Score: 94/100
Add List of Tables and label all the tables. Document needs additional formatting

Deep Space

Prepared by
Ethan Luong, Eldin Vujic, Victor Fong
***for use in* CS 440**
at the
University of Illinois Chicago

November 2020

Table of Contents

I Project Description

1 Project Overview

Deep Space is an adventure-puzzle game that allows players to explore a spaceship while simultaneously combating enemies while trying to figure out a way to get off of the spaceship. The player overcomes puzzle-like obstacles, enemies they may encounter, gathering resources, and get past obstacles to fully unlock the escape pod room which then the user can use to escape with.

2 Project Domain

The domain of the project, Deep Space, is to provide users with the appropriate application software that would allow them to access the application which then they can play the game with the given levels, enemies, and items that the game will provide.

3 Relationship to Other Documents

Documents that are related to the project include,

Group10_ProjectReport.pdf by Nicole Mihaylova, Eson Musabelliu, Alejandro Serrano, Timothy Werkheiser of Spring 2018.

Group28CombatSummary.pdf, AdvGameEnvironmentScenario.pdf, Group28_DeepSpace_CodePresentation.pdf, theSkeld.txt, testFile.txt, all by Ethan Luong, Eldin Vujic, Victor Fong of Fall 2020.

4 Naming Conventions and Definitions

4a Definitions of Key Terms

Map file: A file the user will load in holding information about the game including rooms, enemies, items, and control panels.

Item: An object that is loaded in from the map file, there can be multiple items in the map file. And from these items there are numerous different types of items. The first type of item is equippable and this is an item that can be picked up and equipped by the user. The subtypes of equippable items include helmet, leggings, chest piece, and weapon. These are four of the equippable items and all these items are very similar because they all have one purpose which is that they increase the users stats in the game. The only difference is that that belongs to different body parts of the users body, where leggings are used for the legs, chest pieces are used for the chest, helmet is used for protecting the head, and lastly weapon is used for the user holding the weapon. Other types of items include healing and power-up items which these types of

items are not equippable but instead are consumed and when they are consumed they provide either health or strength benefits to the users stats, but they are not temporary. The last type of item that we have are items that unlock, which these items are neither stat changers to the user, but instead help unlock rooms.

Room: Rooms are loaded from the map file, they are challenges that the user faces and they are either unlocked or locked, unlocked rooms are rooms that the user is able to access without any restrictions. Unlocked rooms can be either unlocked with items that unlock them or with control panels that are laid throughout the rooms.

Enemies: Enemies loaded from the map file, they're challenges that the user faces and they are either aggressive or non-aggressive. Aggressive enemies drag the user into a battle which then the user can either flee or fight back. While non-aggressive enemies are neutral which the user can choose to ignore the enemy or fight them. Enemies all have health points and attack damage that all vary, which they can also drop any of items as specified.

HP: Health Points of the player. When this stat reaches 0 the player will receive a game over.

AD: Attack Damage. Determines how much damage the player will do in combat.

4c Data Dictionary for Any Included Models

The textfile known as theSkeld.txt is a date file format that is relative to this project due to this textfile holding all of the rooms that are all connected to each other due to the path system that takes place in our game. Where the path class helps detect which rooms are in which direction of the game. It also holds all of the enemies and item locations and where they would be placed when the game is loaded up.

II Project Deliverables

The group 28 of fall 2020, including Ethan Luong, Eldin Vujic, Victor Fong,

and Saude Chaudry have produced the first release which was titled as the

AdvGameEnvironment Scenario, and the second release which was titled as

the CombatSummary Scenario.

5 First Release

The first release was released on the date 10/09/20 and the functionality of the system as of that point was that the project was working in terms of the command line. At this point of the release the project was heavily reliant on the use of the command

line in terms of playing the game. So the game was only working in the command line and the GUI still was not set up at this point. The other functionality this release had was how the player was only able to move through the levels, pick up items and unlock doors. So overall this release was for the users getting familiar with the environment in the game and how the user can traverse through rooms.

