CST8333

Assignment 3

Space Guy: Blockade

Andrew Gaal

14 April, 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](#_Toc100841559)

[1.1 Objectives 2](#_Toc100841560)

[1.2 Scope 2](#_Toc100841561)

[1.3 Timeline 3](#_Toc100841562)

[1.3.1 Milestones and Deliverables 4](#_Toc100841563)

[1.4 Risks 4](#_Toc100841564)

[1.5 Assumptions 4](#_Toc100841565)

[1.6 Technical Constraints 5](#_Toc100841566)

[1.7 Budget 5](#_Toc100841567)

[2 Status 7](#_Toc100841568)

[2.1 Introduction 7](#_Toc100841569)

[2.2 Accomplishments 7](#_Toc100841570)

[2.3 Goals 7](#_Toc100841571)

[2.4 Roadblocks 7](#_Toc100841572)

[2.5 Lessons Learned 8](#_Toc100841573)

[2.6 Next Steps 8](#_Toc100841574)

[3 Source Code Implementation 9](#_Toc100841575)

[3.1 Introduction 9](#_Toc100841576)

[3.2 Coding Checklist 9](#_Toc100841577)

[3.3 Submit Code 10](#_Toc100841578)

[4 Testing 11](#_Toc100841579)

[4.1 Introduction 11](#_Toc100841580)

[4.2 Methods 11](#_Toc100841581)

[4.3 Requirements Traceability Matrix 11](#_Toc100841582)

[5 Deployment 13](#_Toc100841583)

[5.1 Introduction 13](#_Toc100841584)

[5.2 Software Integration Plan 13](#_Toc100841585)

[6 References 14](#_Toc100841586)

# 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.

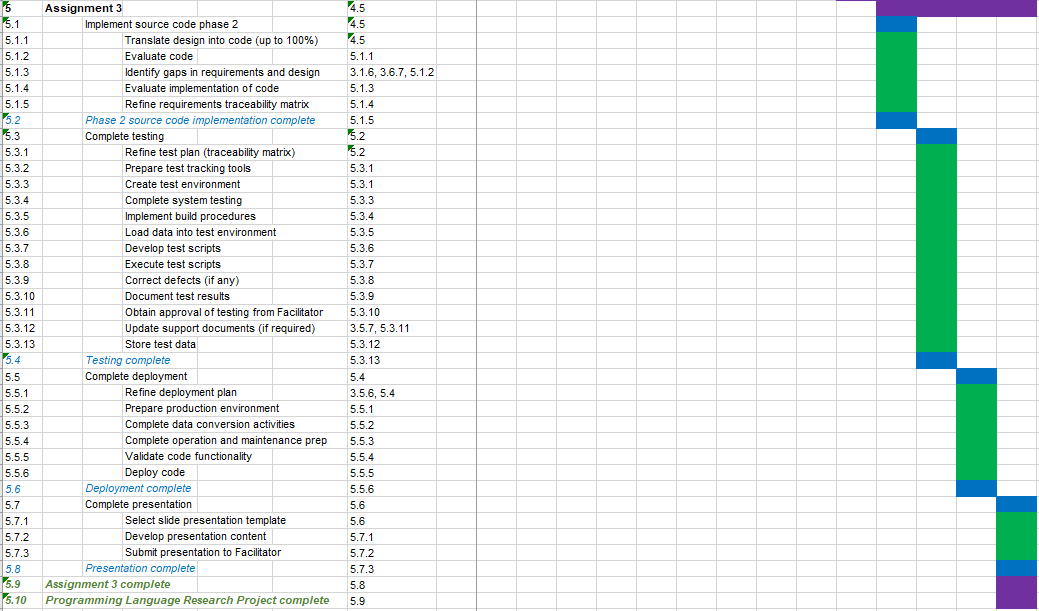
Table 2: Project Scope

| **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.

Table 3: Project Timeline



Legend

* Purple bars = Level 1 tasks
* Blue bars = Level 2 tasks
* Green bars = Level 3 tasks



### 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 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 principle 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.

Table 8: Project Budget





# Status

## Introduction

This section details the progress made on the project since the last report. Included are the high-level goals that were set for this period of work, specific accomplishments, roadblocks encountered, and lessons learned during development.

