CST8333

Assignment 1

Space Guy: Blockade

Andrew Gaal

February 10, 2022

All material prepared for this assignment was produced by the author. Material from all third parties has been cited and referenced.

Table of Contents

[1 Introduction 2](#_Toc94725328)

[1.1 Objectives 2](#_Toc94725329)

[1.2 Scope 2](#_Toc94725330)

[1.3 Timeline 2](#_Toc94725331)

[1.3.1 Milestones and Deliverables 3](#_Toc94725332)

[1.4 Risks 3](#_Toc94725333)

[1.5 Assumptions 4](#_Toc94725334)

[1.6 Technical Constraints 4](#_Toc94725335)

[1.7 Budget 5](#_Toc94725336)

[Requirements 6](#_Toc94725337)

[1.8 Introduction 6](#_Toc94725338)

[1.9 Business Requirements 6](#_Toc94725339)

[1.10 Technology Requirements 8](#_Toc94725340)

[1.11 Compliance Requirements 10](#_Toc94725341)

[1.12 Requirements Traceability Matrix 11](#_Toc94725342)

[2 Feasibility 12](#_Toc94725343)

[2.1 Introduction 12](#_Toc94725344)

[2.2 Alternative Solutions 12](#_Toc94725345)

[2.3 Recommendation 13](#_Toc94725346)

[3 Design 15](#_Toc94725347)

[3.1 Introduction 15](#_Toc94725348)

[3.2 System Architecture 15](#_Toc94725349)

[3.2.1 System Requirements 15](#_Toc94725350)

[3.2.2 Logical Architecture 15](#_Toc94725351)

[4 References 17](#_Toc94725352)

# Introduction

In this section background information on the project is provided, including the reasons for undertaking the project, specifically the business problem to be solved and how the proposed system will solve it, as well as the key stakeholders who will benefit from the project results.

## Objectives

In this section measurable project objectives, business outcomes to be derived from achieving the objectives, and the measurement criteria to be used to confirm that an objective and the outcome have been achieved are listed.

Table 1: Objectives and Business Outcomes

| **No.** | **OBJECTIVE** | **BUSINESS OUTCOME** | **MEASUREMENT CRITERIA** |
| --- | --- | --- | --- |
| 1 | Deliver a finished piece of software written using Python by the end of April 2022. | Developer learns foundational Python skills. | Increase in marketability and profitability in the job market. |
| 2 | Develop a top-down arcade style space shooter game. | Finished piece of software for the developer’s portfolio. | Increase in marketability and profitability in the job market. |
| 3 | Deliver fast-paced, challenging gameplay. | Cultivation of a dedicated player base. | Increase in number of downloads on game marketplaces (e.g., itch.io) |

## Scope

In this section the features and functions that characterize the product, service, or result to be delivered by the project are described. That is, the major activities that must be completed to complete the project are listed. Activities that are out of scope for the project also are listed to reduce ambiguity.

