

## **Game Document Group Project: G51FSE COURSEWORK 2014**

- ☐ Barnabás Forgó
- ☐ Oliver Gee

### **Game concept:**

**1.1** The goal for the project was to create an inventive 2D game at a professional standard of quality, for this we produced a game loosely based on the traditional sci-fi spaceship "shoot em up" style. Involving the player controlling the actions of an avatar being able to shoot and damage enemies in order to progress. The player's ship has the ability to also be killed through collision with object such as asteroids. On collision the asteroid will disappear but the player loses a life; Loss of all life's results in game over displaying their score and allowing the player to start from the beginning again however their score will become saved, allowing the player to compete against their self's and friends.

### **Requirements Definition and Specification**

#### **Functional requirements**

##### **Menu priority order:**

- 2.1.1** The system must display a main menu for the user before the main game is played
- 2.1.2** The system's menu must display a selection for the user to quit the program, play the main game and move to a high score page
- 2.1.3** The system's menu must clearly show what the user is currently selected
- 2.1.4** The system's highscore page must display the highest top ten scores saved
- 2.1.5** The system's highscore page must display a clear way to return back to the menu page
- 2.1.6** On selection of the "Highscores" option on the menu the menu should be closed and the highscore page should be opened
- 2.1.7** On selection of the "Start Game" option the menu should close and start the main game sequence

##### **Main game priority order :**

- 2.1.8** The main game should display the controls before the game starts
- 2.1.9** The main game should display the level number before each round starts
- 2.1.10** The main game must have a space related background
- 2.1.11** The player's current score should be shown at all times
- 2.1.12** the players number of lives should be shown at all times
- 2.1.12** The system should be able to pause the game

##### **Avatar priority order:**

- 2.1.12** The system should allow the user's avatar to move up using the w key
- 2.1.13** The system should allow the user's avatar to move down using the s key
- 2.1.14** The system should allow the user's avatar to move left using the a key
- 2.1.15** The system should allow the user's avatar to move right using the d key
- 2.1.16** The system should allow the user's avatar to be rotated forward relevant to the position of the cursor on the screen
- 2.1.17** The system should allow the user's avatar fire a projectile on the left click of the mouse

**2.1.18** The projectile from the ships avatar should follow a straight path from the direction it was "fired"

**2.1.19** If the players avatar collides with an asteroid then the player should lose a life the collided object removed and classed as destroyed producing no further asteroids in this case

**2.1.20** If the projectile collides with an asteroid then the asteroid is removed, if it is a parent asteroid then child asteroids should be created

**2.1.21** If the projectile collides with an asteroid the relevant score increase should be made

**2.1.22** If all asteroids are removed from the game the next level should start after a brief warning

**2.1.23** On the asteroids generation they should have a randomly generated path of direction and should not generate with a suitable radius of the players avatar

**2.1.24** If an asteroid moves outside the screen, it should appear from another side of the screen retaining its direction of travel

