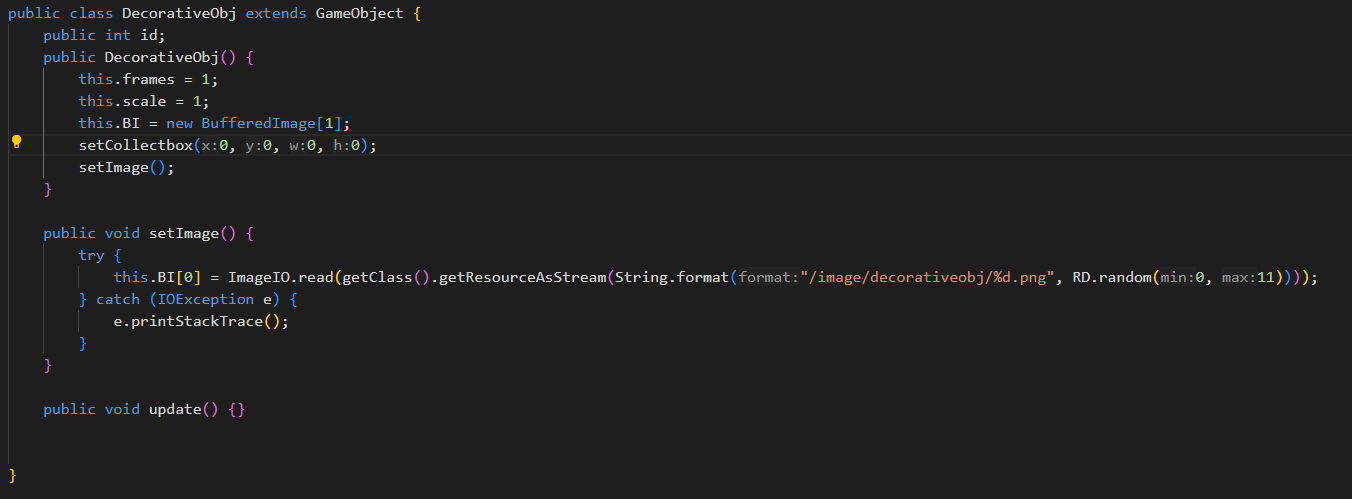


Sub class of GameObject

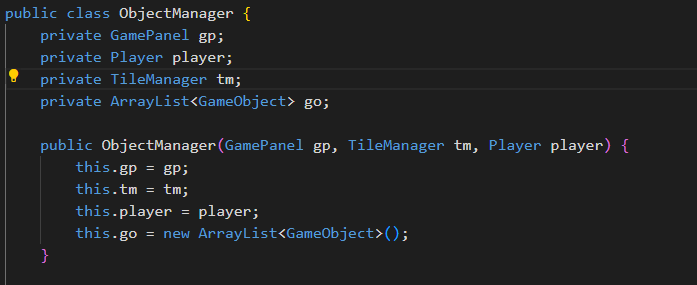








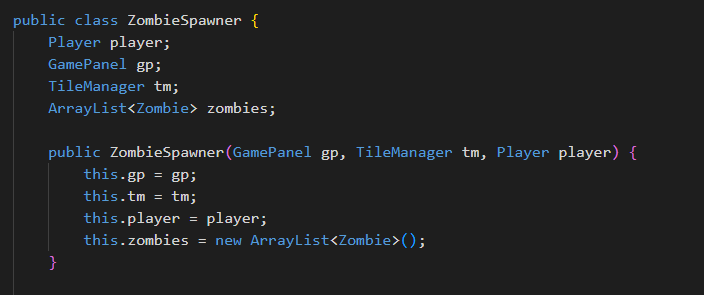
Polymorphism

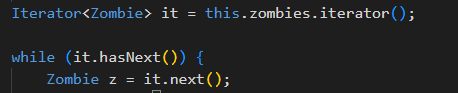




