Asteroid Destroyer Report

Mobile Technology: MOD002663

Academic Year: 2016/17

Semester: 2

SID: 1409046

**Contents**

Introduction……………………………………………………………………………………3

Aims & Design ………………………………………………………………………………..3

Implementation Description……………………………………………………………4

Testing……………………………………………………………………………………………6

Critical Evaluation……………………………………………………………………………7

Conclusion………………………………………………………………………………………7

**Introduction**

Over the past twelve weeks I have completed an ‘asteroid destroyer’ game in a new language to me, javaScript. Asteroid destroyer games have been classics for decades, with Atari releasing the original ‘Asteroids’ in 1979 (Atari, 1999) as a cabinet arcade game. Since its original release, it has been ported to many different platforms, and has become the inspiration for many other games, including the one I have been making for this module. I have been creating the game in an online editor called ‘Plunker’ (Plunker, 2006), this editor allows the use of the ‘AngularJS’ (Google, 2010) framework, so I have also been using this to do most of the ‘leg work’ before I have started coding on the stage. This document will contain the various mobile and web features I have implemented, and will outline the design decisions I made and why I made them. The implementation of these features will then be highlighted,, and I will also demonstrate the various tests I undertook to make sure the game was completely playable and how I wanted it to look. At the end of the document there will be a critical evaluation to summarise success and potential areas which could be developed in the future. The deliverables for this project include the game described below, and this report that will look in more depth at the components described in this introduction.

**Aims & Design**

The basic principle of the game is to fly around the stage and destroy as many asteroids as you possibly can without colliding with any. The game will get progressively harder as more asteroids enter the stage. The ships rotation will follow the movement of the mouse, and on touching an area of the screen, it will fly towards that area until it reaches the top corner of the stage. This movement will occur until there is another command from the user, in which case it will head towards that area of the stage. Shooting from the ship will be automatic and will always fire out of the front of the ship. The levels will start with larger asteroids, but as the user shoots at these asteroids, they will become smaller and will move at a quicker pace than the larger ones. Asteroids wrap around the screen edges, so if it was to hit the left wall it would re-appear on the right-hand side of the stage at the opposite Y value, so that the movement has a fluid appearance. The player has 3 lives when the game starts, when all three lives have been used the player loses. When this occurs his score, which is calculated by the number of asteroids destroyed, is stored in a leader board.

A lot of the justification for the design came down to me trying to make the game feel like the original Asteroids game. This includes the rotation of the ship towards the mouse, the larger, slower asteroids at the start of the game, the wrapping effect for the asteroids and the 3 lives system. These elements, I feel, were the core design features that make an Asteroids game. The automatic shooting feature I will implement as the controls may be getting too complicated for the mobile device, and this is an effective way of cutting down the confusion for the user and will allow them to concentrate on destroying and avoiding asteroids.

The game will continue to play as long as the player can last on the stage without losing all of their lives, so technically there is no complete winning state, but as the player reaches 10 score, there will be no more asteroids appearing when shooting them. When the last asteroid is destroyed and score is above 8, a new level is instantiated and the local score will be reset to 0, the global score will continue to count as each of the asteroids is hit. The global score will be displayed when the player enters the fail state. This fail state is activated when the player loses all of their lives.

The UI for the game consists of a dashboard which is always available to the player at the top of the screen, this will allow them to navigate between the leader board and the game itself. The other main element of UI within the game will be the input of the player’s name, this will be available underneath the dashboard within the game scene. This will be accompanied by a button which will store the name variable when clicked. There will be no audio within this game, as the audio tracks that I have listened to online are not appropriate and I do not have the skills to produce my own audio / sound effects.

As with any clone of the classic ‘Asteroids’ from Atari, this game is within the arcade genre. The fact the asteroids game falls within this genre was a deciding factor when choosing the type of game I wanted to make. This is because I am most interested in this genre, and many games that I play are classed in the arcade genre. Beyond this module and degree, I will be looking to explore careers in mobile arcade game development. The target audience for this game will be mobile users aged between 5 and 35, I have set the maximum age of players this high as the game can still entertain older gamers through nostalgia. For many older gamers, the original Asteroids game could have been one of the first computer games they played. This leads me to believe the target audience for my game will be mainly males, due to the stigma that was attached to video games during that generation, but will not be exclusively for men, as women now make up 42% of the gaming demographic in the USA (Statista, 2017).

A use case diagram for the asteroid game is available at appendix 1.

**Implementation**

As stated in the introduction, the web software used to create this project was Plunker, this allowed me to use the rendering engine of Pixi.js (Groves & Goodboy Digital, 2013) which was used for the animation of the asteroid, bullets and the player movement. A continuing issue I had with the use of Plunker, was that all the code to control the page was on one script, and this made navigating to a certain part of the code difficult as the project expanded and the script got increasingly longer. Another issue I had with the web software was that when I originally loaded the game using a different browser outside of chrome, the game would no longer be playable. An additional piece of external software that I attached to the game, on-top of the use of Pixi, was the framework called ‘Bootstrap’. This allows the game area to be readjusted to the size of the browsers window if the player changes it during the playing session, this also allowed the game to readjust within other browsers. In terms of assets, the asteroid asset was retrieved from the web and stored at imgur.com, which is then accessed by the code as to not have to store the asset locally. This is the same for the bullet image. The ship component is created within the controller script, and is redrawn each time the player moves the ship within the stage.

