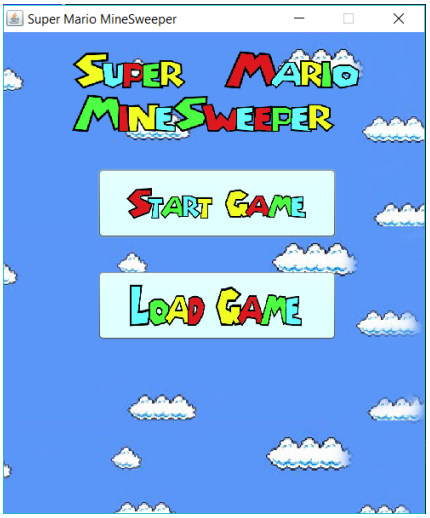
Individual Final Project – Super Mario MineSweeper

Introduction

The individual final project was based on creating a MineSweeper game from scratch using Java and the NetBeans IDE. For my project I decided to make the MineSweeper with the thematic of Super Mario. Gameplay wise it is still the same as the normal MineSweeper but the thematic gives it a fresh look. Besides, you also can also save the game in 3 different save files.

List of features

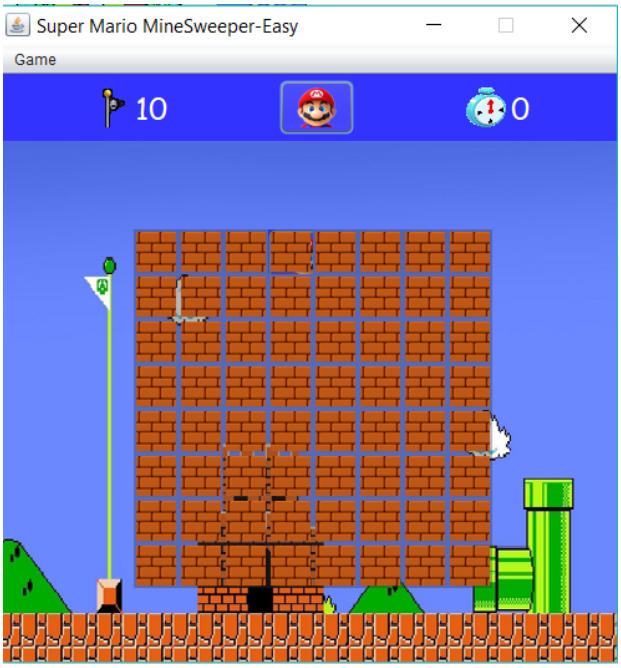
- **Main Menu**



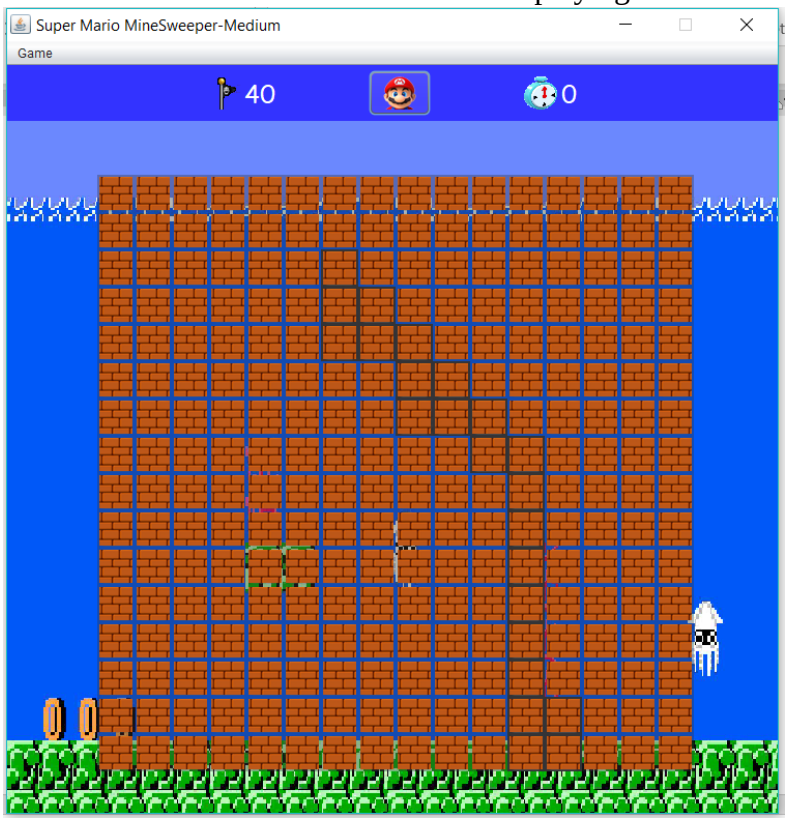
The game counts with a main menu where the player can select if they want to start a new game or to load a saved game.