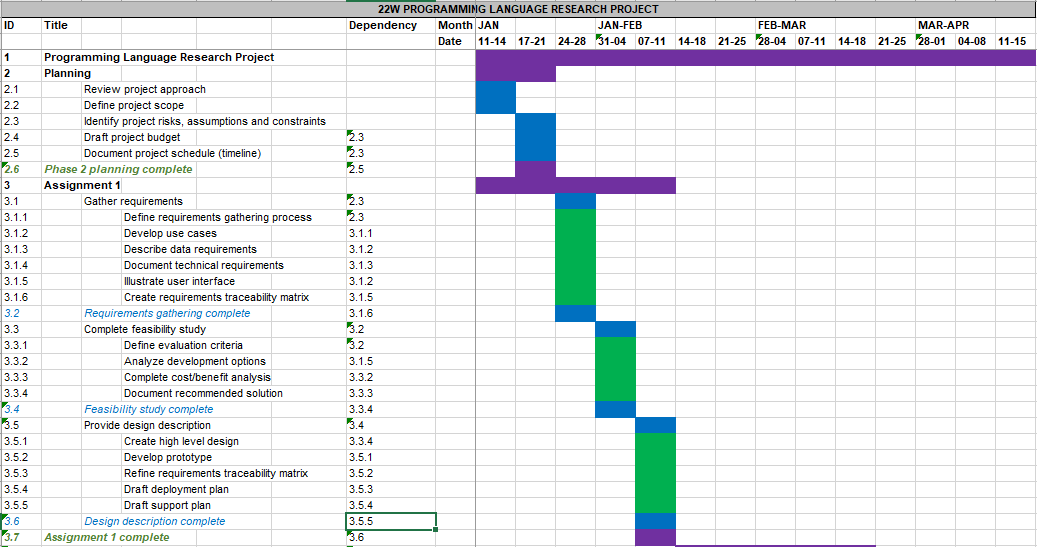
Table 2: Project Scope

| **ACTIVITIES IN SCOPE** | **ACTIVITIES OUT OF SCOPE** |
| --- | --- |
| 1. Requirement Collection and Analysis | 1. Apply for business financing |
| 2. Feasibility Study | 2. Direct requirement elicitation techniques, e.g., interviews, brainstorming, workshops |
| 3. Project Planning |  |
| 4. Testing |  |
| 5. Deployment |  |

## Timeline

In this section the project timeline is illustrated. The project duration is based on the CST8333 course calendar. Tasks included in the timeline are based on checklists included in CST8333 course modules. It is understood that unforeseen events and changes may result in revisions to the project timeline. Complete timeline is linked below Table 3.

Table 3: Project Timeline





### Milestones and Deliverables

In this section significant events in the project and their associated deliverables are defined.

Table 4: Project Milestones and Deliverables

| **MILESTONE** | **DATE** | **DELIVERABLES** |
| --- | --- | --- |
| 1. Assignment 1 complete | February 10, 2022 | Written report and slide presentation |
| 2. Assignment 2 complete | March 17, 2022 | Written report and slide presentation, 60% of source code complete |
| 3. Assignment 3 complete | April 14, 2022 | Written report and slide presentation,  100% of source code complete |

## Risks

Project risks are uncertain events or conditions that, if they occur, have positive effects (opportunities) or negative effects (threats) on one or more project objectives, such as scope, schedule, cost, and/or quality.

In this section the principal project risk is identified (e.g., schedule slippage, requirements inflation, conflicting requirements, deliverables quality) the likelihood it will occur is estimated (high, medium, low), its impact if it occurs is estimated (high, medium, low), and mitigation strategies are described (how likelihood and impact will be minimized).

Table 5: Project Risks

| **No.** | **RISK DESCRIPTION** | **PROBABILITY (H/M/L)** | **IMPACT (H/M/L)** | **MITIGATION** |
| --- | --- | --- | --- | --- |
| 1. | Schedule slippage | M | H | Track project scope and timeline |

## Assumptions

Assumptions are factors that you believe to be true, although they are not confirmed to be true. Assumptions add risk to a project since it is possible that they will turn out to be false. Assumptions can impact any part of your project life cycle and resulting solution implementation, so it is important that they be documented.

In this section the principal project assumption is identified.

Table 6: Project Assumptions

| **No.** | **THE FOLLOWING IS ASSUMNED** |
| --- | --- |
| 1. | Fundamentals of new programming language will be learned and put to use, timely, to complete project |

## Technical Constraints

Constraints are fixed boundary conditions or limits on what you can do. They are the things you cannot change but that you need to be aware of and manage to. Technical constraints focus on architecture decisions that may limit your solution design. They tend to be inflexible and unchanging and may impact your solution implementation. They include areas such as development languages, hardware, other infrastructure, and software that must be used for your project.

In this section the principle technical constraint is identified.

Table 7: Technical Constraints

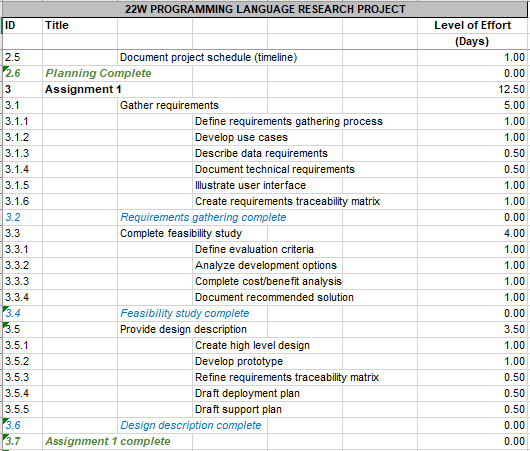
| **No.** | **TECHNICAL CONSTRAINTS** |
| --- | --- |
| 1. | Programming language selected does not accommodate all of the functionality desired in the solution |

## Budget

The project budget is a tool that is used to estimate all the costs that are likely to be incurred before the project is completed.