**2.1.25** If the player loses all their lives a game over screen should be displayed and their score saved

**2.1.26** On firing of a projectile a firing sound should be played

**2.1.27** On collision of asteroid and avatar a colliding sound should be played

**2.1.28** On starting of the game a background theme should run in loop

**2.1.29** Pressing R should restart the game

**2.1.30** Pressing "Esc" should pause the game"

**2.1.31** When a level is finished the preceding level should contain 3 more fully sized asteroids than the last

**2.1.32** The player's avatar should not be able to move outside of the screen

**2.1.33** Pressing "q" while paused must return the player back to the main menu screen

### **Non-functional requirements priority order**

**2.2.1** Starting the program should take no longer than a second

**2.2.2** Changing to either from the main menu to the leaderboard or back should be near instantaneous

**2.2.3** Starting the main game should take no longer than a second

**2.2.4** Main menu font should be of a consistent size and colour

**2.2.5** All text throughout should be clear and readable

**2.2.6** The game's mechanics should be understood easily

**2.2.7** The game's controls should be unambiguous and consistent with other games

**2.2.8** The game's sound must be of "good" quality

**2.2.9** The game's sprites and images must show basic detail and look presentable and clear throughout game play

**2.2.10** The game's difficulty must increase as level gets higher

**2.2.11** The system should be responsive to the players actions with no noticeable lag time

**2.2.12** There should be no delay time in pausing and unpausing the game

**2.2.13** All game screens should not put the user under any cognitive stress and keep to a minimalist style

**2.2.14** All images, sprites, fonts and sounds should be representative of the space theme

## Requirements specification

Requirements specification	Linked requirement definitions
The system will loosely based on the traditional sci-fi spaceship “shoot em up” style	2.2.14 2.1.10
The system will have a main menu allowing users to move around options in the game such as highscore	2.1.1 ... 2.1.7 2.2.2
The game will have a method of storing and presenting a high score's to the user	2.1.2 ... 2.1.6
The game should emit audio for relevant actions	2.1.26 2.1.27 2.1.28 2.2.14 2.2.8
The game should have a small tutorial outlining the controls of the game and what level the player is currently on	2.1.8 2.1.9 2.2.7
The game should have multiple levels	2.1.9 2.2.10 2.1.22
The game's avatar should be able to move	2.1.12 ... 2.1.16 2.1.32
The game's avatar should be able to fire a projectile	2.1.17 2.1.18 2.1.20 2.1.21
The game should be able to win or lose and indicate this	2.1.12 2.1.19 2.1.20 2.1.22 2.1.25
The game should run quickly; having fast loading times and no lag time	2.2.1 2.2.2 2.2.3 2.2.11 2.2.12
The system should be readable and clear at all times, avoiding cognitive stress	2.2.4 2.2.5

	2.2.9 2.2.13
The game should be pauseable	2.1.12 2.1.30 2.2.12
The game should become harder as time progresses in one play through	2.2.10 2.1.31
The main menu should be to be returned to if the user wants/needs	2.1.33

## Game Design:

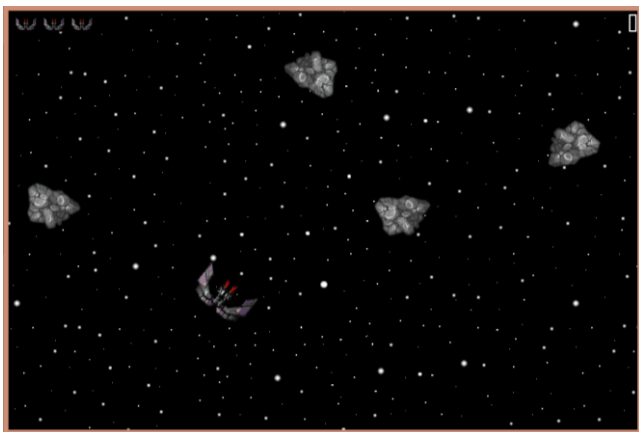
### Interaction design/structure:

The game will open with a small menu option allowing for the user to select either to play the game, to see the high scores. From the highscore page the user should be able to either quit or return to the main menu. The main menu and the highscore page will have the same style retaining font, font size, background and sound to keep consistency throughout the game.

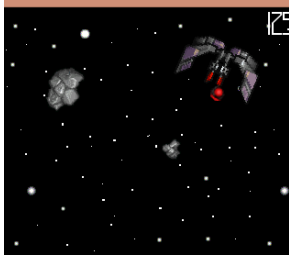


From selecting to play the game from the main menu, the menu would close only displaying the game to reduce the amount of information visible at one time. A small tutorial will be displayed before anything else to reassure the user of the controls then the game will start.

This player's avatar would start in the middle with the number of life clearly visible in the top left, this will decrease dynamically as the game progresses along with the score that will be shown in the top right. On loss the game will display a prompt, this will be the same for a victory however in a different colour to symbolise victory from defeat allowing the user to have a better understanding on what is occurring in the system. If the user pauses the game then another prompt will be shown; however this time the prompt will be flashing in a good pace to remind the user what they have done and why they would not be able to move their avatar etc. Throughout the game the same background will be visible, traditional of the archaic design whilst retaining a consistent image.