- **Different difficulties**

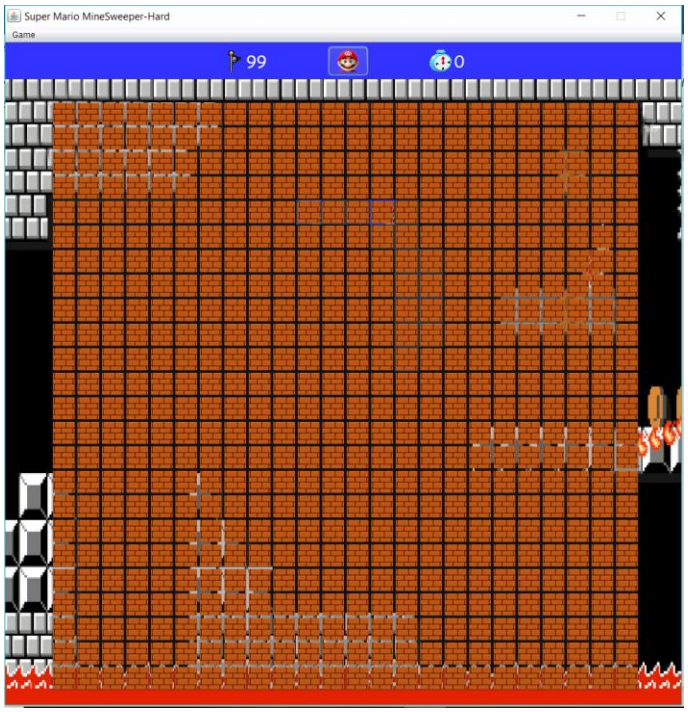
If the player selects to start a new game they will have to choose between one of the three difficulties.



The easy difficulty has a grid of 8x8 blocks with 10 mines. The theme of this level is the overworld first level of Super Mario Bros. which can be seen in the background and the iconic theme can be heard while playing.

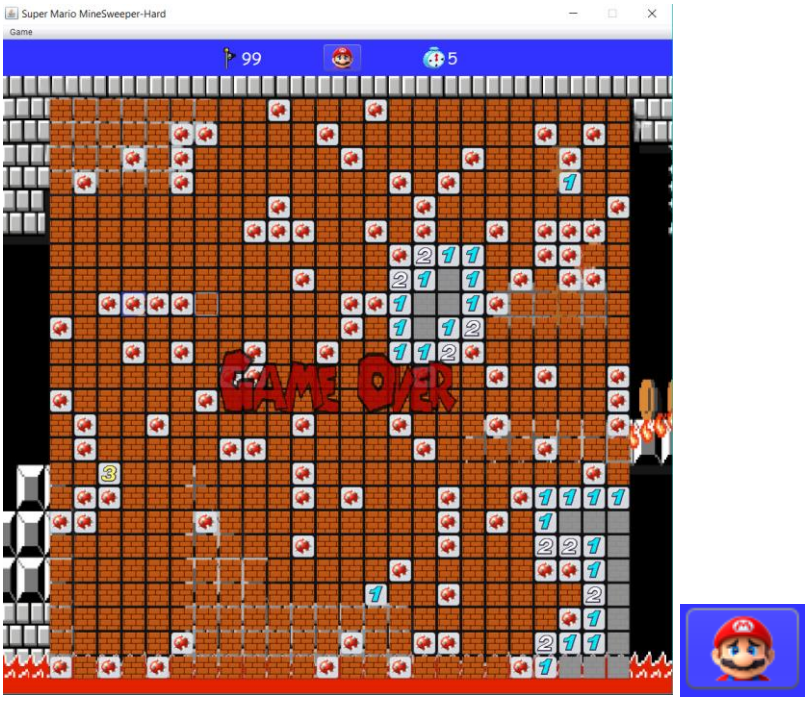


The medium difficulty has a grid of 16x16 blocks with 40 mines. The theme is the underwater levels of Super Mario Bros. which can be seen in the background and the underwater music theme can be heard while playing.

