Project Final Report

Ranjan Biswas
CS 4237 :Space Invaders
George Washington University
December 10th 2011

Table of Contents

Introduction

Space Adventures is a game based on the original Space Invaders arcade game created by Tomohiro Nishikado in 1978. The objective of the game is to shoot enemy drones as they descend downwards towards the hero. The hero spaceship is equipped with a canon, that fires towards enemy ships as well as a bonus ship that hovers above the enemy ships. The score updates itself, as more enemy ships are killed. The hero has three lives and after losing all the lives the game ends. However, if the hero is able to shoot all the enemies the game moves on to the next round.

Background

The application is written in java (JDE 1.6) using the Eclipse IDE. The game is written for Blackberry device JRE 6.0.

Classes

Display Class

This class is primarily used for displaying message screen for the about and instructions button on the main screen. This class extends the main screen class and allows the user to return to the main application.

Space Invaders Class

This class is the entry point to the main application. It pushes the menu screen into the fore front, so the user can make the selections. This class extends the UIApplication class.

Space Screen Class

The Space screen class extends the Main Screen. It provides a layout for the user to select the various menu options as well as to start the game. If the user selects the game starting option, the user is taken to the Game class, once the game is over, the game ends and the memory is reclaimed.

Game Class

The Game class extends the full screen. This game is the main driving force behind the game. In the game constructor, the class creates the application background, creates the main vector for storing all the objects, processes the game level, starts the game sound and launches three threads.

Enemy Thread: The enemy thread is responsible for controlling the movement of the enemy drones.

Bonus Thread: The bonus thread is responsible for controlling the lateral movement of the bonus drone.

Refresher Thread: This thread constantly polls to see, if there has been any collisions between objects, if so, it will remove the item from the main vector. The game also checks to see if the hero object has been hit more than 3 times, it will then stop the game and launch a game over dialog. It will also take the user to the next level automatically if all the enemy drones have been killed.

The Refresher thread is also responsible for recreating all the objects by redrawing using the invalidate() function.

Obj Class

This class is primarily responsible for keeping track of objects like the drones, bonus, hero, and the fire ball. The drones, bonus and the hero object each have lives, velocity and thinking capabilities. These values are set using the overriding operation of the object. The enemy object is given random shooting capabilities; however the hero shooting capability is not over ridden, and is instead controlled by the keyboard. The Damage routine controls the life of the object. There are two important routines in the class, the collision detection and the cleanup routine. The collision detection routine, basically looks to see if any two objects, who type is filtered, are within each other's x-position overlap or y-position overlaop. If the bounding boxes are over lapping, the application realizes it is a hit and calls the damage function. The cleanup routine, basically goes through all the objects in the vector to see , if they are out of the game screen or have lost sufficient lives, if so, it will remove them from the game vector.

SND and GFX Classes

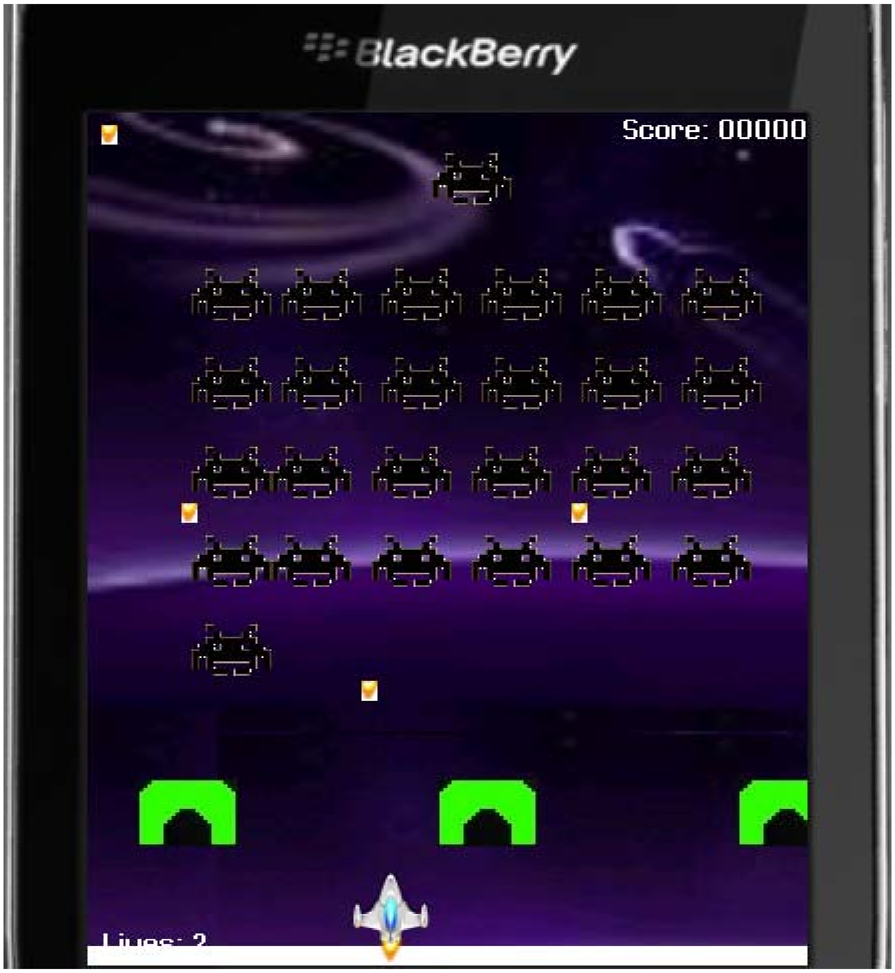
These two classes are primarily responsible for running the sound in the background and for paint operation. The GFX class also draws the live score.

Challenges Encountered

This was my first handheld device application, it was challenging at first, trying to figure out where to start! Positioning the menu screen was tricky, but once I understood the layout functions it become a lot easier. The threading concept was a little difficult, it was hard to get all the objects working side by side at first, but once the object class was written, it turned out to be easier to control them in a thread safe manner. I looked around to find some good images for the enemies and hero object, and then manipulated them using paint application to the size desired, it took a few times before I could make them look desirable. Apart from that, the rest of the application was just trial and error, before everything started working as required.

Conclusion

The class was exciting and I learned a lot about handheld device programming. The application itself taught me how making games isn’t as difficult as I had initially envisioned. I initially designed the look and feel of the application, and then architected some of the classes. I worked on each class separately, and then brought them all together. The final application worked well in the demo and I felt satisfied.


BlackBerry
Score: 00000
Lives: 2