Some of the main issues during the development for this project include the movement of the ship to where the player touches on the stage. This was originally set to work with the arrow keys on the PC, but there is no functionality for this on the mobile device. This lead me to setting the movement depending on where the user touches, meaning that the rotation of the ship could now no longer be set to where the player has the mouse – there is no functionality for this on mobile device. To fix this problem, I sourced some code from another project produced by Dinita Razvan (Razvan, 2017), the link to this project is in the reference list. I then modified this code to suit the needs for my project. Another issue I faced in the development, was the ‘splitting’ of the asteroids when the player hits the larger asteroids. This was an issue as when the player was hitting the asteroid before, it was just bouncing off and adding score but not adding any more asteroids to the stage. To fix this problem I removed the asteroid that was getting hit by the bullet and spawned two new ones in the place of the collision, the two new asteroids will then have opposing velocities to simulate the ‘splitting’ of an asteroid. The code that does this is displayed below:

var

// position for the first new asteroid

x1, y1,

// position for the second new asteroid

x2, y2;

// determine x positions of the two new asteroids

if (this.x + this.width >= stageWidth) {

x1 = stageWidth - this.x - this.width \* 2;

x2 = stageWidth - this.x - this.width;

} else if (this.x - this.width <= 0) {

x1 = this.width \* 2, x2 = this.width;

} else {

x1 = this.x - this.width \* 2, x2 = this.x - this.width;

}

// determine y positions of the two new asteroids

if (this.y + this.height >= stageHeight) {

y1 = stageHeight - this.y - this.height \* 2;

y2 = stageHeight - this.y - this.height;

} else if (this.y - this.height <= 0) {

y1 = this.height \* 2;

y2 = this.height;

} else {

y1 = this.y - this.height \* 2;

y2 = this.y - this.height;

}

A gameplay issue that I encountered during the implementation, was that the game was too difficult with just the three lives, I found when players were testing the game that they would often not make it past the first level. This was a problem as it would frustrate players, and make them not want to continue to play the game. To fix this issue I added 7 more lives to the players bank of lives, bringing the total to 10. I also decided to change the ability to fly off the stage and the continued moving in the same direction as first command. This was because during testing, there were many bugs arising concerning the rotation of the ship and the direction the bullets were heading in, but I also think it adds a new dynamic to the game which can be even more challenging to the user at times.

**Testing**

The main testing for this project was done during the development of the project, what I have outlined below is the final round of testing that was conduct when the game had all the components added to it. Some of the main changes that have come as a result of the testing period include; the lack of change in ship positioning when the ship collides with an asteroid. In the original build, the ship would reset to the starting position when the player collides with an asteroid, I have removed this component, as during the testing there was issues when there was an asteroid in the place of respawn. This meant that the player would go through a loop of dying and respawning instantly - until no lives were left to take and the game would end. This has now changed so that when an asteroid hits the ship, it will bounce off the player and reduce the health value. To ensure the narrative makes sense I have changed the name of the health value to ‘Shield Level’, as a hit from the asteroid is taking away some of the shield strength.