6 Second Release

The second release was released on the date 11/06/20 and the functionality of the system as of that point was that the system was almost complete. The game at this point went from being command line based to GUI based which made it easier for the users to play the game. And it was more visually appealing compared to typing every command to play the game. Also another functionality this system had at this release was the use of combat, heals, power-ups, enemies, and equippable items. So at this point the system required minimal testing to ensure that there were no bugs, but overall the system was basically close to being complete where bug fixing was the only thing that was meant to be done.

7 Comparison with Original Project Design Document

The prototype produced compares with the full project described by Group 10 of Spring 2018, very similarly. The project overview stated, “In order to succeed in the game, the character will need to proceed to a certain location in the spacecraft; however, the character will face challenges throughout the spacecraft that they will need to overcome.” (Group10_ProjectReport). So overall, this prototype succeeds in doing these objectives, the prototype took place in a spacecraft where the user was met with challenges such as enemies, puzzles, and locked doors. Also where the user was supposed to proceed to a certain location in the spacecraft which in the prototype was the escape pod.

III Testing

8 Items to be Tested

Item Class, theSkeld.txt, Game Class, Player Class, GameController class, main fxml file, Path Class, Room Class.

9 Test Specifications

1# - Multiple-Items

Description: This test will display multiple items in one room location.

Items covered by this test: Item, theSkeld.txt, Game, Player, GameController class, main fxml file, and Room Class.

Requirements addressed by this test: The requirements addressed by this test would be

Environmental needs: This test requires the use of the text file with loaded items objects, the test file needed will be theSkeld.txt.

Intercase Dependencies: Test 3 - Creation, Maintenance, and Movement between Places.

Test Procedures: The steps needed to run this test would be the requirement of adding two items to the text file which is known as the date file. So to do this the tester will need to fill in the appropriate values for the table below (Table 1 - Multiple-Items), where id must be different and every other value can be zero besides the location. The input must require that the two items location must be the same. So for example, Table 2 - Item 1 and Table 3 - Item 2 will be two items with the same room location.

Table 1 - Multiple-Items

```
//  location
//  id  isaDrop isEquipped  hpValue attackDamageValue isLeg
isHelmet isArmor isWeapon  value  weight  keycode title
//  lines-in-description
//  description
```

Table 2 - Item 1

```
2
1 0 0 0 0 0 0 0 0 0 0 0 item1
2
Description of item 1
```

Table 3 - Item 2

```
2
2 0 0 0 0 0 0 0 0 0 0 0 item2
2
Description of item 2
```

. . .

Input Specification: The input of this test will be the tester entering the same room location as the item room locations they specified.

Output Specifications: The output of this test will be when the tester goes into the room that matches the room location, the tester will see that there are two items in that one room. For example, taking the information from the Table 2 -

Item 1 and Table 3 - Item 2, the tester will see that there will be the output of,

Name: item1	Name: item2
Description: Description of item 1	Description: Description of item 2

Pass/Fail Criteria: If the GUI displays the name and description of both items then the test will pass, but if it only displays one item's name and description the test fails.

2 - Main Game GUI

Description: This test will display the correct images and descriptions based on the given data file.

Items covered by this test: GameController class, main fxml file

Intercase Dependencies: Test 3

Test Procedures: The tester will run the program with a custom data file and images. When the main game GUI loads in, the room description, room name, images, and buttons should be recorded. The tester should move to the next room and record the same things.

Input Specification: A main data file should be the primary input for the text. This data file should include at least two rooms with unique names and descriptions. One room should have an enemy that can damage the player that will drop an item. The other room should have a panel to interact with. An test file is provided named “testFile.txt”

Output Specifications: The room description and name should match the ones provided in the data file. The background image, item image, and enemy image should match the ones provided. When entering the combat room, buttons allowing the option to fight should appear. When entering the panel room, a button to interact with the panel should appear.

Pass/Fail Criteria: If the GUI displays the correct images, room descriptions, and buttons it passes.

3 - Instantiating Relevant Objects Programmatically

Description: Cover the test case of the general functionality of creating a game.

Items covered by this test:

Room
Path
Game
Item
Enemy
StringParser
PreGame

Environmental needs: A working environment using openjdk as well as JavaFX are required, as well as makefile support in order to compile and run the code.

Intercase Dependencies: NA

Test Procedures:

Run the program, using the included .txt map ro run the game. As the txt file is input the Game class and other relevant classes will be instantiated if according to the txt file so long as it matches the given format.