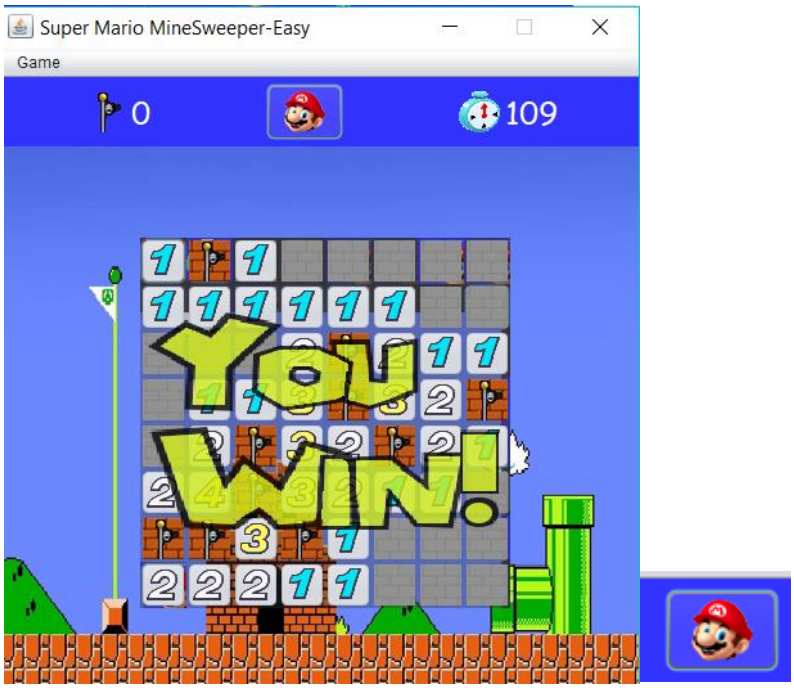


The hard difficulty has a grid of 24x24 blocks with 99 mines. The theme is the castle level from Super Mario Bros. which can be seen in the background and the castle theme can be heard while playing.

- **Game over/Win screens**

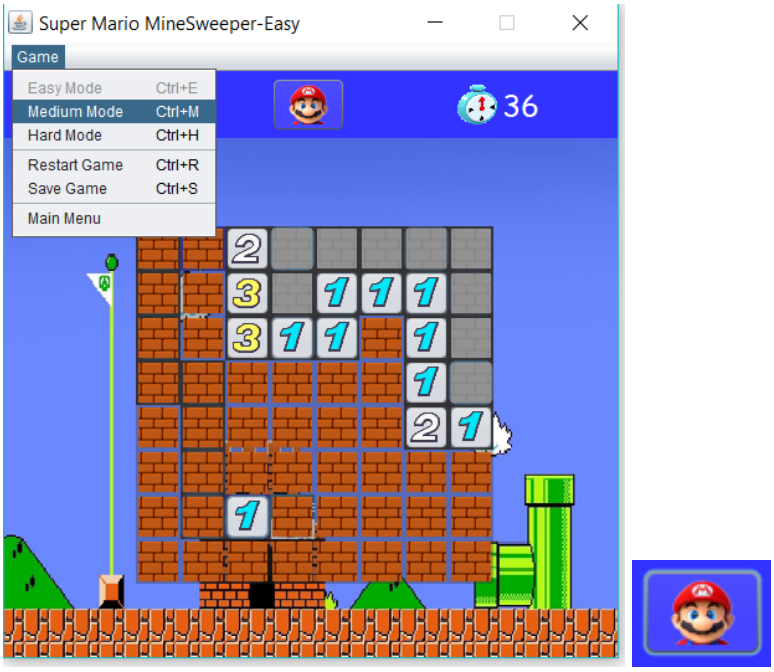


When you lose a game, the classic loosing sound can be heard, and the game displays a game over text. The face Mario in the restart button also changes to a sad Mario and the mouse listeners from every block button are removed.



