Cs 319 – Object Oriented Software Engineering

Final Report

Space Invaders

Group 2-C

Arda Gültekin

Koray Gürses

Yalchin Aliyev

Enes Erol

Supervisor: Uğur Doğrusöz

Contents

1. Introduction

By the time this report is written we are very close to finishing our implementation. After the first iteration demo, we've added sound files, effects, a main menu, and different levels to our game. We are currently implementing two bonuses with two different speciality. One of the bonuses is a health bonus, which will increase the user's ship's health. Second bonus is to increment the damage of user's ship. These bonuses can appear randomly on the screen and when the player ship hits a bonus, player ship will gain its effect. Our menu has four options on it: Play game, How to Play, Settings and Quit. Play Game option will start the game from its first level. It also starts the audio for that level and draws the enemies for that level. How to play option will take the user to a screen where they can learn how to play the Space Invaders game. User can mute the game's sound effects and the music. Quit button will simply close the game.

We are planning to finish our implementation before our demo in the instructor's office. We have divided implementation tasks among each other after the first iteration, this helped us to speed up the implementation process. We used Git Bash to update our code in GitHub. Whenever somebody made changes to the code, they've committed that part to GitHub so the rest can continue from the updated code. We made some changes to our design in order to have a smoother implementation, those changes will be mentioned in the following part of this report.

2. Design Changes:

Main changes in our design has been done In the Input Manager class. First of all we've added a Mouse Listener. Input Manager is responsible

for handling the mouse input. In the Menu class we've created our main menu's UI. We've created couple of rectangles rather than directly using JButton for the menu buttons. Our Input Manager class checks for the mouse press events in the specific rectangles. For example, in order to start the game, Input Manager will check if there is any mouse pressed in the range of "Play Game" rectangle. If there is a mouse pressed event seen in the boundaries of that specific rectangle, game will start. It works like that for the other three rectangles (basically we used rectangles as our buttons).

While implementing our Input Manager at first, we encountered a bug: even when we are in the game screen when we clicked the locations of four buttons in the main menu, their mouse pressed actions still called. In order to fix this bug, we have implemented states in our Game Manager class. There are currently three states: "MENU", "GAME", "HOW TO PLAY". These basically indicates states of the game. If the user is on the main menu screen it means current state is "MENU". If the user presses the Play Game button, than game screen will show up and the state will be "GAME". Similarly, if the user clicks the How to Play button on the menu game state will be "HOW TO PLAY" and a guide screen will show up. Bonuses will appear randomly on the screen and user can use player ship to shoot these bonuses in order to get their effects. One of these bonuses will increase player's health while other one increase the player ship damage for a while.

3. Lessons Learnt:

- Using sound files with a JAVA project.
- Implementing .png files to a JAVA project.

- Refreshed and improved our knowledge of implementing key & mouse events in JAVA.
- Improved our GUI knowledge.
- Using Git efficiently.
- Designing a project in an iterative matter.
- It was the first time we had experience with writing code for design patterns such as Singleton and Façade.
- It was the first we used many different diagrams to create an understandable yet sometimes technically complicated designs.
- We've learnt to work together as a group, every group member understood the piece of code that others implemented and can move on from the point where other members left with no trouble.

4. User's Guide:

4.1. System requirements & installation:

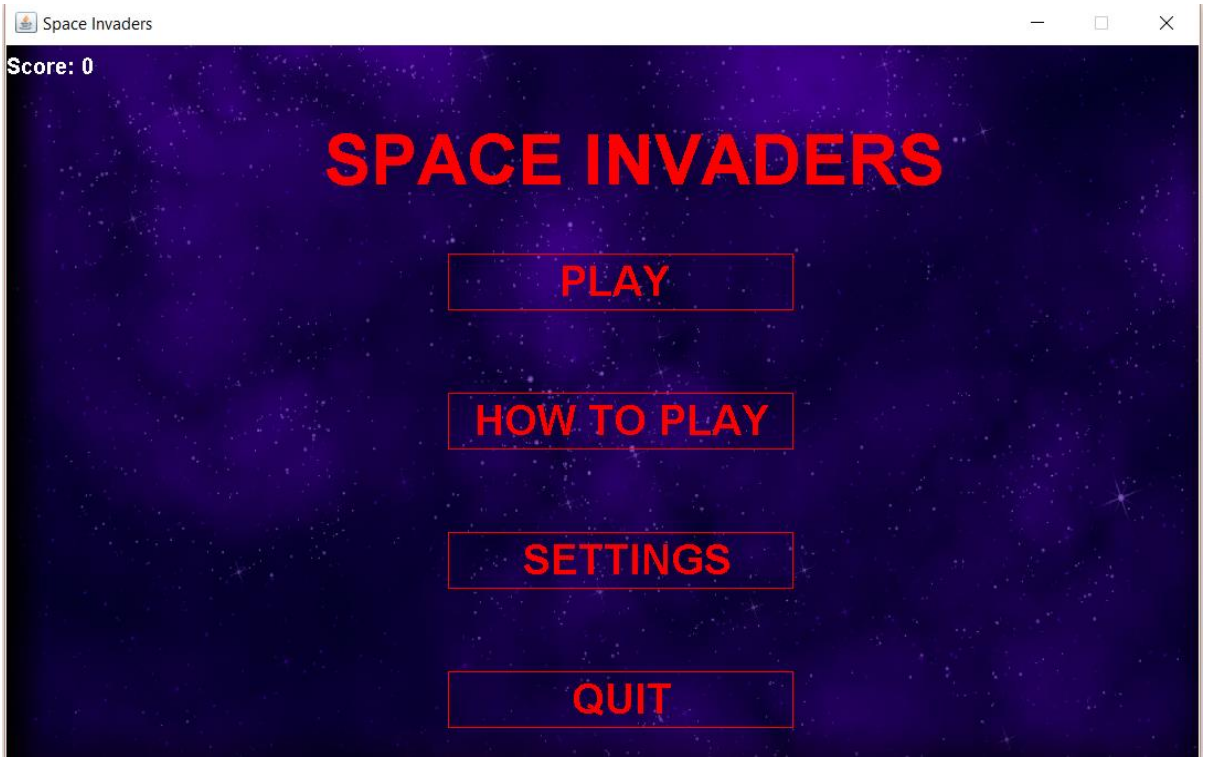
Space Invaders will be a jar file, a simple desktop application. It requires JAVA run time environment version 1.8. You can find our game's final version from our GitHub repo: https://github.com/yalchinAlv/2C.Space-Invaders.