In this section a preliminary budget is estimated. Only in-scope items, as identified in section 1.3 above, are included. Out of scope items are excluded. It is understood that unforeseen events and changes may result in revisions to the project budget. Complete budget is linked below Table 8.

Table 8: Project Budget





# Requirements

## Introduction

Requirements define the characteristics or features of a desired solution. They must be actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for solution design. Categories of requirements include the following.

**Business Requirements:** The behaviours of the solution, including functionality, performance, and audit/reporting requirements.

**Technology Requirements:** The architectural elements of the solution, including network connectivity, system communications, user interface, hardware specifications, and operational/support needs.

**Compliance Requirements:** The legislation, regulations, and/or policies to which a solution is subject and with which it must comply.

**Security Requirements:** The security features that will be implemented to mitigate attacks. Since the end product will be fully offline, security risks are minimal and will be considered out of scope.

Each category of requirement is discussed in the sections that follow. Requirements in each category are defined using the following parameters.

1. Number
2. Name
3. Description
4. Subcategory
5. Driver
   1. Mandatory
   2. Desired
   3. Optional

As pertains to records retention requirements, records shall be kept in storage according to legislative, regulatory, and policy requirements, whichever is most strict.

Given short development cycle of this project, requirement solicitation from end users is considered out of scope. Instead, the following requirements were based on generally accepted conventions from the arcade scrolling shooter genre (Codex Gamicus, 2011), as well as the developer’s own experience with similar games. Examples of such games include Galaga (The Free Video Game Project, n.d.) and Platypus (Flack, 2002).

## Business Requirements

Business requirements mirror the business drivers of the solution. Subcategories of business requirements include functional, performance, and reporting requirements.

Table 9: Business Requirements

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
| --- | --- | --- | --- | --- |
| B-01 | Fully Offline | Program will not require network connectivity | Functional | Mandatory |
| B-02 | Consistent Framerate | Framerate will be kept to a steady 60fps | Performance | Desired |
| B-03 | New game | When the new game option is selected from the Menu scene, the program will transition to the Game scene | Functional | Mandatory |
| B-04 | Element stack | Menu elements are added to a first-in-last-out stack when they are opened. The element at the top of the stack is considered active | Functional | Mandatory |
| B-05 | Exclusivity of control | When a menu element is active, controls for other UI elements are disabled | Functional | Mandatory |
| B-06 | Closing an element | When a menu element is closed, it is removed from the element stack. | Functional | Mandatory |
| B-07 | Player hitbox | The player object’s hitbox will be a point at the centre of the player sprite | Functional | Mandatory |
| B-08 | Enemy hitbox | Enemy hitboxes will cover the entire area of their sprites | Functional | Mandatory |
| B-09 | Bullet hitbox | Bullet hitboxes will cover the entire area of their sprites | Functional | Mandatory |
| B-10 | Player-Enemy Collision | If an enemy hitbox overlaps the player hitbox, the player is destroyed | Functional | Mandatory |
| B-11 | Enemy-Bullet Collision | If a bullet hitbox overlaps an enemy hitbox, decrement the enemy’s hit points by the bullet’s damage value | Functional | Mandatory |
| B-12 | Enemy destruction | If an enemy’s hit points are reduced to 0, destroy the enemy | Functional | Mandatory |
| B-13 | Losing Lives | When the player is destroyed, decrement their remaining lives by one | Functional | Mandatory |
| B-14 | Player Respawn | After losing a life, if there are any lives remaining, create a new player object | Functional | Mandatory |
| B-15 | Game Over | After losing a life, if there are no lives remaining, go to the High Score scene | Functional | Mandatory |
| B-16 | New High Score | If the player’s current score is higher than an entry on the high score table, insert the new score into the table at that point. Allow user to enter their initials for that entry | Functional | Mandatory |
| B-17 | Enemy behaviour | New enemies are periodically created at the top of the screen and advance toward the bottom of the screen, being destroyed when they leave the screen again | Functional | Mandatory |

## Technology Requirements

Technology requirements are related to the information systems aspects of solutions. Subcategories of technology requirements include application, hardware, data, interface, availability, and maintenance requirements.