### Scoring system:



On initial load up of the game the players score will be set to zero and given three lives. Hitting the largest asteroid with a projectile will earn the player 25 points, the rock will split producing two smaller ones. Hitting these will give the player 50 points, on collision again the rocks will split producing each two smaller rocks that will award 75 points but not produce further rocks. The scoring rewards the player for shooting the smaller asteroids due their size makes them harder to hit and avoid in large groups as each will have its own path of direction.

On completion of a level the score will be retained to the next round, this is repeated until the user has no lives left. In this eventual case since the game has no fixed ending the score is saved in which it is compared to the current standing scores. If the new score is greater than one of the ten that will be displayed in the display page then the new score is placed in the correct location and the lower score shuffled along.

### Rules and controller behaviour:

```
class LevelChangeOverlay(TextOverlay):
    def __init__(self, level, game):
        TextOverlay.__init__(self, "level " + str(level), GREEN)
        self.level = level
        self.tutorial = [TextOverlay("wasd to move", GREEN, 50),\
                         TextOverlay("use the mouse to aim", GREEN, 50),\
                         TextOverlay("press the left mouse button to shoot", GREEN, 50),\
                         TextOverlay("esc to pause", GREEN, 50),\
                         TextOverlay("then hit q to exit to the main", GREEN, 50),\
                         TextOverlay("press r to restart", GREEN, 50)]
        self.fight = TextOverlay("fight", RED)
        self.game = game
        self.frameNum = 0
```

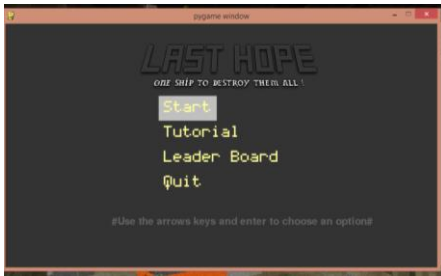
For the user to navigate through the menu they must use "w" to move upwards through the selection and "s" to move downwards. Using the left mouse click to register their selection.

For moving their avatar the controls have been made the

same to give the user ease of life making them simple and easy to use. "WASD" to move the avatar, mouse to aim and left click to fire. These control are common for this genre and will allow the user to pick up and understand the game quickly preventing stress from not understanding the controls.

As mentioned in the small tutorial "Esc" would pause the game and pressing "q" while paused will return the player to the main menu. The ship will not be able to leave the screen, preventing the user from effectively losing control of their player causing frustration and preventing the game from crashing/bugging. Asteroids cannot also generate on top of the ship at the start of a new level creating instant loss of life to the user and would be considered unfair, this is also the same in the case an asteroid collides with the player the asteroid will not create its subsequent smaller forms if possible as they would instantly also cause further loss of live to the user.

## Evaluation of prototype and design refinements:



Through spending a lot of time getting a strong prototype together we presented the foundations of the game with the promise for more features such as a shop, unlockables, skins and bosses. However, under review it was decided that it just wasn't feasible given the time constraints to meet such targets, opting for quality rather than quantity. Realising we had a bad case of feature lust for things that were just not needed for the finished product, we added to the display that

was already partially set up and creating a smart, easy-to-use UI. Despite producing a good prototype, we did find issues such as some of our original sounds were too loud and annoying to the player, and that our hit detection and pathfinding of the asteroids was a bit off and some would become stuck or float around the edges. The skin of the player's main avatar also had to be redone in order to make the projectiles fire more realistically, as well as creating a moving background that made the game feel less static and more engaging to a user.