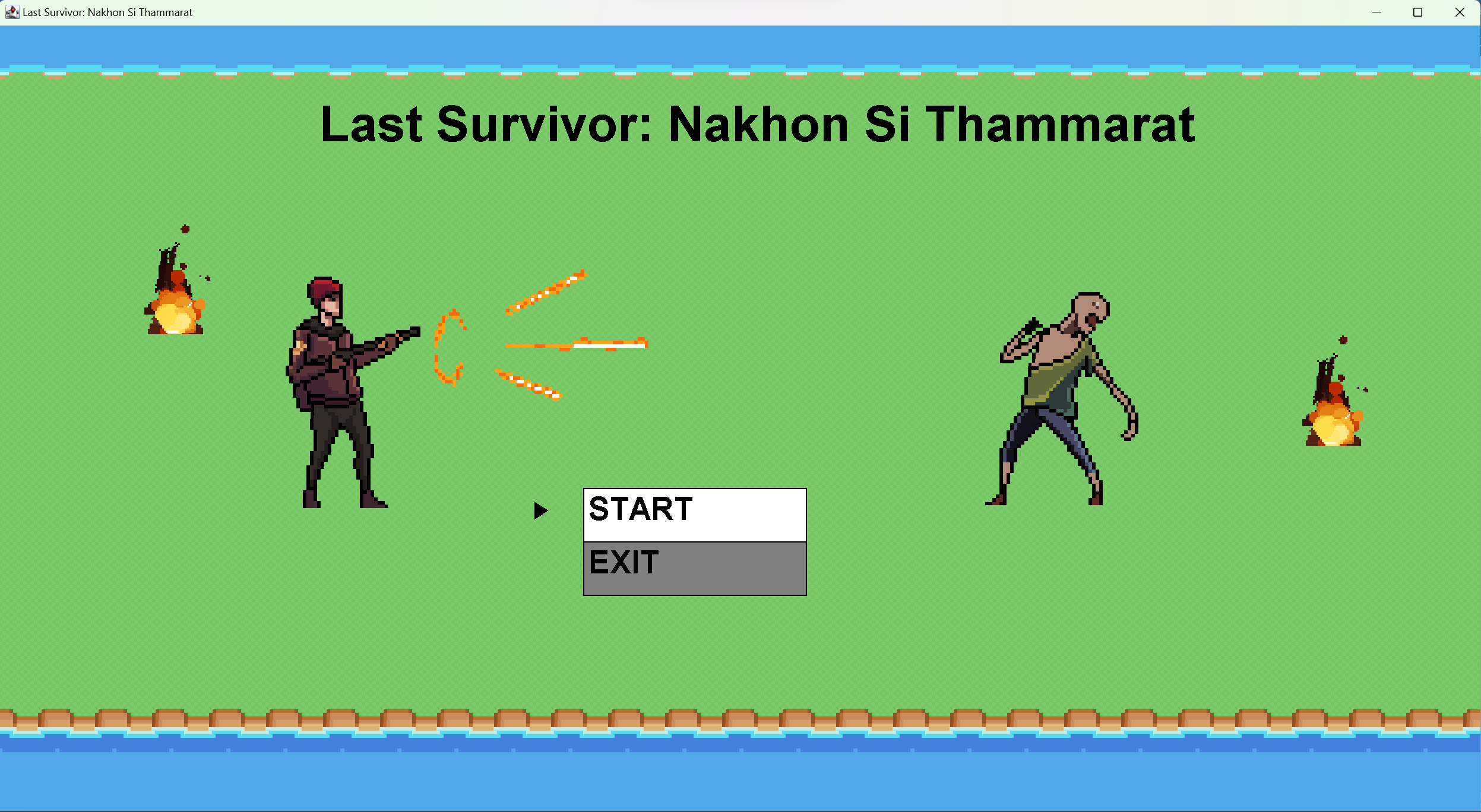
In ObjectManager this code stores arraylist of GameObject

In ZombieSpawner





For each item in ArrayList also uses polymorphism!