Table 10: Technology Requirements

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
| --- | --- | --- | --- | --- |
| T-01 | User interface | Function across multiple platforms | Application | Desired |
| T-02 | Text file storage | High scores will be stored and read from a plain text file | Data | Mandatory |
| T-03 | Program scenes | Program will be split into self-contained “scenes” | Interface | Mandatory |
| T-04 | Menu scene | The menu scene will provide an interface to access the other program scenes and to exit the program | Interface | Mandatory |
| T-04 | Menu opens on launch | The menu scene will be the first displayed when the program launches | Interface | Mandatory |
| T-05 | Game scene | Contains the main gameplay loop and associated UI | Interface | Mandatory |
| T-06 | Current score | UI element in Game scene will display current score | Interface | Mandatory |
| T-07 | Remaining Lives | UI element in Game scene will display number of remaining lives. | Interface | Mandatory |
| T-08 | Info text | UI element in Game scene will display game related information as appropriate | Interface | Mandatory |
| T-09 | Keyboard controls | Menu navigation and player actions will be controlled by keyboard inputs | Interface | Mandatory |
| T-10 | Mouse controls | Menu navigation and player actions will be controlled by mouse inputs | Interface | Desired |
| T-11 | Pause menu | UI element in Game scene will allow user to access options menu or end the game early | Interface | Mandatory |
| T-12 | High Score scene | Displays list of ten highest scores and corresponding user initials | Interface | Mandatory |
| T-13 | Return to Menu button | UI element in High Score scene that allows user to navigate to the Menu scene | Interface | Mandatory |
| T-14 | Options menu | Pop-up UI element that allows users to adjust gameplay and UI settings | Interface | Desired |

## Compliance Requirements

Compliance requirements articulate the legislative, regulatory, and policy requirements to which a solution is subject and with which it must comply. Individual laws and/or policies may be considered subcategories of compliance requirements

Table 11: Compliance Requirements

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
| --- | --- | --- | --- | --- |
| C-01 | Original Assets | Program will use original art, music and sound assets | Compliance | Mandatory |

## Requirements Traceability Matrix

This initial version of that requirements traceability matrix is the first step in the process identifying the tests that will be performed to validate whether documented requirements have been achieved.

Table 13: Requirements Traceability Matrix

| NUMBER | CATEGORY | REQ’T | TEST | EXPECTED RESULT | ACTUAL RESULT | PASS/FAIL | COMMENTS |
| --- | --- | --- | --- | --- | --- | --- | --- |
| B-03 | Functional | New Game | “New Game” main menu option is selected | Program transitions to Game scene and a new game begins | TBD | TBD | TBD |
| B-07 | Functional | Player hitbox | Mouse is moved over player hitbox | “Player collision” message is written to console | TBD | TBD | TBD |
| B-08 | Functional | Enemy hitbox | Mouse is moved over enemy hitbox | “Enemy collision” message is written to console | TBD | TBD | TBD |
| B-09 | Functional | Bullet hitbox | Mouse is moved over bullet hitbox | “Bullet collision” message is written to console | TBD | TBD | TBD |
| B-10 | Functional | Player-Enemy collision | Enemy hitbox collides with Player hitbox | Player is destroyed | TBD | TBD | TBD |
| B-11 | Functional | Enemy-Bullet collision | Bullet hitbox collides with Enemy hitbox | Enemy hit points are reduced | TBD | TBD | TBD |
| B-12 | Functional | Enemy destruction | Enemy hit points reduced to 0 | Enemy is destroyed | TBD | TBD | TBD |
| B-13 | Functional | Losing Lives | Player is hit and destroyed | Player lives reduced by one | TBD | TBD | TBD |
| B-14 | Functional | Player respawn | Player is destroyed with lives remaining | Player respawns | TBD | TBD | TBD |
| B-15 | Functional | Game over | Player is destroyed with no lives remaining | Program transitions to High Score Screen | TBD | TBD | TBD |
| B-16 | Functional | New High Score | Player gets new high score | Score is inserted at the right spot | TBD | TBD | TBD |

# Feasibility

## Introduction

In this section possible game designs are evaluated and a preferred solution is identified. Business, technical, and cost data are presented to help answer the following questions.