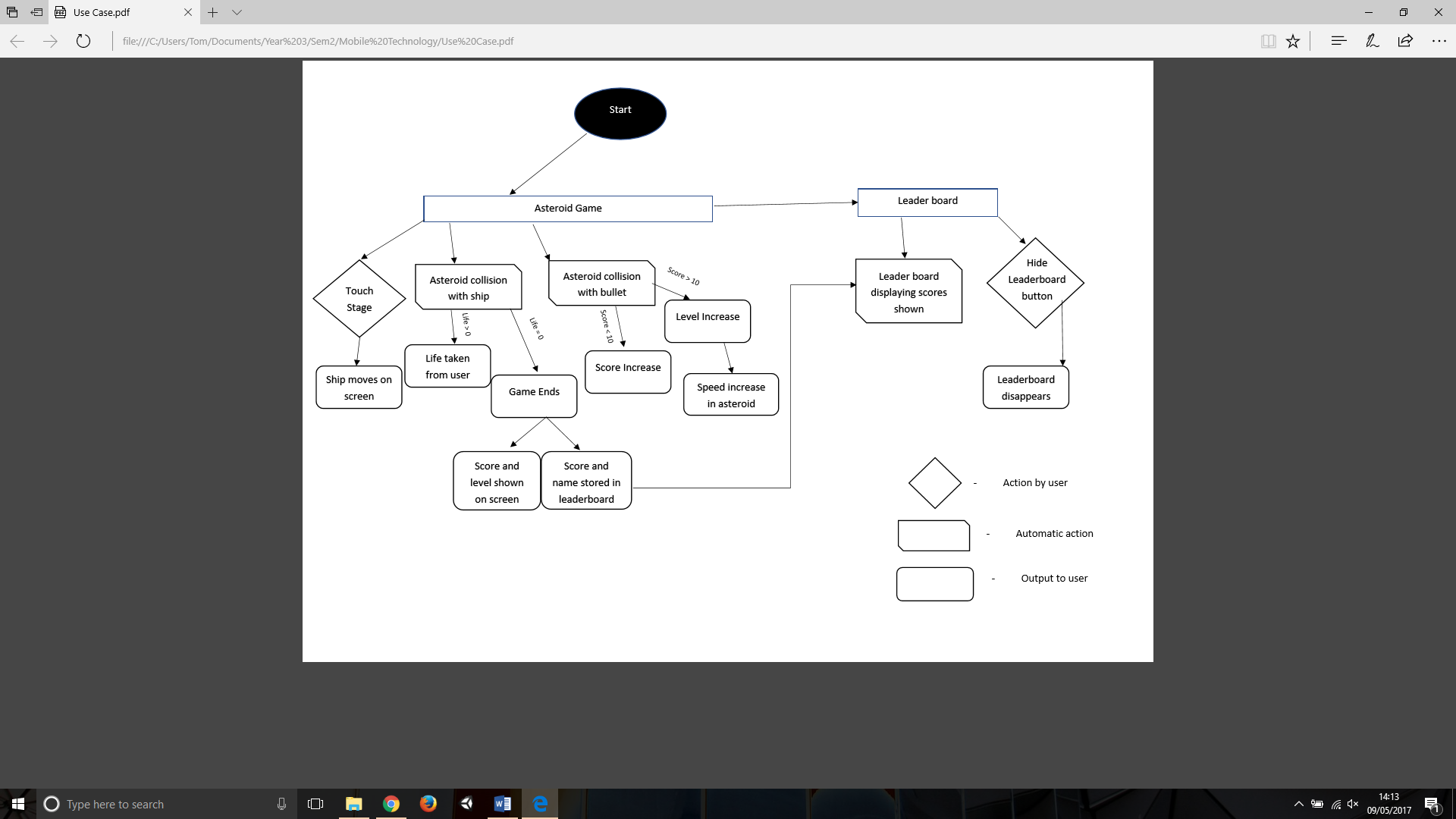
**Critical Evaluation**

Some of the highlights for this project include the splitting of the asteroid and the sequential level progression, the piece of code that handles this has been part outlined in the implementation. Accompanying these lines of code, there is a structure for marking the asteroid as a ‘copy’ or not, if the asteroid is a copy then it will not spawn any extra asteroids when it is hit. Using this set up, when the player reaches a score of 8, all the new asteroids spawning will be copy’s meaning the amount of them on the screen decrease. When the player has hit the last remaining asteroid, then the next level will start. This acts as positive re-enforcement for the user, allowing them to see some progress within the game. This is important as I found the long single level until death can seem tedious and feel unrewarding. Another highlight is the movement of the asteroids using the wrapping effect, this is the effect I think of as the most defining design feature of an Asteroids game, so it was important to get this into my iteration of the game and working correctly.

Some weaker areas of the game that need addressing include the fact that sometimes there is a ‘left over’ asteroid that was the last to be destroyed from the previous level. Unfortunately, this asteroid still has the functionality of a new asteroid apart from the movement, this means that the player can lose health and gain score by hitting the asteroid with the ship or bullets. Another aspect that needs improving is the speeding up of sprites when the player navigates between the instruction page, leader board and the asteroids page, this is due to a memory leak between the destroying and reloading of the scene. The bug is not so noticeable if the player stays in the game as the asteroids are speeding up as levels progress but can make the first level difficult if returning to the game from another page.

**Conclusion**

Overall, the success of the project can be seen in the quality of game produced, there are many good features implemented within the project that lead to a game that is enjoyable to play, and there is enough challenge to draw the player back. For future developments, I would look to expand the project to work effectively on all browsers, as there is still some issues with text not appearing on IE. I would also look to deploy the game as an offline package, as the game is not playable without internet connection and is difficult to navigate when on GPRS. For feature development, I would look to include an alien spaceship that shoots at the player and pickups that add to player lives.

**Appendix 1**

Use Case Diagram for Asteroid Game

Plunker project URL:

https://plnkr.co/edit/xoUPTY

# References

Atari. (1999). *Production Numbers*. Retrieved 03 17, 2017, from http://www.atarigames.com/atarinumbers90s.pdf

Google. (2010). Retrieved from https://github.com/angular/angular.js (Repository), https://angularjs.org/ (Website)

Groves, M., & Goodboy Digital. (2013). Pixi.js. Retrieved from https://github.com/pixijs/pixi.js (Repository), http://www.pixijs.com/ (Website)

Plunker. (2006). Montreal. Retrieved from https://plnkr.co/

Razvan, D. (2017). Mobile Technology - Week 9 - Dungeon Hunter - Character. Retrieved from http://plnkr.co/edit/Mx3RwMuFoECf5ivWrKyz?p=info

Statista. (2017). Distribution of computer and video gamers in the United States from 2006 to 2017, by gender. Retrieved from https://www.statista.com/statistics/232383/gender-split-of-us-computer-and-video-gamers/