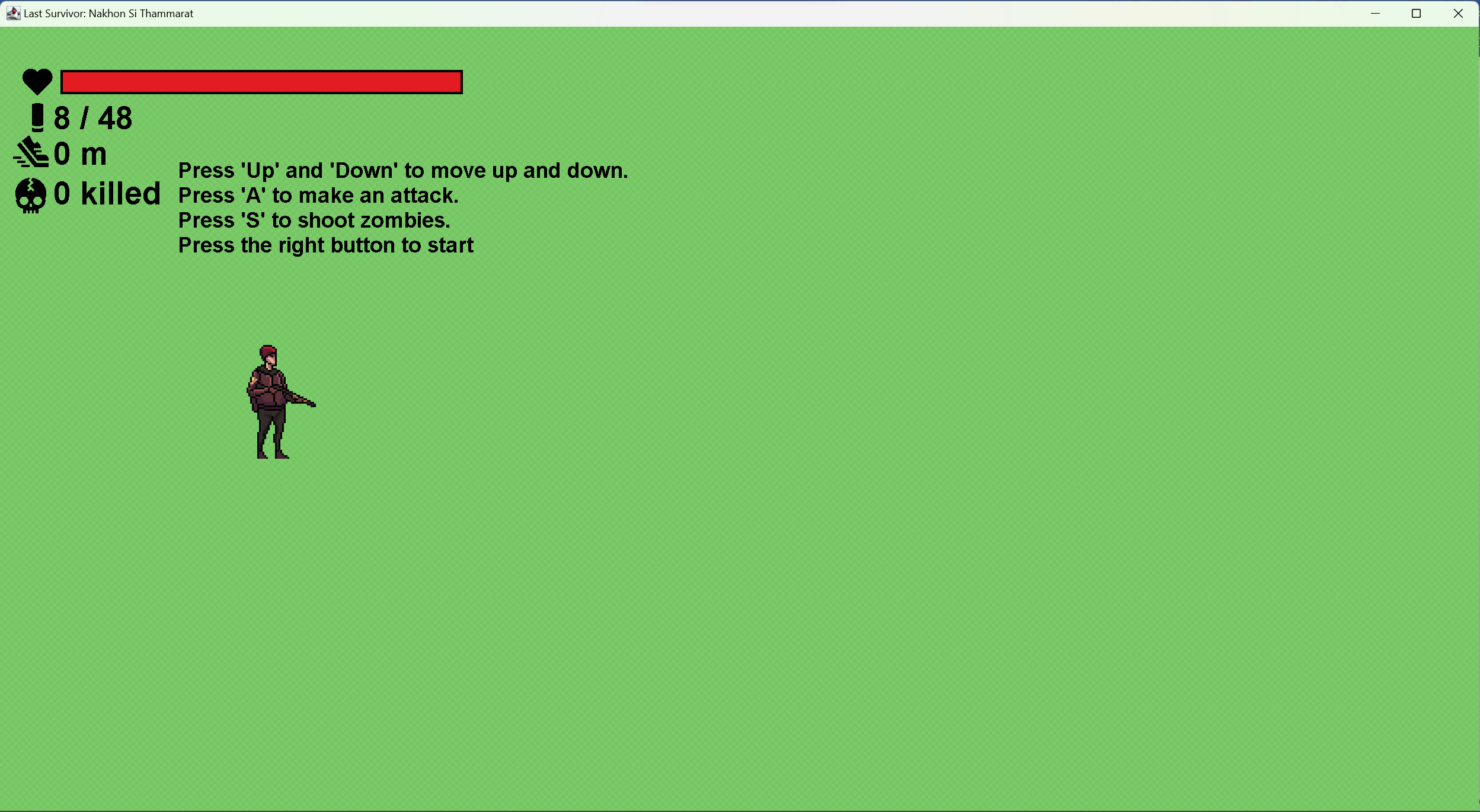


Components:

* Game title
* Start button
* Exit button
* Back ground

Event handling

* Press up&down
* Press spacebar to select



Components:

* Player health
* Ammo
* Run distance
* Kill count
* How to play message
* Icon

Event handling

* Press up&down to move
* Press A to attack
* Press S to shoot



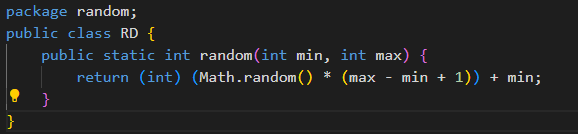
* Restart button
* Exit to home button

Event handling

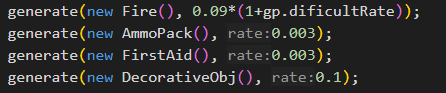
* Press up&down
* Press spacebar to select

My algorithms

Random integer between min to max. It’s used to random position of generate game object

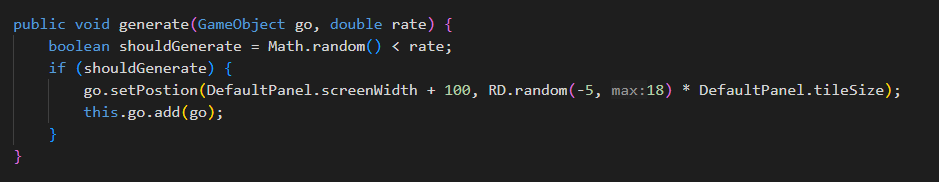


Generate game object to screen with probability (this.go is an array list of GameObject)



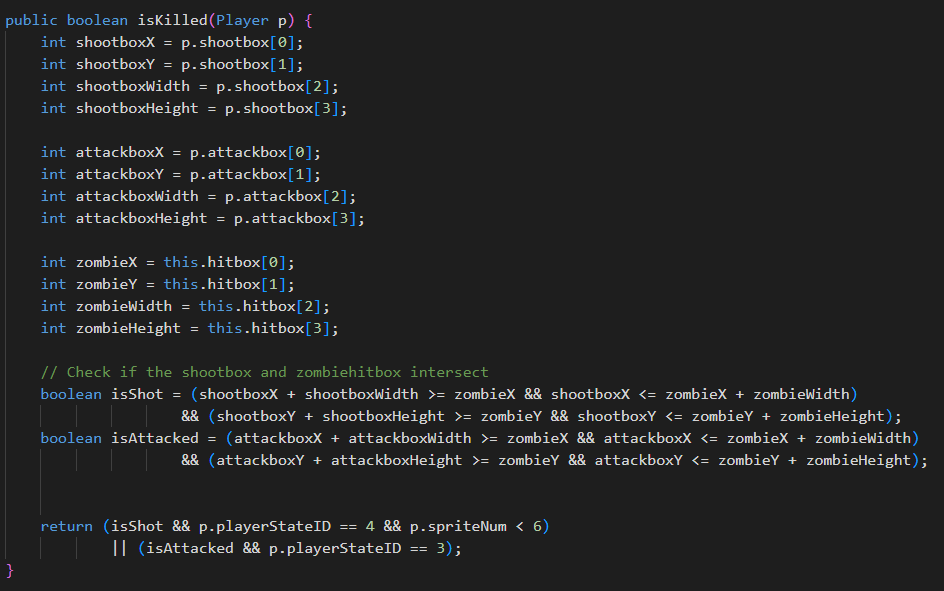
0.3% per 1/30 seconds to generate Ammopack

10% per 1/30 seconds to generate Decoration



Check hitbox of player and attackbox of zombies (Check if two rectangle intersect)

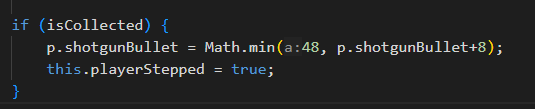




Health bar will change slowly and smoothly



When player step on ammo pack then increase player shotgun ammo by 8. And 48 is limit using Math.min()



**Chapter 3: Summary**

Problems encountered during development:

1. Game lag, difficult to fix.
2. The game didn't go as planned due to time constraints.

Highlights of the program:

1. Fun game, difficult to play

Suggestions:

The game should be simpler, with fewer requirements since it's just a mini-project.