1. Given the developer’s current skills and experience is it reasonable to expect?
   1. The project will be completed within the time allowed.
   2. The new system will perform adequately.
2. Are there any environmental factors that need to be considered before undertaking the project (e.g., other private and/or professional commitments)?
3. Are there sufficient knowledge and technical resources available to complete the project and then maintain the system after it goes live?
4. Will the results of the feasibility study be used as inputs in the decision to design and implement the system?

I will work through the evaluation point by point. In reference to the project timeline presented in section 1.3, I have every reason to believe that the project can be completed on time. Likewise, given that a primary advantage of arcade-style space shooters is their simplicity, I am confident that the system will be simple enough to develop and will perform well.

Environmental factors in play are mainly related to my existing workload. Naturally, I have other coursework that requires my time and attention. I also have work commitments totalling approximately twenty hours per week and a young family to care for. However, these commitments have been factored into the creation of the project budget (section 1.7) and I am confident that the project will be completed on schedule.

The system will be completely offline once completed and will not require database technologies or other systems to support it. Bug fixes and quality of life updates may be released periodically but system maintenance will be fairly minimal overall. As for knowledge and technical resources, there is an active and healthy community surrounding the Python language and Pygame library, so filling the gaps in my current knowledge of these systems will not be difficult.

Finally, this feasibility study will serve to inform the design and implementation of the system. A fully-featured design will be preferred but alternatives will be proposed should the complete design prove to be infeasible moving forward.

## Alternative Solutions

In this section each alternative solution is explained in terms of its technical, operational, and economic feasibilities.

**Solution 1**

The first solution is a fully-featured arcade-style space shooter. This solution will include the game scene, a high score scene, and an options menu for adjusting user preferences.

Economic Feasibility

The hardware and software needed to develop and release this game are already in place and will need no further monetary investment. The costs incurred will mainly be opportunity costs related to development time. This fully featured version of the game will incur the greatest cost due to increased development time but will also produce the most value in terms of quality of work. Since the game will primarily be used as a portfolio piece once completed, higher quality of work will translate to better and more profitable employment opportunities in the future.

Structural Feasibility

The game, regardless of the features included, will be lightweight enough to be run on nearly any consumer PC. Likewise, the development environment will be similarly lightweight and not require additional specialized equipment.

Operational Feasibility

The workload involved in undertaking this project, as outlined in the project budget (section 1.7), is entirely manageable for a single developer and no additional staff will be necessary.

**Solution 2**

The first solution is a scaled-down arcade-style space shooter. This solution will include the game scene, a high score scene, but do away with the options menu.

Economic Feasibility

This pared down version of the game will take less time to develop, thus incurring lower costs, but produce a less impressive game overall. This could potentially detract from its value as a portfolio piece. It will, however, still serve as a practical demonstration of skill with Python and may make the difference between securing a lucrative position or not.

Structural and Operational Feasibility

Like the fully featured version of the game, no additional investment in equipment or staff will be necessary.

## Recommendation

In this section recommendations are prioritized according to the results of the analysis of the alternatives, beginning with the preferred alternative.

My first recommendation is Solution 1, the fully-featured version of the game. While both solutions have similarly positive structural and operational feasibility, the fully-featured game stands to be more profitable in the long run for economic reasons.

My second recommendation is Solution 2, the scaled-down version of the game. While there is more to gain from pursuing Solution 1, there is a risk of development costs overrunning the budget. Solution 2 is more likely to turn a profit, even if the margin is lower.

Conclusion

The purpose of this feasibility research report was to address the issue of which design to pursue. This report offered two alternative solutions to the problem: The fully-featured version and the scaled-down version. In addition, the report ranked each alternative according to its strengths and its benefits. The fully-featured version was found to be the preferred alternative. The scaled-down version was the second ranked recommendation.

# Design

## Introduction

The purpose of this section is to outline the project architecture, installation, and configuration procedures with the following goals.

* Provide a general description of the system
* Describe the logical architecture of the application/software
* Describe the physical architecture of the environment in which the software runs
* Provide proof of alignment between the architecture and systems requirements

## System Architecture

In this section descriptions of the system are provided using narrative and visually descriptive documents where appropriate.