4.2. How to Use:

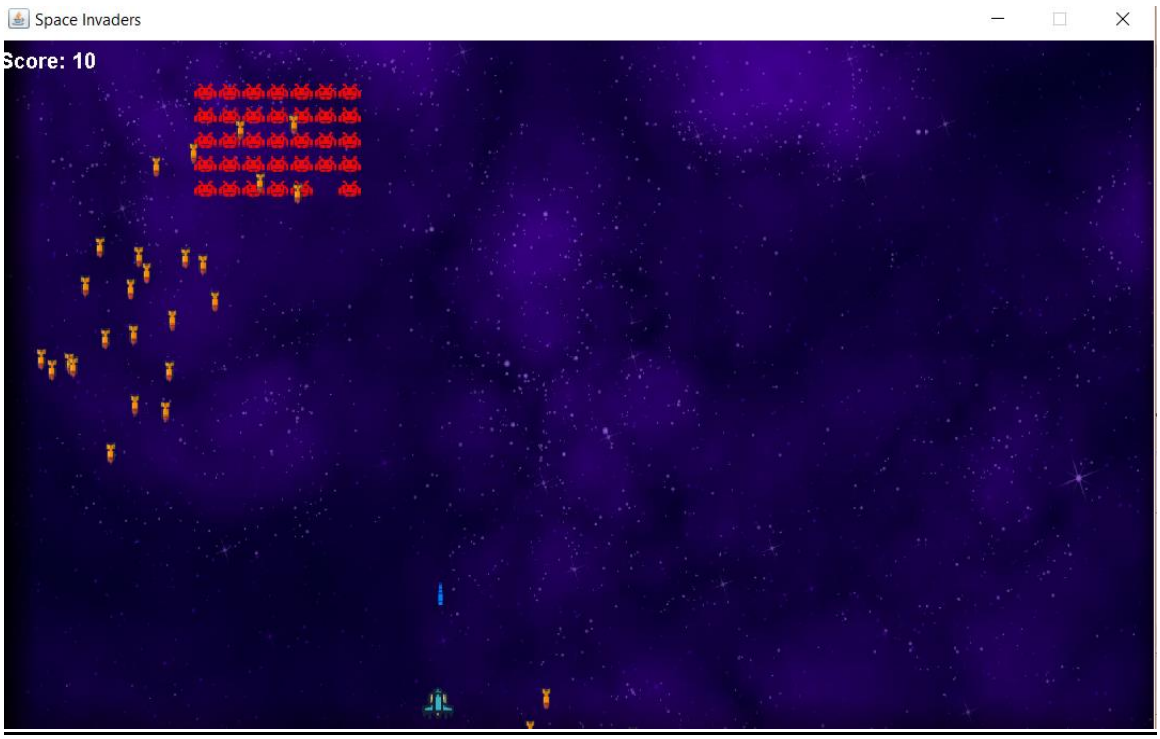
Players can learn how to play Space Invaders game by clicking "How To Play" Option from main menu screen. Game will start when the player presses "Play Game" button. Game will initially start from the first level, the goal is to kill all the alien ships before they reach

to the player's ship or killing player's ship. Player should take advantage of bonuses in order to proceed to next level in an easier way. "Setting" button will bring out a menu where player can mute the game's sound effects and music. If you want to close the game, please click "Quick" button from the main menu. User can also get scores by killing aliens. Your score will increase for each enemy you kill in the game.

Main Menu:



Game Screen:



How To Play Screen: