Coffee Maker Quest Test Plan

Completed 02/13/2020

By:

Eric Buck and Nicholas Lasichan

Index

Cover Page 1
Index 2
Introduction 3
Requirements 4
Test Cases 5
Traceability Matrix 9
Defects Found 10
Who Did What 11

Introduction

The major difficulty encountered by the team was learning how to run the .jar application. Our previous experience with java was based only on creating and running applications using java in the IDE Eclipse. This was the first time either other one of us had to run a .jar application. Once the initial issue was resolved by using the command java -jar followed by the filename of the .jar application, in this case, coffeemaker.jar we started exploring the program.

The other concern was in the process of creating a test plan itself. Neither tester has ever created one and so this will be our first. Our plan will be to follow the format as much as possible given to us by our instructor, Dr. Jonathan Hardwick.

Requirements[1]

- ***FUN-ITERATION –*** *At each iteration of the game, the user shall be able to enter one of the six commands – "N" to go North, "S" to go South, "L" to Look for items, "I" for inventory, "H" for Help, or "D" to Drink.*
- ***FUN-UNKNOWN-COMMAND –*** *If a player enters a command not specified by* ***FUN-ITERATION****, the system shall respond with the phrase "What?".*
- ***FUN-INPUT-CAPS –*** *The system shall be case-insensitive in regards to input values; that is, it shall accept capital and lower-case letters and treat them as equivalent.*
- ***FUN-MOVE –*** *The system shall allow a player to move North only if a door exists going north and south only if a door exists going south.*
- ***FUN-WIN –*** *The player shall win the game if and only if Coffee, Sugar and Cream have been collected by the player and then drunk.*
- ***FUN-LOSE –*** *The player shall lose the game if and only if the player Drinks but has not collected all of the items (Coffee, Sugar, and Cream).*
- ***FUN-INVENTORY –*** *Upon entering "I" for inventory, the player shall be informed of the items that he/she has collected (consisting of Coffee, Sugar, and Cream).*
- ***FUN-LOOK –*** *Upon entering "L" for Look, the player shall collect any items in the room and those items will be added to the player's inventory.*
- ***FUN-HELP –*** *Upon entering "H" for Help, the player shall be shown a listing of possible commands and what their effects are.*
- ***FUN-UNIQ-ROOM –*** *Each room in the house shall have a unique adjective describing it.*
- ***FUN-UNIQUE-ROOM-FURNISHING –*** *Each room in the house shall have one and only one unique furnishing visible to the user upon entering the room.*

[1] Requirements supplied by Dr. Jonathan Hardwick from https://github.com/it480-spr20/week-4-test-plan-team-twinkie-purple-table.

Test Cases

Test cases:

1. Lower case N.
    a. Requirements covered: FUN-INPUT-CAPS
    b. Summary: Ensure that the user can go north but typing lower case n.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type lower case “n” in the command line
        ii. Press enter
    f. Output values
        i. Program writes “What?” in command prompt
    g. Post Conditions
        i. User hasn’t made additional movement in game
        ii. Game still running
        iii. Instructions for the game are displayed
2. Help function
    a. Requirements covered: FUN-HELP
    b. Summary: Ensure that the help function is accessible by the user
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “h” or “H” into the command line
        ii. Press enter
    f. Output values
        i. Program writes “What?” in command prompt
    g. Post conditions
        i. Game still running
        ii. Instructions for the game are displayed
        iii. User hasn’t made additional movement in game
3. Going in direction that has no specified door
    a. Requirements covered: FUN-MOVE
    b. Summary: Ensure that program behaves correctly when user goes in a direction that does not have a specified door.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed

d. Input values
    i. Typed commands by the user
e. Execution steps
    i. Type either “N” or “S”
    ii. If the program says that there are two doors present, repeat step 1
    iii. Choose to go in the direction that does not have a specified door
f. Output values
    i. Program writes “You are in a magical land! But you are returned to the beginning!” in command prompt
g. Post conditions
    i. Game is still running
    ii. Instructions for the game are displayed
    iii. User is back to where they started in the game.

4. “Look” Command Functionality
    a. Requirements covered: FUN-ITERATION, FUN-LOOK
    b. Summary: Ensure that the “look” command is functioning correctly.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “S” or “N” and press enter
        ii. Type “L” and press enter
    f. Output values
        i. Program writes result of the user’s in game search in the command prompt.
    g. Post Conditions
        i. Game is still running
        ii. Instructions for the game are displayed
5. “Inventory” Command Functionality
    a. Requirements covered: FUN-ITERATION, FUN-INVENTORY
    b. Summary: Ensure that the inventory command is functioning correctly.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “S” or “N” and press enter
        ii. Type “I” and press enter
    f. Output values
        i. Program gives a description of the user’s inventory in the command prompt.
    g. Post Conditions
        i. Game is still running
        ii. Instructions for the game are displayed