### System Requirements

The game will be designed to run on consumer PCs and to not require network access. While the true minimum system requirements can only be determined in the testing phase, I have listed below the specifications of the machine that will be used for development and testing. The program should, at minimum, be able to perform optimally on this device.

* OS: Windows 10 Home
* Processor: 4.00 GHz
* Memory: 16 GB RAM
* Graphics: 8 GB

### Logical Architecture

The following architectural diagrams were produced using the requirements listed in sections 1.9, 1.10, and 1.11, as well as the primary solution explored in the feasibility study, as inputs.

The general architecture describes a series of nested elements with Scene objects forming the base and various Game Objects making up the component parts. Scenes represent distinct parts of the program, such as the main menu, high score table, and gameplay area. Game Objects represent the various component parts of each scene, such as buttons, images, text and windows. Listener classes were also defined to handle keyboard and mouse inputs.

Everything that happens in the program is tied back to the main loop. The loop is synchronized to a clock and repeats sixty times per second. When a new scene is instantiated, its start method is called, which populates the scene with game objects via the game object manager class. The current scene’s update method is called once per loop. When this happens, the scene calls its game object manager’s update method, which cycles through its list of game objects and calls each one’s onUpdate method. This allows the state of each game object to change over time, facilitating things like movement and animation. Calling the scene’s end method uninstantiates its game objects and then the scene itself, in preparation for a new scene to begin.

Figure 1: Static Class Diagram

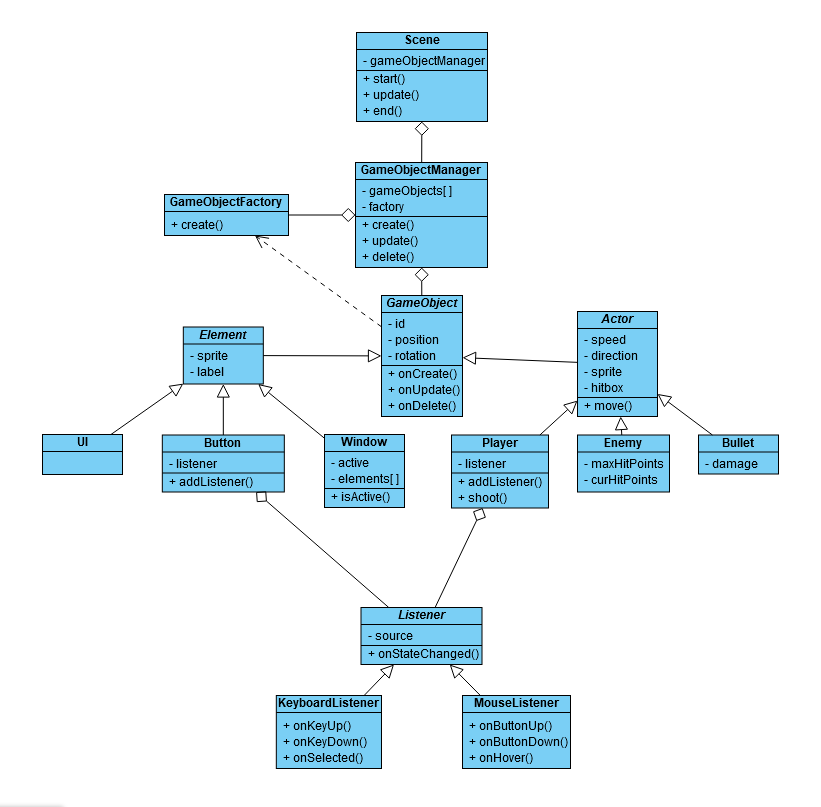
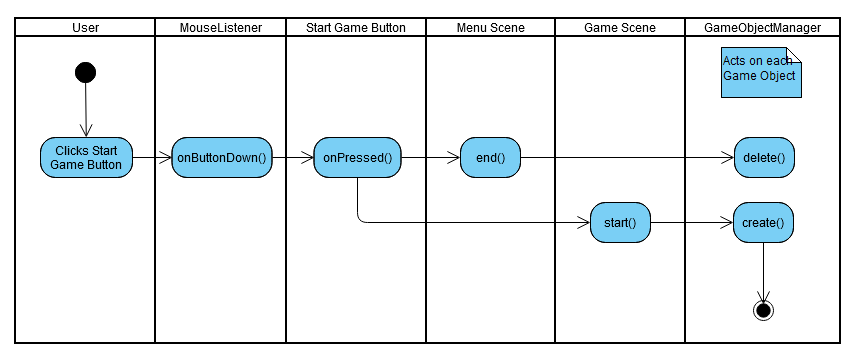


Figure 2: Start Game Activity Diagram



### UI Mockups

The following images were created to represent what various game scenes might look like in action. Figure 3 represents the menu scene, which will be presented to the player when the program launches. It displays a UI Element Game Object with the game’s title, as well as a series of Button Elements allowing navigation to other scenes.

