Ali Sytsma

Professor Arias

Software Development 1

7 April 2017

Final Writeup

Abstract

The program is a text-based adventure game set in the Medieval Ages. Its protagonist is a young farm girl who is sent to court by her uncle to gain influence or money to save her family's farm. It focuses heavily on user choices, and what they decide throughout the game will determine what happens. The game changes depending on their relationships with other characters, how they decide to interact with other characters, and what they decide to do at decision points. The way they play also will impact the ending of the game, of which there are many.

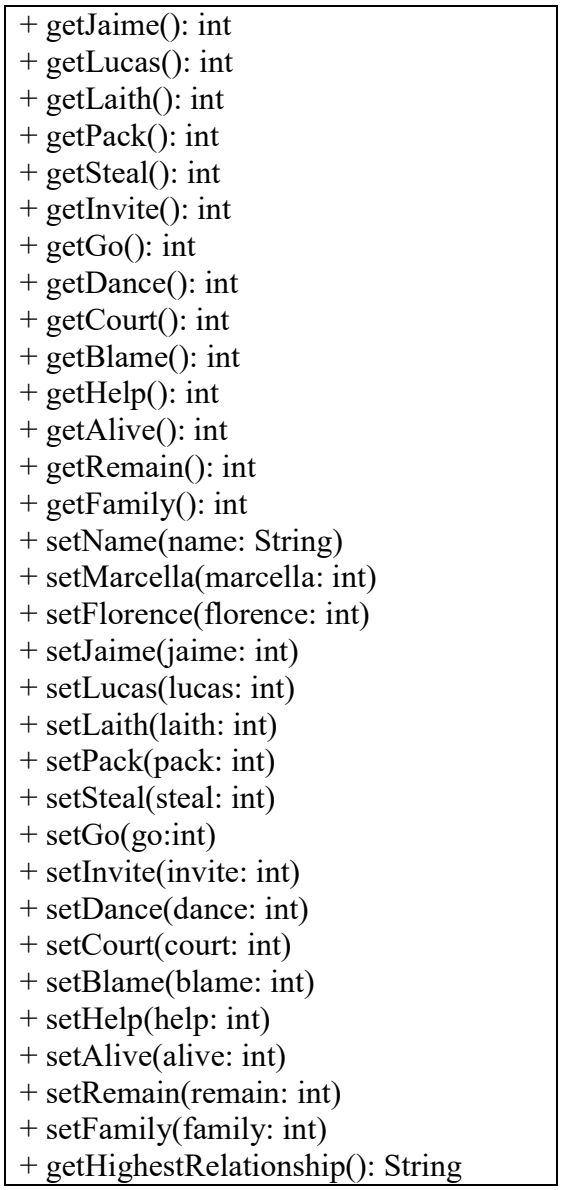
Introduction

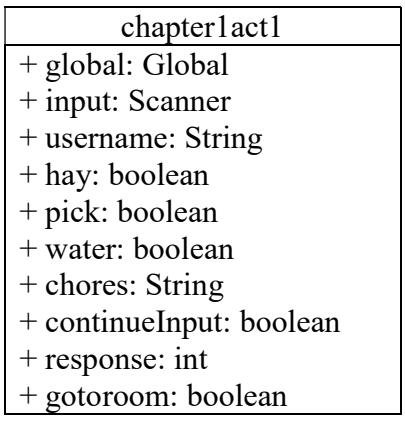
The motivation for this project is to create a fun, detailed, choice-based text game. Giving the player options and different outcomes creates a more interactive experience than a linear game would. There are many different endings that the player can access, as well as many different ways to play the game, providing replay value. If they dislike the ending they received, they can play a different way in their next play-through and make different decisions. In this paper, I will discuss the detailed system description, the requirements, the literature survey, and the user manual.

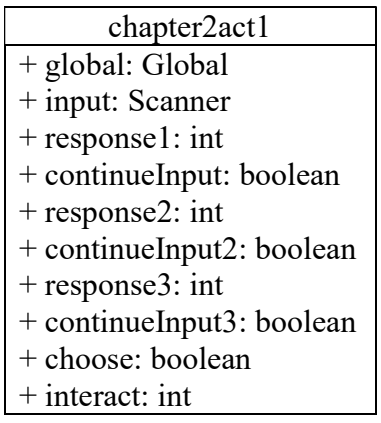
Detailed System Description

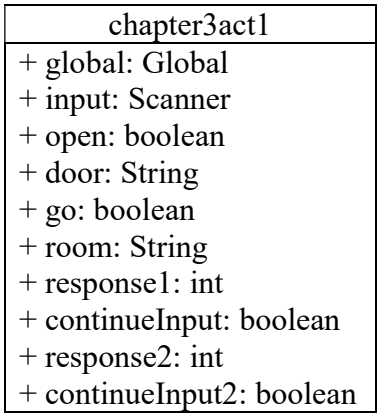
The program was designed so that all of the chapters reference the Global class while the chapters are in the Game class. This way, when a variable is altered in one chapter, it is altered for the future ones as well. For example, if your relationship with Laith is 5 by the end of chapter 1, it will be stored in the Global class so that your relationship will remain at 5 for the beginning of chapter 2. The program uses many switches and if statements for decisions, including nested ones, as well as while loops to re-prompt the user if their input is invalid. The program also contains input mismatch exceptions to catch input that does not match what the program asked for. This way, the game does not quit but instead re-prompts the user. The game is a text-based game, but is different from most text adventures. Most text adventures will have commands such as “examine chest” or “go north.” This game provides more structure to the user in order to provide the choice aspect and have their decisions matter rather than it being a puzzle game.