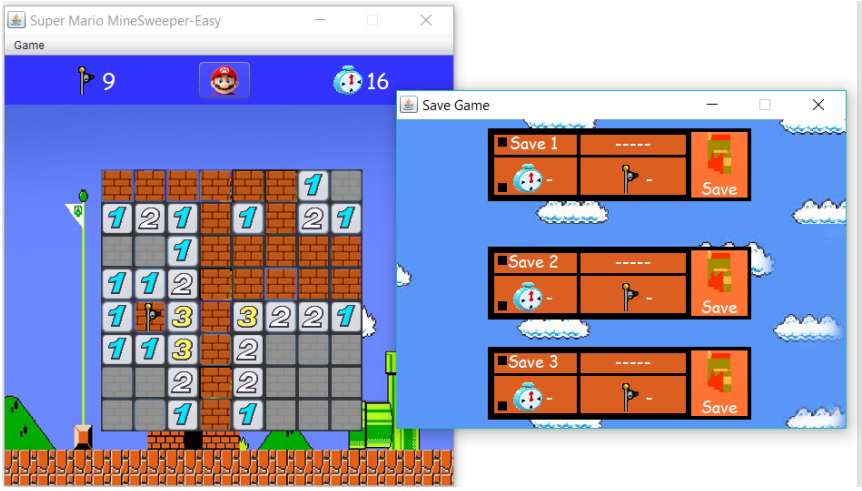
When you win the game, the Super Mario Bros. “Clear World” sound can be heard, the game displays a “you win” text and the Mario restart button changes to a happy Mario.

- **In-game menu and restart button**

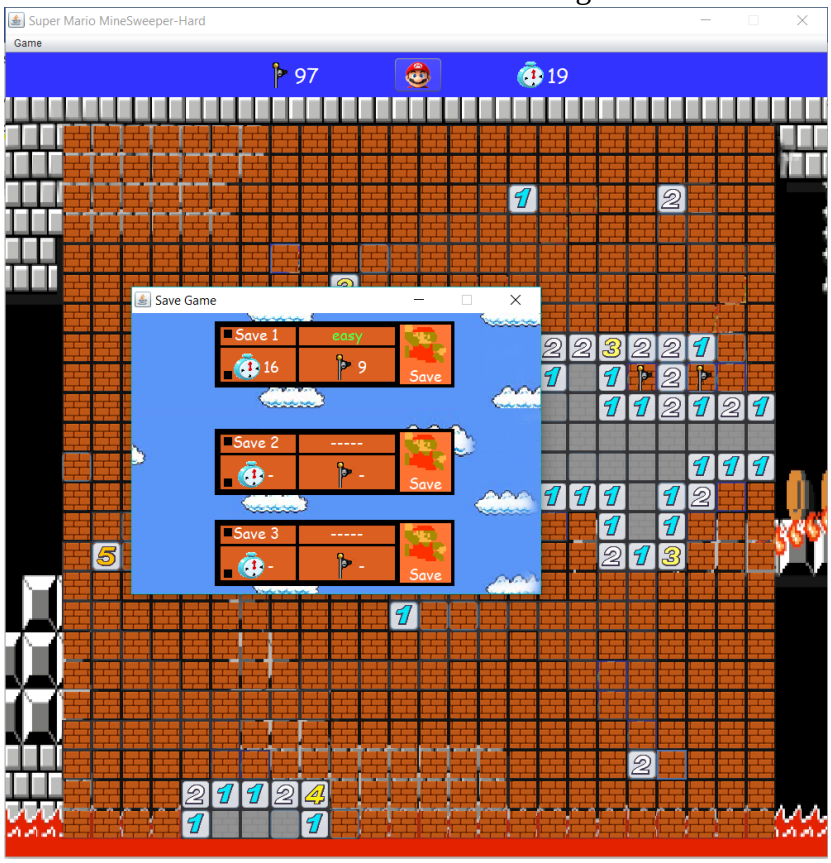


When the player is in the game, they have the ability to open an in-game menu that allows them to go to a game with a different difficulty, restart the game, save the game or return to the game menu. Most of the options also show a combination of key presses that will achieve the same result. The game also features a restart button with the face of Mario which changes depending on the state of the game.