**Figure 3: Menu Scene Mockup**



Figure 4 represents the game scene, where the main gameplay loop takes place. It contains UI Elements displaying the current score and remaining lives, as well as the Player Actor firing Bullet Actors, and a number of Enemy Actors moving in pre-set patterns from the top of the screen to the bottom.

**Figure 4: Game Scene Mockup**

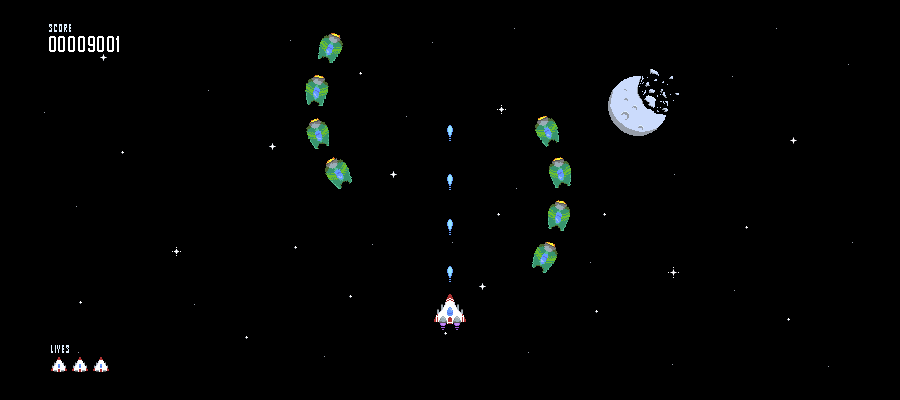
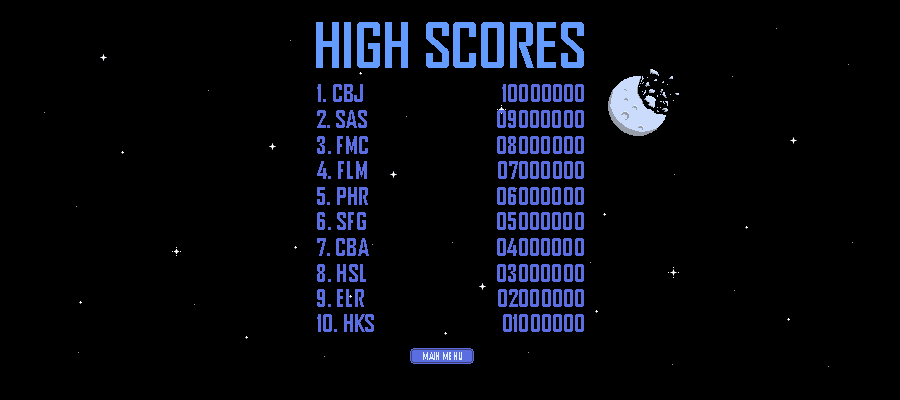


Figure 5 represents the high score scene, which displays the high score table when selected from the main menu or after a game over. The scores are stored in a plaintext file and are read into a string array to be displayed on the screen. If the player gets a new high score, it is inserted into the array at the correct position and the top ten scores are written back to the plaintext file. Players can also enter their initials to be displayed alongside their score.

**Figure 5: High Score Scene Mockup**



# References

*Shoot 'em up video games*. (2011). Codex Gamicus. https://gamicus.fandom.com/wiki/Shoot\_%27em\_up\_video\_games

The Free Video Game Project (n.d.). *Free Galaga*. FreeGalaga.com. Retrieved February 10, 2022, from https://freegalaga.com/

‌Flack, A (2002). *Platypus*. (n.d.). Store.steampowered.com. Retrieved February 10, 2022, from https://store.steampowered.com/app/307340/Platypus/

‌