CSC 413 Project Documentation

Summer 2021

Anmol Burmy

921143454

CSC 0413-01

GitHub Repository Link -

github.com/csc413-su21/csc413-tankgame-Burmy

Table of Contents

1 Introduction

1.1 Project Overview

The goal of this project was to build a 2D Tank Game written in Java. The main objective of building this game was to make it have a smooth performance along with pleasant user experience by using the correct Object Oriented Principle (OOP) along with code reusability.

1.2 Introduction of the Tank game

The Tank Game is a two player cooperative game where the objective of each player is to destroy the other player's tank. Each player has 100 health points with 3 lives. After each death, the tank will be respawned on the map with one life taken away. The map has three types of boxes, breakable (wood boxes), unbreakable (metal boxes) and pickup (powerup boxes). The game has 3 power ups which the players can get from picking up certain boxes around the map. There is an Armor powerup which gives the player 50 armor and adds 50 points to the player's health. There is a Life powerup which increases one life of the player. And there is a rocket powerup which doubles the player's bullet's damage but also decreases the fire rate of the bullet. The game ends when one of the tanks has lost all of its lives.

2 Development Environment

2.1 Version of Java and IDE Used

This project was developed using IntelliJ IDEA with Java version 16.0.1.

2.2 Any special libraries used or special resources

1. Menu Music -> Doom OST - E1M2 - The Imps Song (Link)
2. In Game Music -> Doom OST - E1M1 - At Doom's Gate (Link)
3. In Game Assets (Tanks, Bullets, Walls, Powerup boxes, Map tiles) -> Top-down Tanks Redux Assets by Kenney (Link)
4. In Game Sound Effects -> Sound Effects by Kenney (Link)
5. SoundPlayer class taken from the "Airstrike" game provided by the professor.

3 How to Build/Import your Project

To build/import the project, go to my repo and click the green "Clone or Download" button on your repo's home page. Then select either HTTPS or SSH, If you are not

sure what SSH keys are then use the HTTPS method. Copy the link and then open your terminal. This will either be Git Bash on Windows, Terminal on Mac or Linux, or the terminal for your Windows Linux Subsystem if you have it installed. In your terminal type: git clone repo_url_you_copied. Once the repo is cloned, follow the steps below to import your project into IntelliJ IDEA.

1. Select Import Project
2. Select the “csc413-tankgame-Burmy” folder as the source root of your project.
3. Keep the “Create project from existing resources” radio button selected.
4. Now keep clicking “NEXT” until the import is finished.

After importing the project, to build the JAR, first we need to make sure our src and resource folders are marked correctly. So in order to do that -

1. Click on “FIle” on top right and select “Project Structure”.
2. From the Project Structure window, select “Modules” and make sure that the “src” and “resources” are marked as Sources and Resources respectively.
3. After that to build JAR from the same window, select “Artifacts”, then click on +.
4. Select “JAR” then “From modules with dependencies” and then select your main class and click ok.
5. After clicking “Apply”, click on “Build” on top of the window and select “Build Artifacts” and then “Build”.

4 How to Run your Project

After you have built your JAR, you can press the play button on top of the editor window. This will bring up a window of the tank game and you can start playing.

The Tank Game is a two player cooperative game where the objective of each player is to destroy the other player’s tank. Each player has 100 health points with 3 lives. After each death, the tank will be respawned on the map with one life taken away. The game also has 3 power ups which the players can get from picking up certain boxes around the map. The game ends when one of the tanks has lost all of its lives.

