Text Adventure RPG

Hunter Reeves, Billy Gibson

Abstract

Text Adventure RPG is a game that is a throw back to the first RPGs of its kind. It focuses on literal text based output and all of the graphics are centered around unicode characters. Our target audience is someone who enjoys fun RPG style games. The game has character creation where the player can choose one of three classes. Controls exist for the player to move around the map, along with a story to interact with. The combat system is more or less fleshed out, with a somewhat complex leveling system as well.

1. Introduction

Text Adventure RPG is a text-based RPG adventure game that allows the player to explore a text-filled world, pick a class, level up, fight enemies, and gain gold. Game development is an interesting area and we are looking forward to learning more about what all of that entails. The majority of the game will be text based. We hope to achieve this with minimal graphics, aside from a few images to make the visuals a bit more interesting. Our way of going about this was through the use of unicode characters.

1.1. Background

Our game, as has been previously mentioned, is a text based adventure. Text based adventure games involve text information around every event that occurs within the game. It will be strictly GUI based with text for every event that happens. From picking up loot and gaining gold/experience to fighting rats, goblins, and ogres.

Creating a world for others to explore is a very exciting task to take on. Being able to take part in this project is something we have both always wanted to do. This project should not be too difficult, with a scope that won't be too large for the semester. The game will also be single player, for simplicity.

1.2. Challenges

The feature that will be most challenging to add to this game will be the interaction of the player with the items, or loot, they come across in their journey. Another challenge that we came across during development was the character creation. That ended up being a bit more complex than previously thought.

2. Scope

Due to the game being more text-based, the scope is easier to handle. It involves some basic gameplay mechanics, the storyline, and how the player will interact with the world. It also includes our two stretch goals that we want to complete if we have the time.

1. Interactive GUI for Gameplay
This will be the main window of the game where you will see your level, gold, experience, any quest updates and information, and the basic player controls for moving area to area.

2. The Story
The story will be pretty simple given the time we have. It will most likely involve the player escaping a prison to start our their new life.

3. Class Selection
We are thinking about three classes. A warrior, mage, and thief. Pretty basic selection and shouldn't take much to implement. They will vary in starting stats and abilities/weapon choice.

4. Gold and Leveling System
Pretty straightforward. Include a way to obtain gold and perhaps have a small shop to buy and sell goods. As well as the

leveling system based on gaining experience from quests and defeated enemies in the world.

Stretch Goals

5. Obtain/Use Loot
Obtaining and using loot will be as simple as seeing some text appear and then the appropriate changes regarding the loot obtained or used. It could be a weapon swap, using a potion, or getting a quest item.

6. Different Locations/Areas
The starting location will be a sewer type location after the player gets out of prison. We hope to also add a few more areas. Perhaps a city and a forest for the player to explore.

2.1. Requirements

The requirements were gathered by researching and thinking of what would be the best for our purposes in the game. The functional are the basic abilities that the user should be able to do at any point. The non-functional requirements involve keeping the game stable and of a higher standard.

2.1.1. Functional.

- The user must have the ability to equip and unequip their weapons/armour.
- The user should have private inventories that can't be accessed but from themselves.
- The user should be able to run or fight from any battle.
- The user should have the ability to quit the game at any point.

2.1.2. Non-Functional.

- Stability - the game should always run with as little problems as possible
- Quality - the quality of the game should be to our highest expectations

2.2. Use Cases

Use Case: 1
Name: Select a Class
Description: The player will need to do this almost at the start of the game. They will choose whether they want to be a Warrior, Mage, or Thief class. Once they pick the class they want, they will click "Select" and they will start the game with the class they selected.

1) Player clicks on a class within the GUI.
2) Player reads and decides on which class they want to have.
3) Player clicks "Select" on the class they want and obtain stats associated with that class.

Use Caser: 2
Name: Attack an Enemy
Description: You will come across enemies in the game, some pretty immediately into starting out. When you enter combat with a rat for example, the rat will attack and dialogue will appear to indicate this. Something like "A rat has appeared. It caused X damage." The player will now see an option to attack the rat with their weapon. More dialogue will appear to show this. "You have caused X damage to the rat."

1) Player initiates combat with an enemy.
2) Player fights the enemy until one of them dies.
3) Player will either defeat the enemy or the game closes (permadeath).

Use Case: 3
Name: Obtain Loot
Description: There will be moments in the game where the player can pick up new gear, quest items, or gold. When the player encounters something they can loot, a text dialogue describing the item will appear. The player will then get the option to either pick the item up or ignore the item.