## Accomplishments

The project is now in a completed state. Features implemented since the last report include, but are not limited to:

* Implemented High Score scene with functioning text file I/O for saving and loading score data between sessions;
* Created a pop-up window interface for creating menus and other UI elements;
* Implemented a text entry UI, extending the pop-up interface, for entering user initials when a new high score is achieved;
* Implemented a pause menu UI, extending the pop-up interface, which pauses the game scene when active and allows users to navigate to the menu scene or to quit the game;
* Added animation states to button UI sprites. Buttons now look different depending on whether the mouse is hovering over them or not;
* Updated background image to include scrolling animation;
* Added scrolling starfield parallax effect to game scene;
* Implemented hit points for enemies;
* Created and implemented an animated explosion effect, displayed when the player or an enemy is destroyed;
* Created and implemented a simple music loop which plays while the game scene is active.

An updated list of requirements and their current state of completion can be found in Section 4.3: Requirements Traceability Matrix.

## Goals

* Complete system architecture according to revised design
* Implement remaining subsystems and features outlined in previous reports
* Complete testing phase
* Build deployable application from source code and other assets

## Roadblocks

Development during this phase of the project proceeded very smoothly and no major roadblocks were encountered. All of the major architectural issues encountered during the last phase of development were addressed during that period. Under this new architectural framework, and having gained some experience with the development tools during the last cycle, the remaining features were relatively simple to implement.

## Lessons Learned

The major theme of this development cycle was of taking the lessons learned from the last cycle and learning to apply them. One of the most important lessons could best be described as “measure twice, cut once.” That is, learn as much as possible about the development tools to be used before delving too deeply into system architecture and committing a design that may not be possible to implement.

A major source of schedule slippage during the last development cycle was a fundamental lack of understanding of the capabilities and limitations of Python and PyGame. Time that could have been spent exploring these capabilities during the project initiation phase was instead pushed into the development phase. As a result, not only was that time spent instead on orientation, but it was also spent reworking the design itself.

Having addressed the issues of orientation and design in the last development cycle, this cycle proceeded completely according to plan. This cemented the lessons learned previously and demonstrated their value.

## Next Steps

This project is now in a completed state. That is, the development goals and requirements set during the project initiation phase have been met. The result is a deployable alpha build that can be promoted on an online portfolio or distribution platform. There is, however, a great deal of improvement to be made in terms of game mechanics, quality of life features, and general level of polish. “Nice To Have” features such as fully-animated player and enemy sprites, transition animations between scenes, individual levels, and more complex enemy behaviours could certainly be added post-deployment.

The primary goal of this project, however, was to learn about and practice using a new programming language over a short fourteen-week period, and that goal has been accomplished. If I decide to continue developing this game, I will likely take the lessons learned during this project and apply them to rebuilding the game from the ground up using a purpose-built game engine such as Unity or Godot.

# Source Code Implementation

## Introduction

During the implementation phase of the software development life cycle (SDLC) the development team creates the agreed upon solution to the business problem identified at the outset of the project. Project developers build the software in the staging or development environment using the programming language that was chosen in the previous phase of the project. Also known as building or coding, this stage is where the planning and prototyping come together.

## Coding Checklist

In this section the completion of outstanding items included in the *Coding Checklist* submitted in support of *Assignment 2* is documented. Source code implementation was completed using the workflow illustrated, below, in Figure 1.

Figure 1: Implementation Workflow

Diagram

Description automatically generated

Progress against *Coding Checklist* items is indicated in the table, below.

Table 9: Coding Checklist

| **Description** | **Completed** | **If “No” Please Explain** |
| --- | --- | --- |
| Complete: Everything that is in the requirements and design is implemented | Yes |  |
| Consistent: No mismatched interfaces and consistent with the design | Yes |  |
| Stylistic: Exhibits good programming style. | Yes |  |

## Submit Code

The source code is accessible below, and has also been included with this project’s submission as a zip archive. Additionally, a playable demo has been included which can be played by extracting the Space Guy.zip archive to a convenient directory and launching Space Guy.exe.



# Testing

## Introduction

The purpose of software testing is to identify errors, gaps, or missing requirements in contrast to actual requirements. Testing may be resource-intensive, but in addition to ensuring requirements have been met it supports cost-effectiveness; it costs less to fix defects identified during testing than those identified post-implementation. Fixing defects enhances product quality which, in turn, drives customer satisfaction.