Controls to play the game -
Player 1 Controls (Red Tank)
['W' Key] - Tank 1 moves upwards

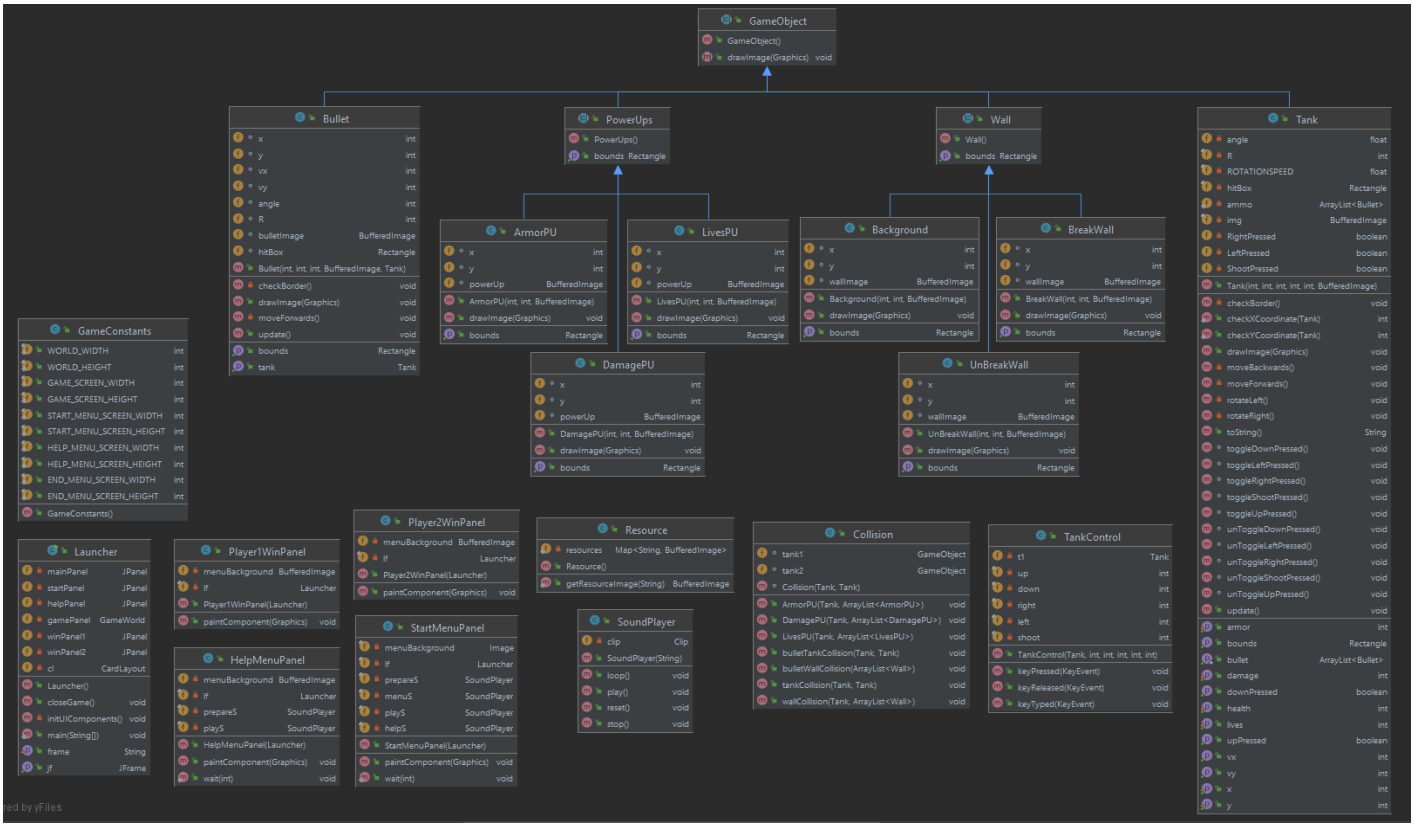
['S' Key] - Tank 1 moves downwards
['A' Key] - Tank 1 rotates left
['D' Key] - Tank 1 rotates right
['Space' Key] - Tank 1 shoots

Player 2 Controls (Blue Tank)
['Up' Key] - Tank 2 moves upwards
['Down' Key] - Tank 2 moves downwards
['Left' Key] - Tank 2 rotates left
['Right' Key] - Tank 2 rotates right
['Enter' Key] - Tank 2 shoots

5 Assumption Made

I assumed that I had to keep in mind the basic requirements of the project and finish them first before working on extra additions to the game.