## Implementation notes and description of the development methodology:

### Implementation:

In our coding practice we ensured that we used an object-oriented model, breaking down our problem into its constituent parts:

- Main - Used for the starting of the program, initializing the game
- Control - The core of the game, holds the main game loop and actions

### Classes:

#### Background

- Loads the image for the background, and sets the container
- Updates the movement of the background image

#### TextOverlay

- Creates new instance of the text overlays
- Supports the flashing/pausing
- Draws the overlays to the screen

#### LevelChangeOverlay

- Creates overlay for opening level details
- Draws overlays to the screen

#### ScoreDisplay

- Takes in the score value, creates font display
- Outputs score to screen

## LivesDisplay

- Takes in the life value, display number of images relevant

## Game

- Initializes the variable in the game e.g score
  - Increments variables each level e.g Number of asteroids
  - Processing events e.g shooting, moving
  - Checking for collisions
  - Updating the position of sprites
  - Updates scoring
- Audio - Created the methods used to start and stop sounds
  - Player & enemies- Holds information on the players character & enemies
  - Constants - For colours etc
  - Menu & Highscore - produces the menu and presenting the scores

From this we were able to specify what jobs were needed to be done, and how they would be implemented into the system, we used camelcase to present our variables and us lower case for methods and a capital letter for a class allowing for consistent code that code be read and understood throughout the program. Whilst commenting on changes and complex areas of the code to make it easier for someone else to see what was changed and how it works.

Using github we were able to control versions, rollback and ensure we both where working from the most up to date code so that there was no inconsistencies that would have slowed down progress having to merge code together. It also allowed us to see what needed to be done and what was left to be finished or improved. Storing our code over the internet allowed for fast accessibility increasing overall productivity from allowing lots of small continued updates whenever was easiest for us

## Debugging

We found the easiest way to debug our program was through the use of the python IDE that on error would display the area the bug happened, this saved alot of time preventing the manual search through the recent entries from last test. Another method was to use breakpoints to step through code to line by line to identify where the issue was coming from.

## Methodology

We found that our methodology was best compared to the spiral model. This was mainly because we would outline the requirements the user wanted, design how we would implement these needs into the system then create a new prototype displaying the change before finally reviewing the system again. Because of this we found that our work was constantly being altered from the changing requirements as we understood more about what the user would want. An exsample of this was identifying new issues such as an asteroid generation issue causing “unfair” deaths to the player that we were able to recognise from playing the prototypes and resolve early.

Despite this we did use some agile methods such as pairs programming in order to refactoring part that did not meet the mark often leading to an increase in the fps of the system from discussing face to face what areas could have been changed to become more efficient whilst reducing the number of errors in writing new code. However late into the project the effects of this became lessened as we came to a point where we knew what was needed to satisfy our requirements and the main concern was doing it efficiently

#### Evidence of testing:

##### Test Suit document copy

Test case ID	Description	Pass/Fail
1.1.1	Pressing “w” moves players ships upwards	Pass
1.1.2	Pressing “s” moves players ships downwards	Pass
1.1.3	pressing “a“ moves players ships left	Pass
	pressing “d” moves players ships right	Pass
1.2	Moving the mouse creates a rotation effect on the ship	Fail only moving in right angles
1.2.1	Moving the mouse creates a rotation effect on the ship	Fail spiralling out of the screen
1.2.2	Moving the mouse creates a rotation effect on the ship	Pass, used a bass center and rect to control rotation
1.3	Clicking the left mouse key fires projectiles	Fail image appearing , not moving
1.3.1	Clicking the left mouse key fires projectiles	Pass changed to the position the mouse was at one the screen
2.1	Generation of one asteroid	Pass, needs to spin only moving in2 liner paths