- **Saves**



The game allows the player to save their progress. When they select the “save game” option, the Save Game frame will be shown and the timer of the game will be paused until they continue playing. In the Save Game frame players will be able to choose one of three different save files to save their game.



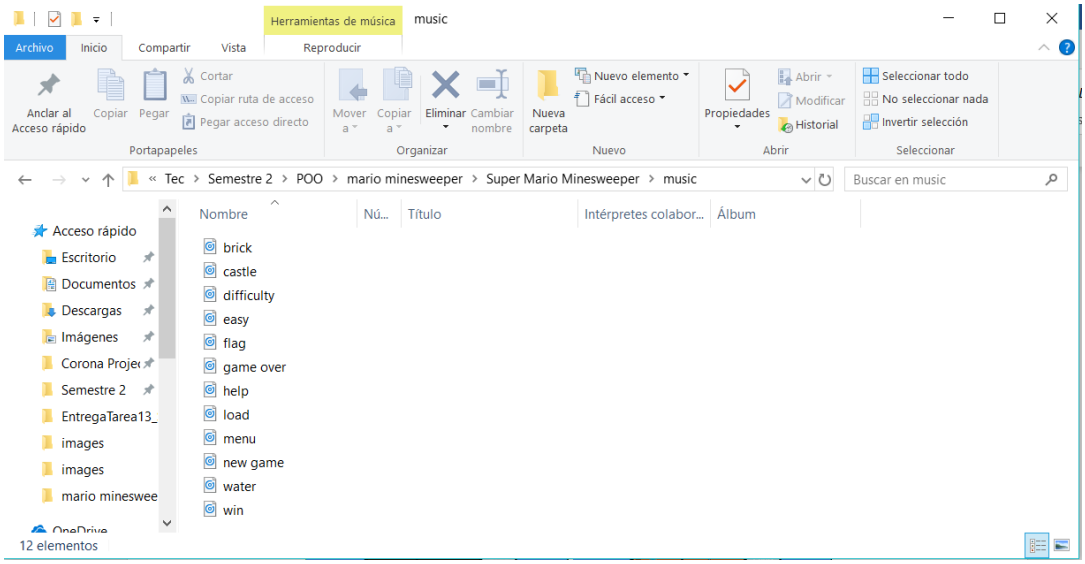
If there is already a saved game in one of the files, the Save Game frame will also display relevant information from those saves.

- **Loading saves**

The game also allows the player to load saved game files. If the player enters the Load Game screen when there are not any save files, the load button will be disabled.

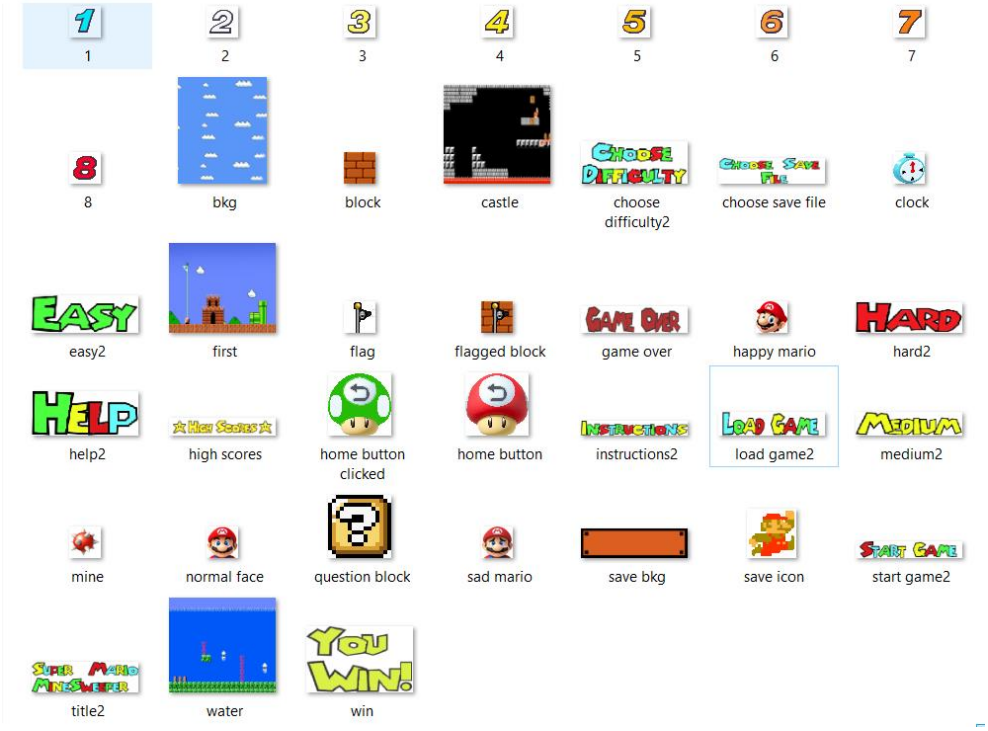
If there are saved files and the player enters the Load Game screen, they will be able to choose the load option which will open the selected game. They will also be able to see relevant information from each save in order to know which save they are going to open. The information that is shown includes the difficulty, time passed in-game and flags left.