6. Winning
    a. Requirements covered: FUN-ITERATION, FUN-WIN
    b. Summary: Ensure that the user can win the game when they meet proper conditions
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “S” or “N” and press enter
        ii. Type “L” and press enter
        iii. Type “I” and press enter
        iv. Repeat steps 1-3 until Coffee, Creamer, and Sugar are in the user’s inventory
        v. Type “D” and press enter.
    f. Output values
        i. Program tells the user that they won
        ii. Program writes “Exiting with error code 0”
    g. Post Conditions
        i. Program terminates
7. Losing
    a. Requirements covered: FUN-ITERATION, FUN-LOSE
    b. Summary: Ensure that the user can lose the game when they do not meet the proper conditions to win.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “S” or “N” and press enter
        ii. Type “L” and press enter
        iii. Type “I” and press enter
        iv. Repeat steps 1-3 until Coffee, Creamer, and Sugar are in the user’s inventory
        v. Type “D” and press enter.
    f. Output values
        i. Program tells the user that they won
        ii. Program writes “Exiting with error code 0”
    g. Post Conditions
        i. Program terminates
8. Unique Rooms
    a. Requirements covered: FUN-ITERATION, FUN-UNIQ-ROOM, FUN-UNIQ-ROOM-FURNISHING
    b. Summary: Ensure that the each room in the game is unique and has only one unique visible furnishing.
    c. Preconditions

i. Game is already running
ii. Instructions for the game are displayed
iii. User has not made any movements in the game

d. Input values
    i. Typed commands by the user
e. Execution steps
    i. Type “S” or “N” and press enter
    ii. Repeat step 1
    iii. If the description of the second room is the same as the description of the first room, stop.
f. Output values
    i. Program gives description of the current room in game.
g. Post Conditions
    i. Game is still running
    ii. Instructions for the game are displayed

9. Unknown Command
    a. Requirements covered: FUN-ITERATION, FUN-UNKNOWN-COMMAND
    b. Summary: Ensure that the program responds correctly when the user enters an unknown command.
    c. Preconditions
        i. Game is already running
        ii. Instructions for the game are displayed
        iii. User has not made any movements in the game
    d. Input values
        i. Typed commands by the user
    e. Execution steps
        i. Type “?” and press enter
    f. Output values
        i. Program displays “what?” in the command prompt
    g. Post Conditions
        i. Game is still running
        ii. Instructions for the game are displayed

iii.

Traceability Matrix

Requirement	Req. Description	Testcase ID	Status
FUN-ITERATION	Commands used: N, S, L, I, H, D	1,2,4,5	1 – Failed 2 – Failed 4 – Passed 5 - Passed
FUN-UNKNOWN-COMMAND	Program response to unknown command.	9	9 - Failed
FUN-INPUT-CAPS	Program recognizes upper and lower-case commands and treats them the same.	1	1 - Failed
FUN-MOVE	Program only allows user to move N or S if a corresponding N or S door exists.	3	3 - Failed
FUN-WIN	Program allows user to win if and only if all three ingredients are collected and then drunk.	6	6 - Passed
FUN-LOSE	User loses if the user inputs “D” at any point prior to collecting all three ingredients.	7	7 - Passed
FUN-INVENTORY	Allows user to see what items they have collected.	1, 5	1 – Passed 5 - Passed
FUN-LOOK	Allows user to look for and collect any ingredients that may be in the current room.	1, 4	1 – Passed 4 - Passed
FUN-HELP	Allows user to see a list of commands and a description of what they do.	1,2	1 – Failed 2 - Failed
FUN-UNIQ-ROOM	Each room has a unique adjective describing it.	8	8 - Passed
FUN-UNIQUE-ROOM-FURNISHING	Each room has one unique room furnishing.	8	9 - Passed

Defects Found

Bug 1:

Description: Lower-case “N” not recognized as a valid command.

Reproduction Steps: In each of the 6 rooms on multiple play throughs, both testers attempted go north where possible, through the N marked door, by using the lower-case “N”.

Expected Behavior: Upon using the lower-case “N” the user should be able to move north into the next room where there is a N marked door.

Observed Behavior: When the user uses the lower-case “N”, the game returns the **FUN-Unknown-Command** output of “What?”.

Bug 2:

Description: Pressing the “H” key, either upper-case or lower-case, not recognized as a valid command.

Reproduction Steps: In each of the 6 rooms on multiple play throughs, both testers attempted to call the **Fun-Help** by pressing the “H” key using both upper-case and lower-case “H”.

Expected Behavior: Upon entering “H” for Help, the player shall be shown a listing of possible commands and what their effects are.

Observed Behavior: Upon entering “H” for Help, the system returns the **FUN-Unknown-Command** output of “What?”.

Bug 3:

Description: In the first room where the only door that exists is the N door, user can still go south.

Reproduction Steps: On every play through, the testers were able to enter the “S” key in the first room and leave that room going south.

Expected Behavior: The user should be unable to move south from the first room when entering the “S” key and some type of warning should be given stating that there are no doors in that direction.

Observed Behavior: Upon entering the "S" key, the system returns *"You are in a magical land! But you are returned to the beginning!"* and the user is placed back into the starting room of the program.

Bug 4:

Description: In the last room where the only door that exists is the S door, user can still go north.

Reproduction Steps: On every play through, the testers were able to enter the "N" key in the last room and leave that room going north.

Expected Behavior: The user should be unable to move north from the last room when entering the "N" key and some type of warning should be given stating that there are no doors in that direction.

Observed Behavior: Upon entering the "N" key, the system returns *"You are in a magical land! But you are returned to the beginning!"* and the user is placed back into the starting room of the program.

Who Did What

Manual Testing – Eric Buck, Nicholas Lasichan

Write Report – Eric Buck

Create Test Cases – Nicholas Lasichan

Create Traceability Matrix – Eric Buck

Identify Bugs – Eric Buck, Nicholas Lasichan

Final Edits – Eric Buck and Nicholas Lasichan