<b>2.2</b>	<b>Generation of one asteroid moving and spinning</b>	<b>Pass however to fast</b>
<b>2.2.1</b>	<b>Generation of one asteroid moving and spinning</b>	<b>Pass, moves at a good pace that is fair</b>
<b>2.3</b>	<b>Ship cannot move outside of the screen, asteroids can</b>	<b>Pass</b>
<b>2.4</b>	<b>Collision with projectile causes asteroid to disappear</b>	<b>Fail, game crashes</b>
<b>2.4.1</b>	<b>Collision with projectile causes asteroid to disappear</b>	<b>Fail, passes through</b>
<b>2.4.2</b>	<b>Collision with projectile causes asteroid to disappear</b>	<b>Fail, only the projectile is being removed</b>
<b>2.4.3</b>	<b>Collision with projectile causes asteroid to disappear</b>	<b>Pass both are removed</b>
<b>2.5</b>	<b>Collision causes score to increase</b>	<b>Pass score updated</b>
<b>2.6</b>	<b>collision causes smaller asteroids to be made</b>	<b>Fail, Not generating in correct location, won't move either</b>
<b>2.6.1</b>	<b>collision causes smaller asteroids to be made</b>	<b>Pass, smaller asteroids getting stuck though</b>
<b>2.6.2</b>	<b>collision causes smaller asteroids to be made to move</b>	<b>Pass pathfinding improved</b>
<b>3.1</b>	<b>Collision between player and asteroid loses player a life</b>	<b>Pass</b>
<b>3.1.1</b>	<b>Collision between player and asteroid loses a life and the screen is updated to show this</b>	<b>Pass, life indicator moves down one</b>
<b>3.1</b>	<b>When player moves the background moves slightly</b>	<b>Pass, however causing a clipping issues when done too much</b>

3.1.1	removed clipping issue	pass ,boundaries added
4.1	Moves to next level when asteroids are removed	Pass, score not being carried, lives get reset ?
4.1.1	Moves to next level when asteroids are removed	Pass, variables are passed over to next phase now
4.2	Display shows victory in case of level completion	Pass
4.3	Display shows loss in case of lives hitting 0	Pass
4.4	Restart allowing for player to start again	Fail, carries lives over causing crash
4.4.1	Restart allowing for player to start again	Pass
4.5	pressing Esc pauses game	Fail player can still fire ? but projectiles don't move
4.5.1	pressing Esc pauses game	Pass, added boolean to check the if paused
4.6	Will display flash on and off	Pass
4.7	Will Background theme play on start, is it good quality	Fail, path not found
4.7.1	Will Background theme play on start, is it good quality	Fail, file type issue
4.7.3	Will Background theme play on start, is it good quality	Pass !
4.8	Will sounds play for firing and death	Pass
5.1	Menu appears on start	pass
5.2	User can move their selection to all possible options	pass
5.3	Selecting quit will close the game	Pass

5.4	Selecting highscore option will close the menu, open high scores	Pass
5.5	Highscore menu displaying highscore	Pass, names not working correctly
5.6	Names display on highscore ?	Pass passed over correctly now
5.7	Can user return to menu from highscores page	Pass
5.8	Selecting main game option from the menu will start the game	Pass
5.9	Is there a way to return back to the menu ?	Fail ,”Esc” key not working
5.9.1	Is there a way to return back to the menu ?	Pass, user sent back to main menu
5.11	Beating highscore correctly updates	Fail, wrong order
5.11.1	Beating highscore correctly updates and sort scores correctly	Pass

### Evidence of testing

SpeedoDevo / last.hope

65 commits 1 branch 6 releases 2 contributors

branch: master last.hope


I think now we are rear production ready


SpeedoDevo authored 14 hours ago latest commit b972b78b37


File	Commit Message	Time
Remastered Tyrian Graphics	added sprites i used for graphics	2 months ago
__pycache__	i think now we are rear production ready	14 hours ago
asd	pickling higyscores making it untweakable	14 hours ago
data	hey found a menu base added some stuff	2 months ago
image	proper main menu, brief tutorial, fixed asteroid stuck out of screen	5 days ago
last.hope	pickling higyscores making it untweakable	14 hours ago
sound	proper main menu, brief tutorial, fixed asteroid stuck out of screen	5 days ago
Scores.txt	nickling higyscores making it untweakable	14 hours ago

1.1 Github screen overview: Showing contributes and changes of the game



## 1.2 Github post screen, showing bug reporting and changes


**pickling higyscores making it unweakable**  
SpeedoDevo authored 14 hours ago  
[Browse](#)


**added name entry, working on rewriting highscore filesave**  
SpeedoDevo authored 16 hours ago  
[f594d327](#)  
[Browse](#)