- **Music and sounds**



The game features several music themes and sounds from Mario games throughout history. All music and sounds can be heard in the game except for the “help” music since that feature is still being worked on.

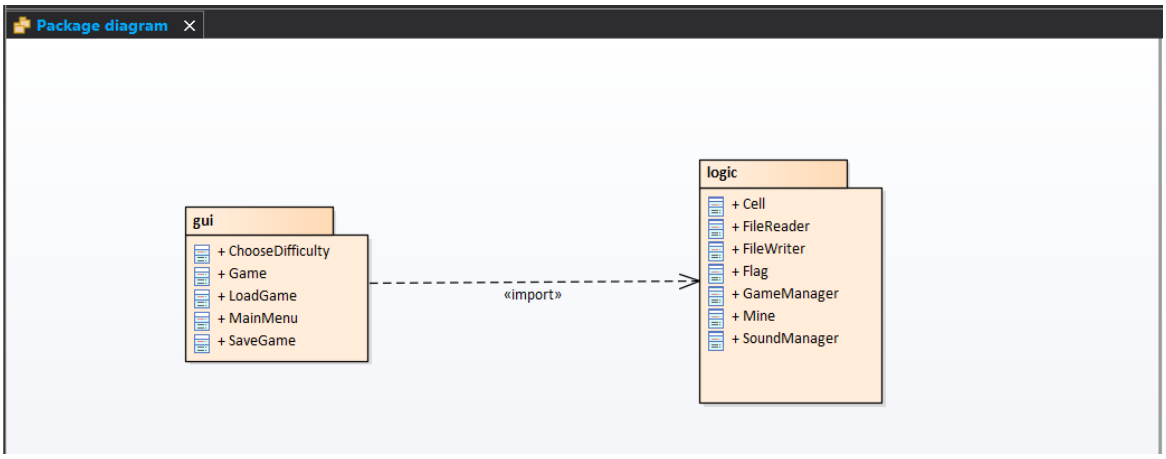
- **Custom sprites**



All the sprites from the game where originally downloaded from the internet but they were edited to fit in the game. Every sprite except for “high scores” and “help2” can be seen in the game. The mentioned sprites are being used for features that are still in progress.

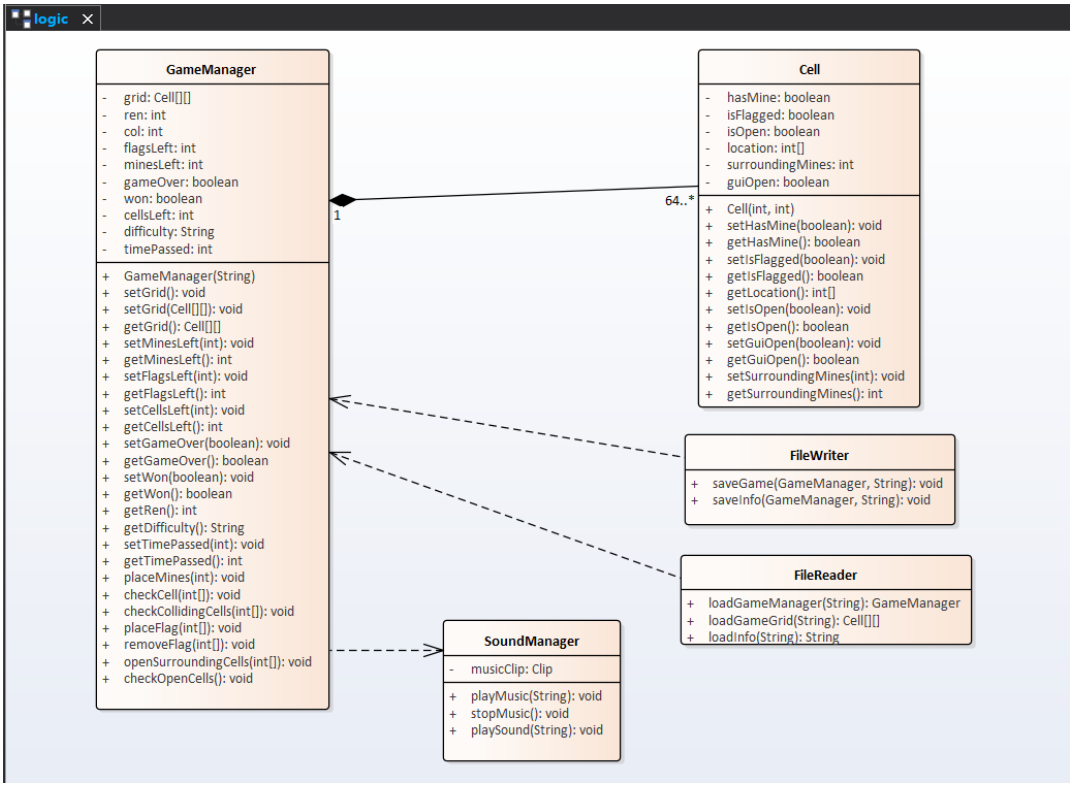
Analysis and Design

Package diagram



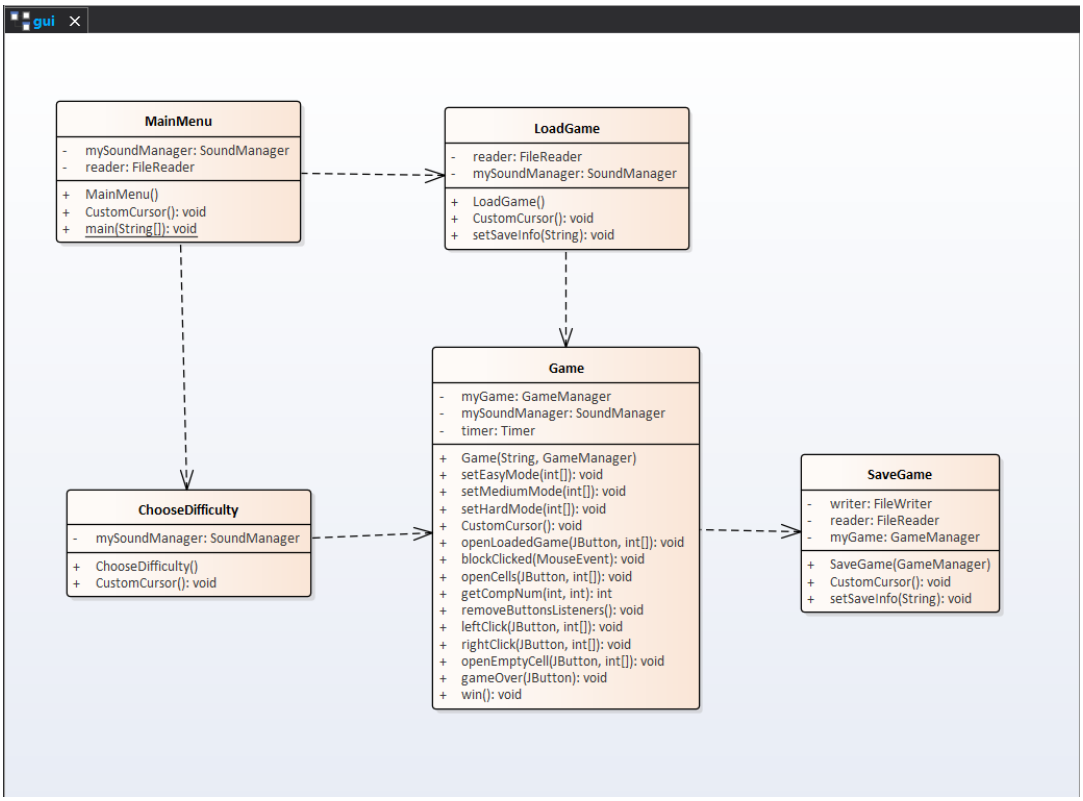
The game was made using two packages. One for the gui and one for the logic.

Logic package diagram



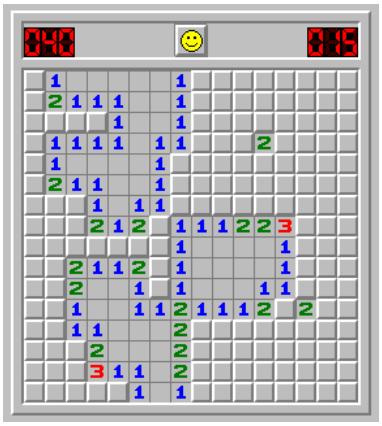
In the logic package, 5 classes were made, and the relationships were mainly dependency and composition between GameManager and Cell.

gui package diagram



In the gui package 5 classes were made, the relationships are all dependency since each frame just opens the one that it is related to.

Initial mockups



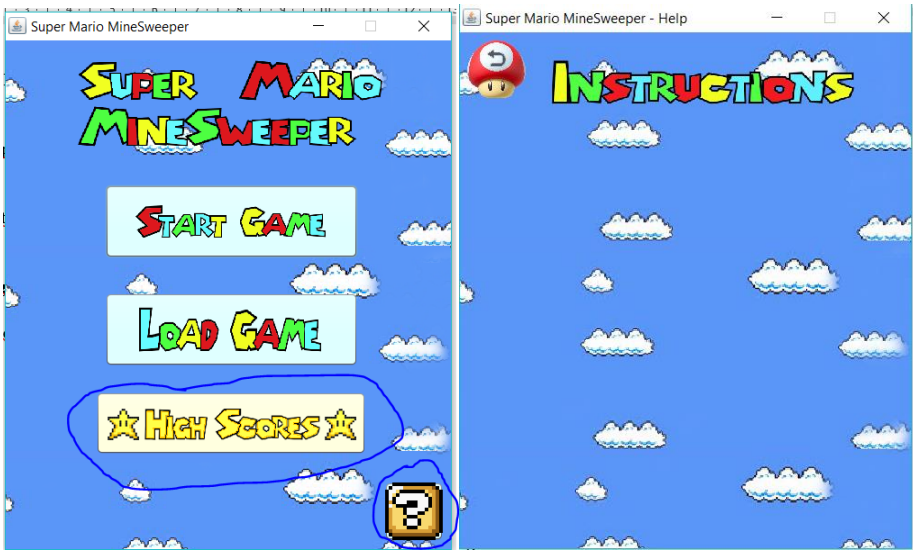
The initial mockup for the minesweeper was just a screen with the title and buttons of different difficulties. For the game I just had the idea that it would have a similar look as the original minesweeper (which had the game grid, a flag counter, a smiley face that acted as a restart button and a timer) with a Mario “reskin”.

Total Time Spent

Total (min)	1773
Total (hrs)	29:33

Future Updates

Due to time restrictions, the following features were not finished but they are being worked on for future updates.



- **Saving and displaying highscores**
  The "high scores" option will allow the player to view the high scores of the game. This will be displayed in a new frame where the high scores from each difficulty will be displayed in separate tabs. Each tab will list the name of every player that has won the game and the time they took to win in order from the player that took less time on the top to the player that took more time in the bottom. In order to know what to display in the high scores, the game will ask every winning player to enter their names which will be saved along with the time they took in a text file.
- **Instructions screen**
  The question block button will take the player to the Instructions screen which is going to display instructions on how to play the game and provide explanation for any necessary feature that requires it. The instructions will be displayed in different labels which will be placed inside a panel with a card layout to show one thing at a time. There will be arrow buttons which will allow the player to move to the next/previous "page" (label) of the instructions screen.