03-06-2019

Space invaders

By: Patrick Sirich & Matthias Phillip Skou

Table of contents

[Introduction to the game 2](#_Toc10379831)

[Structure of the game 2](#_Toc10379832)

[Structure of the code 3](#_Toc10379833)

[Challenges during development 3](#_Toc10379834)

[Improvements and future implementations 4](#_Toc10379835)

[Things we could have done better 5](#_Toc10379836)

# Introduction to the game

We have decided to create our version of the popular arcade game Space Invaders.

Space Invaders is a Japanese arcade game made popular in the U.S in the 1980’s.

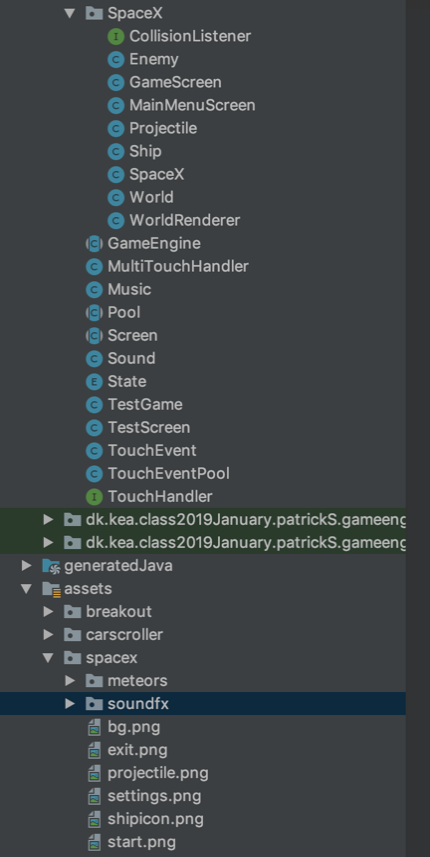
Space Invaders is a fixed 2D shooter where the player controls a ship by moving it horizontally across the bottom of the screen and shoots a laser at the advancing enemy aliens. The enemy aliens is fixed horizontally and vertically and moves together at all times making player placement fairly predictable. The aim of the game is to eliminate all enemy aliens without dying, the enemy aliens will also shoot at the player ship as it advances and as soon as the enemies hit the bottom of the screen, you lose a life. When you shoot an enemy you gain points. When you shoot the enemies the enemy movement and the music speeds up to stress the player and increase the difficulty.

For our game, we have simplified the original game due to time restrictions. For our game we currently have just one enemy type and we have no barriers as the enemy does not shoot at you. We also made our shooting function so that it fires immediately after hitting and enemy or once the bullet gets to the top of the screen.

# Structure of the game

1. Once the game is up and running, the start menu is shown, from here the player has the option following options:
   1. ‘Start’
   2. ‘Guide’
      1. If the player chooses this, they will see a description of how to play the game and what rules there is. There will also be a return button to the main menu.
   3. ‘Credits’
      1. This will show a list of credits and a return button to the main menu.
2. If the player chooses to ‘Start’ the game, a game-screen will appear with the enemies at the top, a score counter and amount of lives in the upper left corner.
3. In the upper right corner there is a pause button, if the player chooses to pause they will have the options to resume the game or end the game and return to the start menu.
4. The ships laser cannon will start shooting right away and the enemies start advancing. When enemies are hit, they disappear and the score counter is incremented.
5. Once all the enemies have been cleared a new horde of enemies will appear at the top of the screen. The enemy movement speed and projectile speed is increased.

# Structure of the code

We created a package ‘SpaceX’ where all of the classes and interfaces we use throughout the game is placed. The classes outside of this package are part of the game engine that we created during class. We created our classes by standards of the OOP, meaning that our Enemy, Ship etc. are in separate files containing fields and are created using our GameEngine drawbitmap method.

In our assets folder we have all the pictures that are displayed in the game. This is also where the sound effects and music is found.

# 

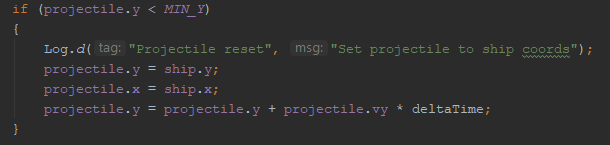
# Challenges during development

Enemy collision hitbox:

During the course of the development we encountered an issue where the enemy hitbox was too small and the projectile would sometimes go through it rather than register it as a hit. We have at the point of hand-in not solved this issue but expect to have it solved before the exam.

Shooting function:

We had some challenges with our shooting function and specifically that it would only shoot once and then disappear from the screen. We solved this by resetting the projectile rather than creating a new projectile every time we wanted to shoot. We however also encountered an issue where the projectile would follow the ship on the x-axis after being shot. We solved this by changing the projectile x-axis to the ship x-axis in the update method at the point of shooting meaning that the x-axis for the projectile would remain at a fixed x value at the correct ship position.



This solution fixed 2 things for us: the projectile while moving up will not change direction on the x-axis and it shoots from the same x and y-axis that the ship is at.

Resetting enemy position:

During development we encountered an issue where after the enemies hit the bottom of the screen and the player loses a life, the enemies would not reset to it’s correct position but however remain at the bottom of the screen causing an instant loss of all lives. At the point of hand-in we have not solved this issue but expect to have solved it before the exam.

# Improvements and future implementations

Barrier & enemy projectile:

Two core functionalities that we would implement in future development is the barriers and enemy projectiles. The enemies would shoot projectiles in a straight line towards the bottom of the screen, adding extra difficulty to the game, since you have to avoid these. This would also require us to add fixed barriers in front of the ship, as in the original game, so the player can hide behind these. These barriers would decay slowly as they are hit by enemy projectile, eventually leaving the player exposed.

Shop:

In future development for this game, we would like to add a shop where you can spend your points on buying upgrades. These upgrades could include a better laser cannon for your ship, armor upgrades for your ship(so you can withstand more enemy projectiles without dying), stronger barricades and special powers that can be used to fight the space invaders.

Levels & bosses:

We would also like to implement an improved level-progression throughout the game. We would do this by adding stronger enemies as you progress that require more shots to kill. Besides this, we would also add bosses that you have to defeat in order to progress in the game.

Enemy types & bonus points:

As the game progresses and the levels increase, we want to implement different types of enemies with both different hitboxes and attack-types. These new enemies would also give a different amount of points once killed and some of them may even drop objects that will award you random upgrades once destroyed. The new enemies would require more hits and move faster, so you need to upgrade your cannon in order to keep up.

Ship projectiles:

A more technical functionality to implement, would be to change the ships projectile so it shoots based on click or based on time. This would add a higher skill-cap allowing the player to better time their shots against enemies.

# Things we could have done better

At the beginning of our development process, we had multiple ideas and explored our possibilities. This resulted in us starting the development process long before we had a structured plan of what to create. This meant that during development we decided to change the game and had to start all over. We could have used tools such as paper prototype, to lay out a plan of what the game had to consist of and the limitations of the game. Since we didn’t have this plan, we spent a lot of time discussing what features should be implemented during our development.

Due to the lack of planning, we were poor at managing our time and spent a lot of time and effort trying to implement functions that should have been ruled out during a planning phase. We also spent a lot of time focusing on advanced functionalities, when we should have focused on core functionality instead. We started out trying to add 10 different types of enemies which we realized would take too much time to implement.