**rewritten highscore table sorry :/**  
SpeedoDevo authored 16 hours ago  
[8622f95a](#)  
[Browse](#)

**May 08, 2014**

**i think i bugged somthing fixed now**  
oligee authored 4 days ago  
  
[2c19dcf2](#)  
[Browse](#)

**add score display using menu style**  
oligee authored 4 days ago  
[83804aec](#)  
[Browse](#)

**added the score saving**  
oligee authored 4 days ago  
[eff729d5](#)  
[Browse](#)

**May 07, 2014**

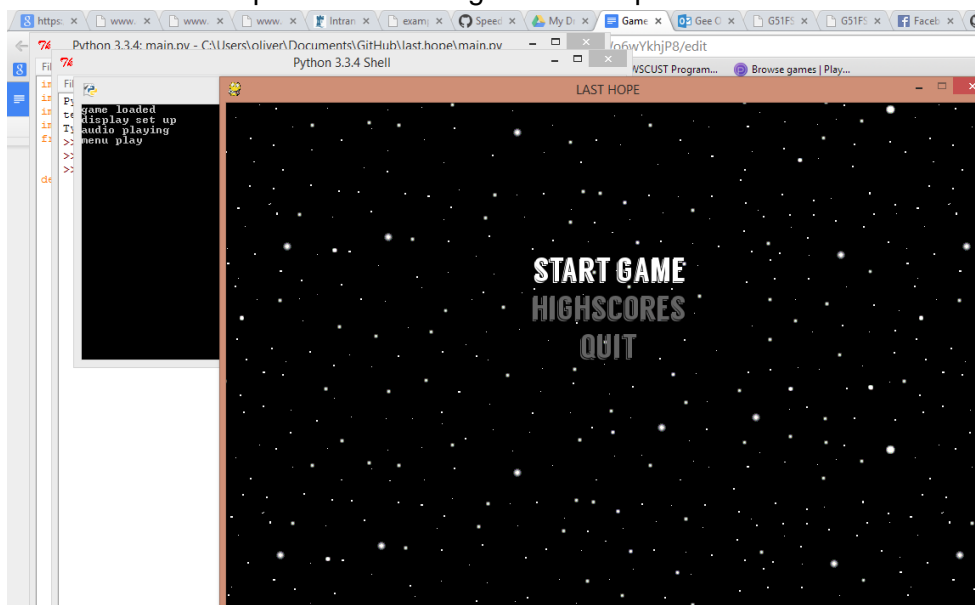
### 1.3 Code changes

```

154 154         # score = 0
155 155
156 156     def __init__(self, mainMenu):
157 157         # self.score = 0
158 158         self.score = 0
159 159         self.paused = False
160 160         self.gameOver = False
161 161         self.written = True
162 162         self.allSprites = pygame.sprite.Group()
163 163         self.lazers = pygame.sprite.Group()
164 164         self.enemies = pygame.sprite.Group()
165 165
166 166     def __init__(self, mainMenu):
167 167         self.winScreen = TextOverlay("victory", GREEN)
168 168         self.victory = False
169 169         self.levelChange = True
170 170
171 171         self.audio = audio.Sounds()
172 172         self.score = 0
173 173
174 174         self.asteroids = 1
175 175
176 176         self.level = 1
177 177
178 178         self.levelChangeOverlay = LevelChangeOverlay(self.level, self)
179 179         self.scoreDisplay = ScoreDisplay(self.score)

```

#### 1.4 Command line prints to show regions of completion



[illegible]

Points of consideration:

- Summary:

Despite some of our failures to add extra features, i believe we made the correct decision to focus on producing a strong well coded aesthetically pleasing product rather than rushing to cut corners producing a game that would have likely been allot less refined and probably damaging N2 games reputation. Causing them to waste further time and money in the future handling complaints because we missed out validations and checks. However with more time we would love to continue working on the project to add the extra features we originally wanted.