1) Player comes across loot and receives dialogue about the loot.

2) Player decides whether or not they pick the loot up or ignore it.
3) Player adds loot to their inventory or simply walks away from the loot.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Select a Class	Player	Medium	1
2	Attack an Enemy	Player	Medium	1
3	Obtain Loot	Player	Easy	2

TABLE 1. USE CASE FOR SOME BASIC PLAYER INTERACTIONS

2.3. Interface Mockup

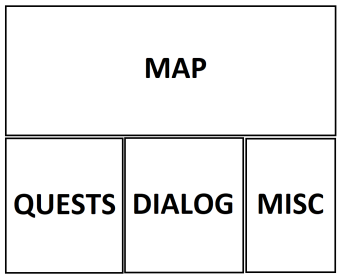

Figure 1. The base game GUI.

3. Project Timeline

This is the timeline of our project. The following figure shows all of the dates, as well as what was or will be completed, the date it was started and the date it is aimed to be finished, as well as some additional notes to go along with the timeline. This is located below at the bottom of this proposal.

4. Project Structure

The game starts off with a character creation screen, where the player chooses their name, and class. Once this is selected, the window closes and the main game window appears. The player's name and class choices will be reflected within the main game window. The player will have access to their dialogues from any quests, current and completed quests, and the map of the game. The map is based on a tile set of unicode characters, which is 32x8 (256) tiles. A player can enter combat with enemies and level up.

4.1. UML Outline

The UML that lists the player creation in it's entirety is down below.

4.2. Design Patterns Used

The first design pattern was utilized in character creation - the Abstract Factory. The other design pattern is the Observer pattern, which is used for the story and quests. It also handles the three boss fights as well.

5. Results

1. Interactive GUI for Gameplay
The GUI for the overall gameplay wasn't too difficult to get going. Get some map controls, combat controls, and we were home free.

2. The Story
The storyline is also pretty straight forward. Have the player interact with the story and complete some quests for completeness.

3. Class Selection
Class selection/character creation was difficult. It was hard to figure out how to handle other classes being null when you only select one class each game run. Because of this, we have tons of null checks in our code. It's sloppy, but I had no idea how else to manage this issue.

4. Gold and Leveling System
Gold/Leveling system as well as the combat system wasn't too hard to implement. The only issue is all of the conditionals for the logic to get it all to work. This is also sloppy, but again - not really sure how else to tackle this kind of mechanic.

5.1. Future Work

I am not sure if we are going to continue work on this or not. Perhaps if we added more to it and cleaned it up a bit, we could potentially put it on the Steam gamestore. But we are not sure. We accomplished everything we set out to do and then some. The only thing we wish we had implemented was the inventory system and items. All in all, this was a fun project. We enjoyed working on it, despite dealing with the wonkiness of WPF. Perhaps we will transfer this idea over to a sufficient game engine.

ACTIVITY	START	END	NOTES
Project Start	9/5/2019		Work done together
Project Proposal 1	9/5/2019	9/24/2019	Work done together
Project Propsal Update	10/1/2019	10/17/2019	Work done together
Project Prsentation Work	10/8/2019	10/8/2019	Work done together
Presentation Day	10/17/2019	10/17/2019	Work done together
Interface Work	10/20/2019	10/22/2019	Completed: Hunter
Hardcoded Map/Map Algorithm	10/30/2019	11/9/2019	Completed: Billy
Class Selection	10/27/2019	11/3/2019	Completed: Hunter
Project Update	11/17/2019	11/17/2019	Work done together
Story and Background for Game	11/13/2019	11/20/2019	Completed: Billy
Obtain/Use Loot	11/18/2019	11/24/2019	Not Completed
GUI Work for Gameplay	11/20/2019	11/28/2019	Completed: Billy
Combat/Leveling System	11/22/2019	12/5/2019	Completed: Hunter
Project End	12/5/2019		

Figure 2. Overall project timeline from start to finish with possible dates.