Input Specification:

- *The data file containing necessary input. For testing purposes, text file "theSkeld.txt" is supplied with all necessary functionality.*
- *Input of .txt file into the PreGame GUI*

Output Specification

Output expected is the creation of objects manifested from the txt file by code. These objects can be manipulated via GUI buttons.

Pass/Fail Criteria:

Test is considered “passed” if all items have been successfully instantiated without error, and ready to be manipulated via GUI input.

10 Test Results

1# - Multiple Items

Date(s) of Execution: 11/20/20.

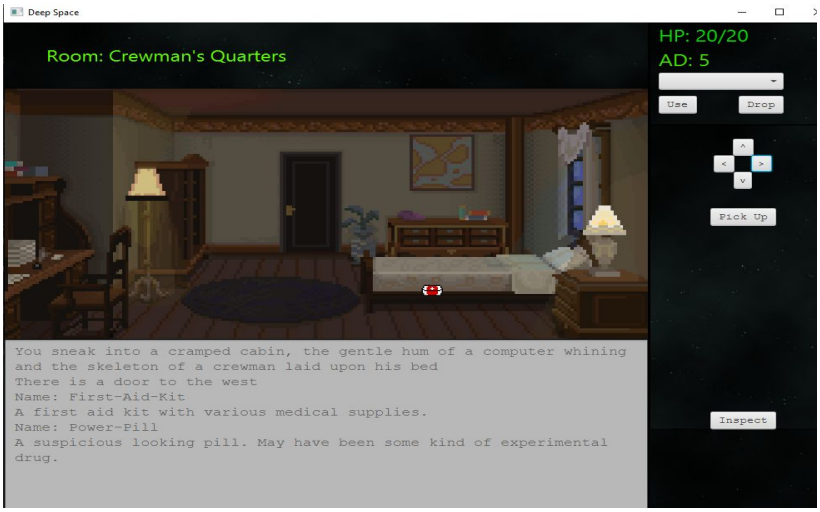
Staff conducting tests: Eldin Vujic.

Expected Results:

Name: First-Aid-Kit A first aid kit with various medical supplies. Name: Power-Pill A suspicious looking pill. May have been some kind of experimental drug.

Actual Results:

GUI:



Command Line:
Name: First-Aid-Kit
A first aid kit with various medical supplies.
Name: Power-Pill
A suspicious looking pill. May have been some kind of experimental drug.

Test Status: The test passed because it shows that in the room Crewman's Quarters there are the two items present which are power-pill and first-aid-kit.

2 - Main Game GUI

Staff conducting tests: Ethan Luong

Expected Results: Room names: Panel Room, Combat Room

Room Descriptions: Test description for Panel room,

Test description for Combat room

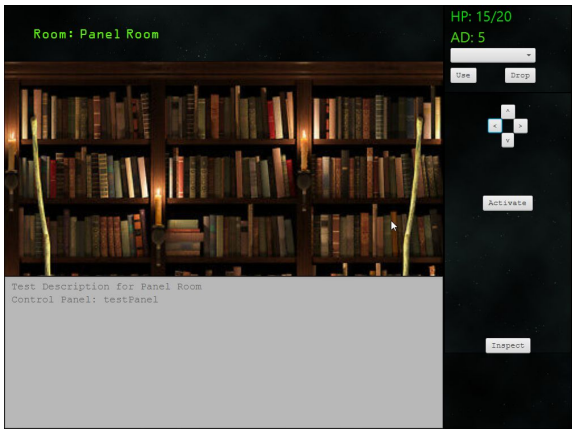
Buttons: Activate button in Panel Room, Fight in Combat Room

Images: Library image in panel room, guestCabin in combat Room

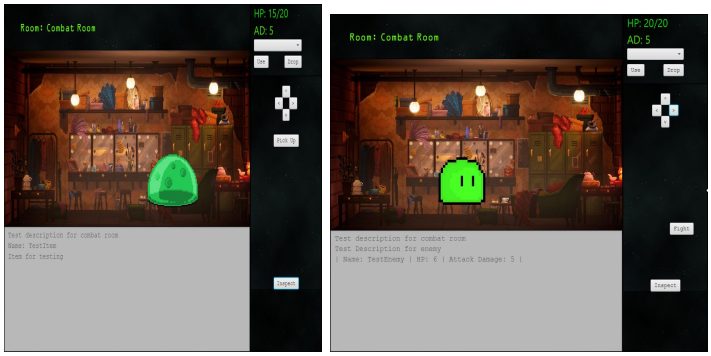
Slime enemy in combat room, slime drop after defeating slime.