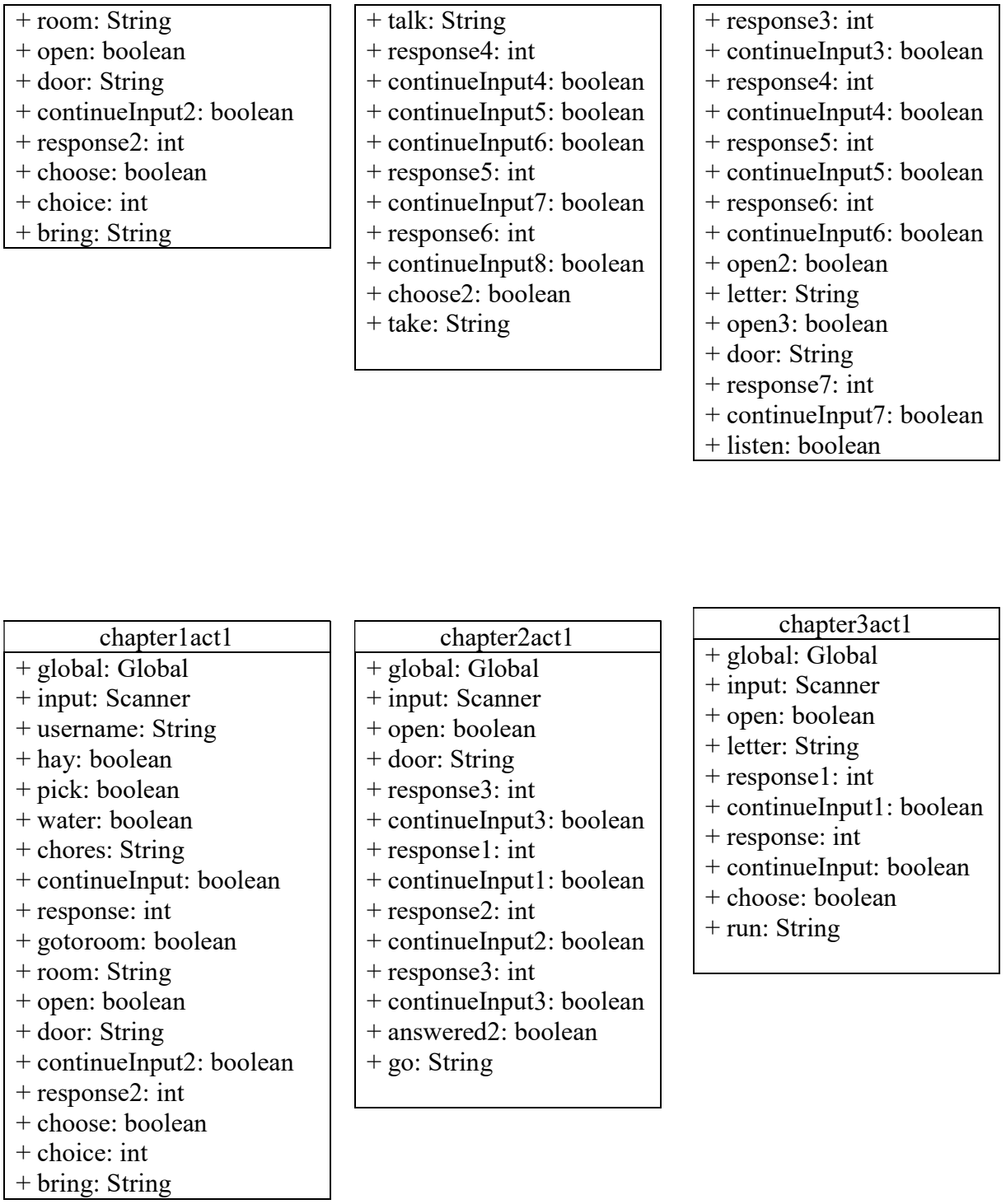
Global
- marcella: int - florence: int - jaime: int - lucas: int - laith: int - pack: int - steal: int - invite: int - go: int - dance: int - court: int - blame: int - help: int - alive: int - remain: int - family: int
+ enter() + getName(): String + getMarcella(): int + getFlorence(): int