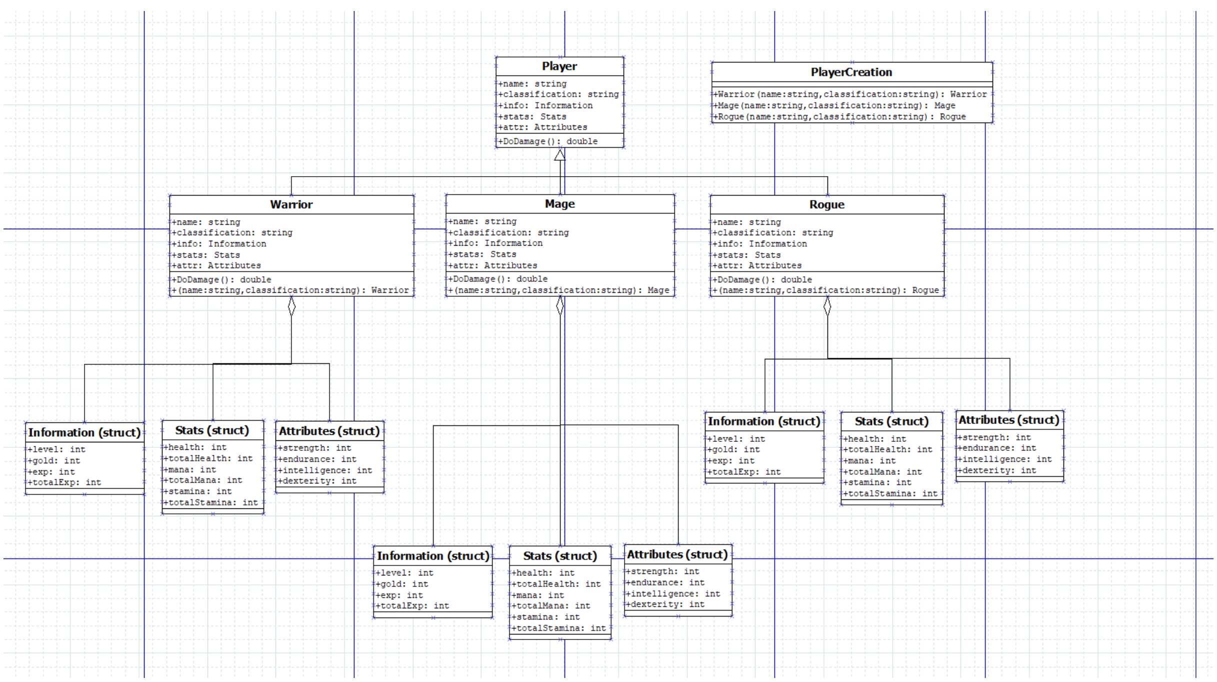

Figure 3. UML for player classes and character creation. Abstract Factory Pattern.

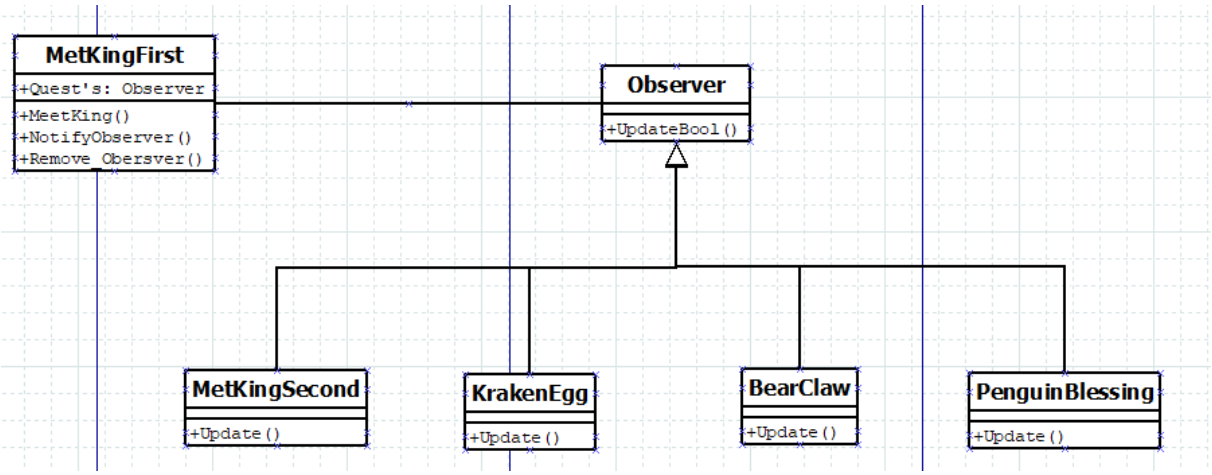

Figure 4. UML for obtaining the four storyline items. Observer Pattern.