Actual Results:

Panel Room



Combat Room



Test Status: Passed

3 - Instantiating Relevant Objects Programmatically

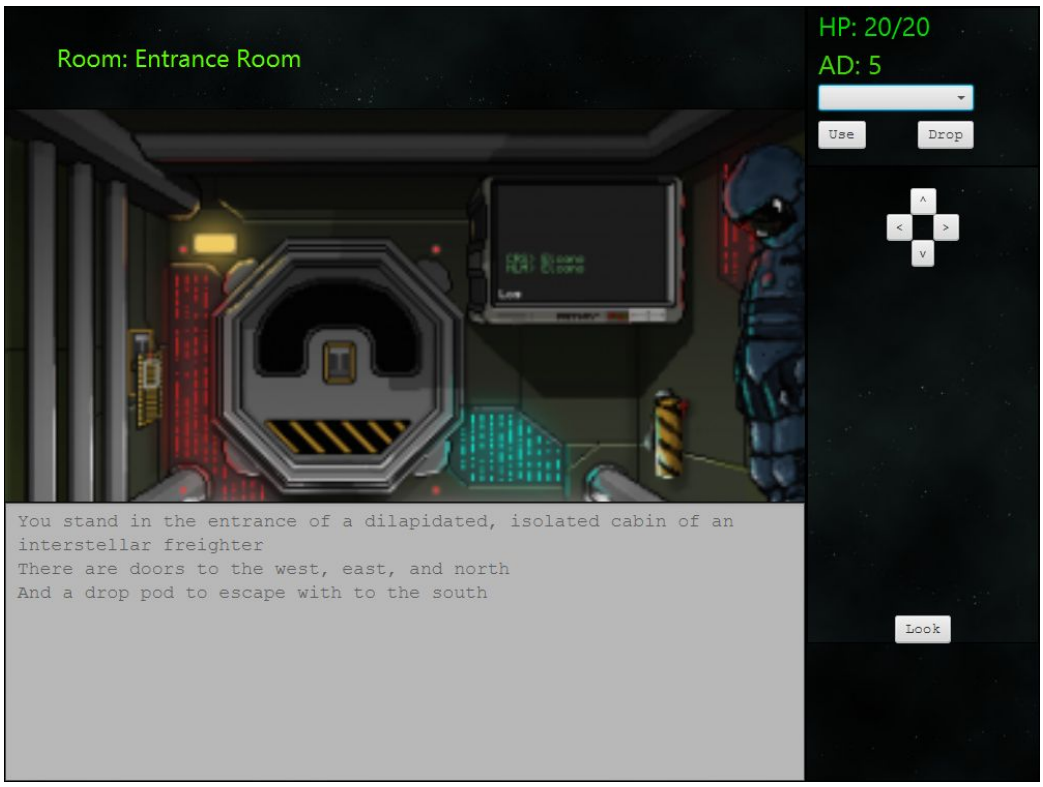
Date(s) of Execution: 11/23/20

Staff conducting tests: Victor Fong

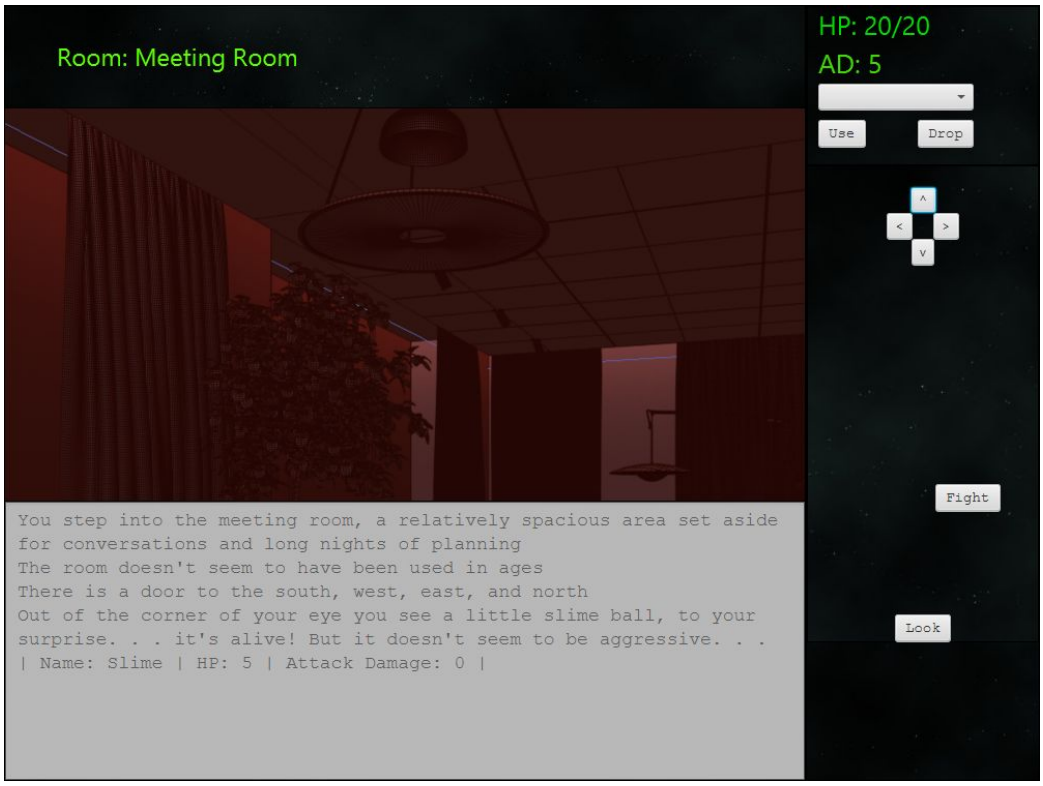
Expected Results:

Instantiation of Rooms, Items, Enemies, and Paths

Actual Results:



Places have been instantiated.



We also see Enemies working. At this point all objects relevant have been loaded in.

Test Status: Test passes, as all objects have been instantiated.

IV Inspection

12 Items to be Inspected

Main game GUI code - GameController class and main fxml

Enemy Class code

Constructors of Objects instantiated from txt file, as well as StringParser class

13 Inspection Procedures

The checklist that we used was from the following source [4] Users.Csc.Calpoly.edu. There were 3 meetings included as part of the process. The work that was done compared to outside vs inside meetings was that most of the work was done outside of meetings. But the results were mostly discussed online, via voice chat during those meetings.

14 Inspection Results

Who Inspected?	Eldin Vujic
What was inspected?	Main game GUI code - GameController class and main fxml
Date of Inspection?	11/25/20
What was discovered?	For this inspection I discovered when

	inspecting the Main game GUI code - GameController class and main fxml, was that most of the checklist was passed for the code. The only thing that I noticed that was wrong was the naming of variable names which falls under initialization and declarations. The variable names were spelled correctly but there was one variable that was never used.

Who Inspected?	Eldin Vujic
What was inspected?	Constructors of Objects instantiated from txt file, as well as StringParser class
Date of Inspection?	11/23/20
What was discovered?	For this inspection I discovered when inspecting the Constructors of Objects instantiated from txt file, as well as StringParser class. Was that most of the code passed the inspection checklist and the thing that stood out to me was the fact that the code did catch file exceptions. So when a user would try to enter a invalid textfile name the program would catch that instantly throwing an exception that the file is not found.

Who Inspected?	Ethan Luong
What was inspected?	Enemy Class
Date of Inspection?	11/25/20
What was discovered?	Inspecting the enemy class showed that all relevant items from the checklist were met. There was nothing wrong with the code and the specified functionality was fully implemented.

Who Inspected?	Ethan Luong
What was inspected?	Constructors of Objects, StringParser
Date of Inspection?	11/25/20
What was discovered?	The code passed all relevant items in the provided checklist. The code that created objects from a provided text file was consistent throughout all the classes. Full functionality was implemented and fulfilled all requirements for the program.

Who Inspected?	Victor Fong
What was inspected?	Enemy Class
Date of Inspection?	11/24/20
What was discovered?	Inspection of code shows efficient implementation and achieves all needed requirements.