## Methods

Testing was performed on a feature-by-feature basis parallel to development. Because Python does not need to be compiled, and because the program is a relatively simple stand-alone offline application, this combination of rapid iteration/integration testing is both suitable and effective. The testing procedure was as follows:

* Write source code for a feature
* Run program
* Check result against expectation (referencing the Requirements Traceability Matrix below)
* If it passes, move on to the next feature
* If it fails, or breaks some other feature, debug source code and test again

## Requirements Traceability Matrix

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

Table 10: 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 | Program transitioned to Game scene and a new game began | PASS |  |
| B-07 | Functional | Player hitbox | Mouse is moved over player hitbox | “Player collision” message is written to console | “Player Collision” message written to console | PASS | Debug message since removed from source code |
| B-08 | Functional | Enemy hitbox | Mouse is moved over enemy hitbox | “Enemy collision” message is written to console | “Enemy Collision” message written to console | PASS | Debug message since removed from source code |
| B-09 | Functional | Bullet hitbox | Mouse is moved over bullet hitbox | “Bullet collision” message is written to console | “Bullet Collision” message written to console | PASS | Debug message since removed from source code |
| B-10 | Functional | Player-Enemy collision | Enemy hitbox collides with Player hitbox | Player is destroyed | Player destroyed | PASS |  |
| B-11 | Functional | Enemy-Bullet collision | Bullet hitbox collides with Enemy hitbox | Enemy is destroyed | Enemy Destroyed | PASS |  |
| B-12 | Functional | Enemy destruction | Enemy hit points reduced to 0 | Enemy is destroyed | As expected | PASS |  |
| B-13 | Functional | Losing Lives | Player is hit and destroyed | Player lives reduced by one | Player lives reduced by one | PASS |  |
| B-14 | Functional | Player respawn | Player is destroyed with lives remaining | Player respawns | Player respawns | PASS |  |
| B-15 | Functional | Game over | Player is destroyed with no lives remaining | Program transitions to High Score Screen | As expected | PASS |  |
| B-16 | Functional | New High Score | Player gets new high score | Score is inserted at the right spot | As expected | PASS |  |

# Deployment

## Introduction

The deployment phase is the final phase of the software development life cycle (SDLC) and results in the migration of the product to the production environment. The objective of the deployment phase is to make the developed software operational in a live environment. Deployment in the operational environment comes only after the product is fully tested and accepted by the business.

## Software Integration Plan

In this section the completion of items included in the integration plan is documented.

Table 11: Software Integration Plan

| NUMBER | CATEGORY | REQUIREMENT | COMPLETE | IF “No” PLEASE EXPLAIN | COMMENTS |
| --- | --- | --- | --- | --- | --- |
|  | Build application | Define development strategy | Yes |  |  |
|  | Build application | Evaluate resources | Yes |  |  |
|  | Code review | Ensure code complies with specifications | Yes |  |  |
|  | Code review | Validate code working as planned | Yes |  |  |
|  | Code review | Validate code is defect free | Yes |  |  |
|  | Deliverable | Submit final code and working application to facilitator | Yes |  |  |

Additional Source:

*Software Deployment.* (2016, March 19). Process Street. <https://www.process.st/checklist/software-deployment/>

# References

(2020). Ohio.Gov. <https://regents.ohio.gov/obrpmcop/forms/templates/temp_systemapplicationsupport.doc>

*How to Write an Effective Test Summary Report (Sample Report Download).* (2020, September 1). Retrieved September 27, 2020, from <https://www.softwaretestinghelp.com/test-summary-report-template-download-sample/>

Parashchenko, R. (2009). *Software testing test plan templates.* Strongqa.Com. <https://strongqa.com/qa-portal/testing-docs-templates/test-plan>

*Sample Test Plan -OrangeHRM Live Project Training Test Plan (a Real Sample).* (n.d.). <https://www.softwaretestinghelp.com/wp-content/qa/uploads/2014/02/Live_Project_Test_Plan_SoftwareTestingHelp.pdf>

*Software Deployment.* (2016, March 19). Process Street. <https://www.process.st/checklist/software-deployment/>

*Software Testing - Checklists | QA and Software Testing Tutorial.* (n.d.). Www.Qatutorial.Com. Retrieved September 27, 2020, from <http://www.qatutorial.com/Software-Testing-Checklists#:~:text=This%20MS-Word%20based%20Checklist%20on%20Software%20Testing%20contains,and%20data%20conversion%2C%20Training%20and%20many%20other%20sections>.

*The QA Software Testing Checklists (Sample Checklists Included).* (2013, November 27). Softwaretestinghelp.Com. <https://www.softwaretestinghelp.com/software-testing-qa-checklists/>