6 Tank Class Diagram



7 Class Descriptions of classes implemented

7.1 Launcher Class

This class is responsible for launching the game and contains the main method. It basically sets up the environment of the game. It includes all of the game screens including start screen, help screen , game screen and end screen.

7.2 GameConstants Class

This class contains all the constants of the game.

7.3 StartMenuPanel Class

This class contains the start screen of the game and contains the buttons to start the game, go to the help menu and exit the game.

7.4 HelpPanel Class

This class contains the help screen of the game. It shows the user the controls of the game and contains the buttons to start the game, and exit the game.

7.5 Player1WinPanel Class

This class contains the end screen of the game when player 1 wins. It contains the buttons to play the game again, and exit the game.

7.6 Player2WinPanel Class

This class contains the end screen of the game when player 2 wins. It contains the buttons to play the game again, and exit the game.

7.7 Tank Class

This class has all the information that is related to the tanks of the game. We have data fields for tanks position, movements, speed and rotation. This class also loads the tanks health bar, lives and armor bar whenever an armor powerup is picked up. This class also manages the fire rate of the bullets after and before picking up the rocket powerup.

7.8 TankControl Class

This class is associated with a tank object. It implements the keylisteners interface and has all the key presses.

7.9 Bullet Class

This class has all the information that is related to the bullets fired from the tanks. We have data fields for the bullet's position, movements, speed and rotation.

7.10 Wall Class

This is an abstract class for all the walls which are breakable walls, breakable walls and the background tiles in our game.

7.11 BreakWall Class

This class inherits from the Wall class. This class represents the wall in the game that can be broken when shooting a bullet into it.

7.12 UnBreakWall Class

This class inherits from the Wall class. This class represents the wall in the game that cannot be broken when shooting a bullet into it.

7.13 Background Class

This class inherits from the Wall class. This class represents the grass tile which is the background in the game. The grass tile is not solid which means that it would not collide with anything on the map including tanks and bullets.

7.14 PowerUps Class

This is an abstract class for all the powerups in our game.

7.15 ArmorPU Class

This class inherits from the PowerUps class. This class represents the armor power up in the game which gives the tank 50 health points.

7.16 DamagePU Class

This class inherits from the PowerUps class. This class represents the damage power up in the game which doubles the damage.

7.17 LivesPU Class

This class inherits from the PowerUps class. This class represents the armor power up in the game which gives the tank extra life.

7.18 Collision Class

This class handles all the collisions of the games which includes collisions of bullet and tank, tank and tank, bullet and walls, tanks and power ups.

7.19 Resource Class

This class manages all the game assets of the game.

7.20 SoundPlayer Class

This class manages all the sounds used in the game.

7.21 GameWorld Class

The GameWorld class is responsible for running the game. This class implements Runnable interface so when the game is launched it is executed on its own thread. This class also draws the entire world being responsible for the split screen and the mini map.

8 Project Reflection

I think this project was a very good experience for me as I have never made a game before. I am quite happy with the game I have made as it passes all the requirements for this project, but I still wanted to make more extra additions to the game but I was not able to, mainly because of not having much time. I had added many sound effects to the game, when a bullet shoots, when a tank picks up any power up etc. But for some reason, every now and then whenever I would play the game, the sound effects would break the game. So I had to comment out all the sound effects as I want the user experience to be smooth. If you want to experience the game with all the sounds, you can uncomment the sounds in the tank and collision class. Also I tried alot to add explosions gif whenever the tank dies, but I was not able to do that either. Also I feel like I could have done a better job of using the GameObject class because I feel some of my code is repeatable and could be more clean. Overall, I was still able to add some background music and

voice overs which worked smoothly and I was pretty happy in the end how the game looks and works.

9 Project Conclusion/Results

I think this project was fairly difficult but fun and needed a lot of time to be built and precise with the instructions that were provided. This was a great learning experience for me as it showed me what Java language is really capable of. I tried to make the code as readable and understandable as possible by adding comments and making the code clean, so people or even I can understand what the program does in the future.