Who Inspected?	Victor Fong
What was inspected?	Main game GUI code - GameController class and main fxml
Date of Inspection?	11/24/20
What was discovered?	Relevant code successfully implements a JavaFX GUI, as well as sets up listeners to access and influence relevant instantiated objects. All needed requirements met.

V Recommendations and Conclusions

VI Project Issues

15 Open Issues

Some issues that have been raised and do not yet have a conclusion is the location of where the user is in the spacecraft. This issue was raised when we began the development of the rooms, but we never found a proper way of being able to create a map for the user which marks down the players current location. With the final release and the class demo we had no conclusion for this issue. Another issue we had was the item class for our project because we had a vast amount of different types of items, so in order to address this issue we tried refactoring the item class. But the issue we had was being able to push these changes to the updated version of the code because we addressed this issue when we added items into the project, but when we tried refactoring we saw many issues arise from this, so ultimately we had no conclusion for the success of that operation.

16 Waiting Room

Better Combat: Combat is crucial for the user in this game and the current combat this system holds is a turn based combat where the player attacks first which then the enemy attacks second. This repeats until either the player or the enemy wins the battle, so the combat overall in the last release was all auto-combat. An idea that may be worth reconsidering at a later date would be the implementation of better combat. This combat includes the system having a more strategic means of player vs enemy, meaning the combat is more complex than the auto-combat. This includes the user

being able to use abilities and select multiple different abilities and attacks during the combat which then the enemy ai does the same.

Map: The rooms are crucial in order for the user to navigate throughout the rooms. An idea that may be worth reconsidering at a later date would be the implementation of a map. This would allow the user to keep track of where their location would be in the spacecraft and how far they are from other rooms from their current location.

Smarter Enemies: Enemies are crucial in the game, they are obstacles that the user faces when traversing through the rooms. An idea that may be worth reconsidering at a later date would be the implementation of enemies that move throughout different rooms. The current implementation that the system has is that an enemy remains in one room which then they can only encounter the user if the user enters the room. So an implementation where the enemies are allowed to move through other rooms can provide some enemies that aren't able to ever meet the user the opportunity to battle the user.

17 Ideas for Solutions

Some ideas we had in terms of coming up with the solution for some of the issues we had was for example for the first issue we had which was the lack of player map indicator, we had the idea of have another GUI window that would pop up which would show the player the location of the rooms and which room they are in. Another idea we had for fixing up our item class and making it more readable was by refactoring the class based on the different type of items for the item class. Which would make it easier to separate certain items to sub-item types.

18 Project Retrospective

There are many things that worked well in this project development, but there are also many things that didn't work that well. The things that worked well in this project was the implementation of items, enemies, and combat. The implementation of these things changed the project completely because now the user had another obstacle that they could encounter which was fighting enemies. So in a way this added to the difficulty to the game while also maintaining the genre of being an adventure game. Also the items worked well and that is due to the fact that the introduction of items allowed them to be so versatile. Meaning these items had multiple functions and their functionality helped the game flow smoothly.

There are also some things that didn't work well in this project development and some of those things included the combat system. Even though combat worked well in the game, there was a lack of strategic combat due to the game being all auto-combat. So overall we felt as if the combat didn't work as well as we wanted it to be due to the fact that it was all auto fighting and we wanted combat where the user can choose how to fight. Also the lack of showing the player the previous rooms they encountered and where they are on the location also didn't work as well. Because of this at times the user can feel lost when traversing through the rooms.

VIII References / Bibliography

[1] Nicole Mihaylova, Eson Musabelliu, Alejandro Serrano, Timothy Werkheiser, Deep Space Requirements Document, 2018

[2]Eldin Vujic, Ethan Luong, Victor Fong, Saude Chaudry, Deep Space Scenario 1 AdvGameEnvironmentScenario Document, 2020

[3]Eldin Vujic, Ethan Luong, Victor Fong, Saude Chaudry, Deep Space Scenario 2 CombatSummary Document, 2020

[4] Users.Csc.Calpoly.Edu, 2020,

http://users.csc.calpoly.edu/~jdalbey/301/Forms/CodeReviewChecklistJava.pdf.