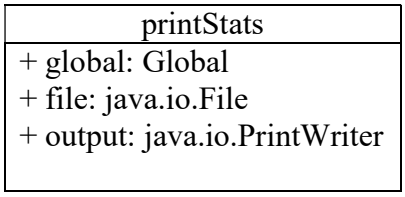

+ getJaime(): int
+ getLucas(): int
+ getLaith(): int
+ getPack(): int
+ getSteal(): int
+ getInvite(): int
+ getGo(): int
+ getDance(): int
+ getCcourt(): int
+ getBlame(): int
+ getHelp(): int
+ getAlive(): int
+ getRemain(): int
+ getFamily(): int
+ setName(name: String)
+ setMarcella(marcella: int)
+ setFlorence(florence: int)
+ setJaime(jaime: int)
+ setLucas(lucas: int)
+ setLaith(laith: int)
+ setPack(pack: int)
+ setSteal(steal: int)
+ setGo(go:int)
+ setInvite(invite: int)
+ setDance(dance: int)
+ setCcourt(court: int)
+ setBlame(blame: int)
+ setHelp(help: int)
+ setAlive(alive: int)
+ setRemain(remain: int)
+ setFamily(family: int)
+ getHighestRelationship(): String


chapter1act1
+ global: Global
+ input: Scanner
+ username: String
+ hay: boolean
+ pick: boolean
+ water: boolean
+ chores: String
+ continueInput: boolean
+ response: int
+ gotoroom: boolean


chapter2act1
+ global: Global
+ input: Scanner
+ response1: int
+ continueInput: boolean
+ response2: int
+ continueInput2: boolean
+ response3: int
+ continueInput3: boolean
+ choose: boolean
+ interact: int


chapter3act1
+ global: Global
+ input: Scanner
+ open: boolean
+ door: String
+ go: boolean
+ room: String
+ response1: int
+ continueInput: boolean
+ response2: int
+ continueInput2: boolean


+ room: String
+ open: boolean
+ door: String
+ continueInput2: boolean
+ response2: int
+ choose: boolean
+ choice: int
+ bring: String
+ talk: String
+ response4: int
+ continueInput4: boolean
+ continueInput5: boolean
+ continueInput6: boolean
+ response5: int
+ continueInput7: boolean
+ response6: int
+ continueInput8: boolean
+ choose2: boolean
+ take: String
+ response3: int
+ continueInput3: boolean
+ response4: int
+ continueInput4: boolean
+ response5: int
+ continueInput5: boolean
+ response6: int
+ continueInput6: boolean
+ open2: boolean
+ letter: String
+ open3: boolean
+ door: String
+ response7: int
+ continueInput7: boolean
+ listen: boolean
chapter1act1
+ global: Global
+ input: Scanner
+ username: String
+ hay: boolean
+ pick: boolean
+ water: boolean
+ chores: String
+ continueInput: boolean
+ response: int
+ gotoroom: boolean
+ room: String
+ open: boolean
+ door: String
+ continueInput2: boolean
+ response2: int
+ choose: boolean
+ choice: int
+ bring: String
chapter2act1
+ global: Global
+ input: Scanner
+ open: boolean
+ door: String
+ response3: int
+ continueInput3: boolean
+ response1: int
+ continueInput1: boolean
+ response2: int
+ continueInput2: boolean
+ response3: int
+ continueInput3: boolean
+ answered2: boolean
+ go: String
chapter3act1
+ global: Global
+ input: Scanner
+ open: boolean
+ letter: String
+ response1: int
+ continueInput1: boolean
+ response: int
+ continueInput: boolean
+ choose: boolean
+ run: String



Requirements

As the main purpose of the game is to be choice-based and player directed, the game must have multiple outcomes and provide a difference experience based on how the user plays. For example, having a negative attitude will result in characters not liking the player, and being pleasant will result in more approval. It is apparent in the way the characters talk to the player how they feel about them. The different print statements that are included not only make the dialogue flow nicely, but alter what the character's say in response to the player's dialogue choices. There are three main categories in which the ending can vary. These are what happens to their family, if they are allowed to remain at court, and if they survive or not. Because of these potential combinations, there are many different possible endings the user can experience. Another requirement is that, upon ending the game, a file will be made with all of their stats and their ending so that players can compare amongst themselves.

Literature Survey

There are many text-based games out there. However, I was only able to find one that dealt with a similar theme, which was "Affairs of the Court." This game is similar in that it is a choice-based game that focuses on a young woman going to court, but the similarities end there.

User Manual

The user interacts with the system by typing. The game will always prompt the user so they will know what they are supposed to put. For dialogue, they enter the corresponding number to the line they want their character to say. If the user enters a number outside of the dialogue options, then they will be re-prompted for valid input. For other situations, they type in commands, such as “take hairbrush” or “go to room.” Again, if the user fails to enter what the program is looking for, they will be re-prompted.

Conclusion

The goal of the program is to provide the user with a fun, interactive, and choice-driven experience. Hopefully, because their choices shape the game, there will be replay value and players will enjoy experiencing different outcomes. If I had more time, I would have made it so that the player could customize their character a little more. Perhaps they could have the option to change their hair color or set different stats, such as charisma or intelligence, and these could influence how the game goes.

References/Bibliography

"JavaScript Errors - Throw and Try to Catch." *W3Schools*. Web. 03 May 2017.

<https://www.w3schools.com/js/js_errors.asp>.