Maze Game Project - IT'S RAW!
CMPT 276 - Fall2023

Group 24 Phase 2 Report

Rashed Hadi

Timmy Tsai

Raymond Chan

Raymond Zhou

Due: November 12, 2023

School of Computing Science

Simon Fraser University

The Approach

Video game design, whether in a large commercial setting, or in a small group setting is a complicated and tedious task. When four individuals come together with the intention of creating a 2D game, with no game design or development experience between them, they are setting themselves up for a journey of discovery and enrichment. Our journey started with a lot of research, not limited to multiple game design Youtube videos along with a plethora of Reddit and Stackoverflow threads. Going down this game development rabbit hole we ended up with the realization that there is a lot more that encapsulates a game that we initially expected, finally deciding to settle down and start by implementing the basics of what we have discovered so far.

We started off by creating all the classes we deemed necessary for our journey, taken from our Phase 1 UML diagram, then proceeded by implementing a basic game loop and game panel. Staring at the black rectangle moving around the game window with a tingling feeling of achievement motivated us to continue pushing forward, to test our skills, discovering where we could take this journey. After grouping related parts of the project together, and splitting up, we each delved into creating our parts, working together when necessary through regular discord meetings aided by good communication. We focused on helping each other succeed by updating each other regularly and creating an environment where we all felt comfortable to both receive and provide help which ultimately allowed us to achieve our goals in the end.

Midway Plan

At the halfway point of our game-making journey, our first milestone/goal was to get the basic stuff sorted out. We focused on creating the game board, figuring out how our main character works, making a user-friendly interface, and setting up the reward system. We kept things simple by not worrying too much about how everything would work together just yet. The idea was to see how these main pieces fit together in our game.

By doing this, we wanted to make sure that each part of the game, like the game board and the main character, was working well on its own before we made them interact with each other. This way, we could really understand how everything fits into the big picture of our game. Then once we've got these core pieces in place we moved on to our second milestone which was making things like characters, items, and the enemy interact with each other. For example, having a score to keep track of rewards, having oxygen levels, etc. By taking it step by step, we made sure we had a solid foundation to build on.

The Design Changes

As our journey progressed, we were constantly reevaluating our design plan, reconsidering the initial decisions we made prior to understanding the 2D game development process. Initially, we intended to create a game with multiple maps and difficulties which later, along with many other ideas were brought to a halt due to the many challenges we ended up facing in the short time span of the journey, along with the reality of our busy schedules. The following are the main design changes we made along our journey coupled with their causes:

Changes: Having multiple levels with various difficulties.
Reason: The main reason we ended up reconsidering having multiple levels comes down to time constraints. Other than the fact that we were all new to game development and had a lot to learn, the majority of our implementation phase occurred during a busy midterms season which limited the amount of time we could spend on implementation, making it unrealistic for us to progress that far into development.
Changes: Items having despawn timers.
Reason: We decided to keep items from despawning since we initially thought that we would have items spawning around the map while the game loop is continuing rather than having all the objects and assets painted in the start of the game loop. We decided to not implement this because it would have made the game too difficult.

Team Management

Our approach for this phase was to divide roles based on our phase 1 UML diagram. Each of us worked on separate git branches and communicated what changes were made. Our first meeting consisted of splitting up the work in terms of what classes had similar behaviours. Subsequent meetings involved discussing what updates or progress was made on our classes and helping each other with issues that arose. We mainly communicated through Discord regularly on voice call and chat and were able to resolve any issues we faced by sharing our screen with each other. On the next page is a breakdown of what each person worked on during this phase.

Rashed • Created the main character class • Created the user/key input for the main character movement • Created moving enemy class • Designed main character and moving enemy sprites
Timmy • Created the game UI class • Created the different game states and menu screen • Created oxygen levels for the main character • Designed sprites for oxygen UI and oxygen tanks

Raymond Chan • Created regular, bonus rewards, punishments, and exit • Created object interaction for main character • Designed sprites for rewards, punishment, barriers, and exit • Worked on collision
Raymond Zhou • Created the Board • Created the games collision system • Created the cell system for the board • Designed the sprites for cells and walls • Worked on moving enemy

External Libraries Used

java.io.IOException	Used to check failed input or output or to catch exceptions when "try" was used.
java.imageio.ImageIO	Reading images
java.awt.Graphics2D	Using graphics to draw screens, extends graphics.
java.awt.image.BufferedImage	Used to load images to the screen
java.awt.Rectangle	For main character collision
java.awt.Color	To choose colours for text and background
java.awt.Dimension	Used for height and width of the screen
java.awt.Graphics	Used to draw objects on screen with our draw method
java.awt.AlphaComposite	Used for transparent effect of the board
java.awt.event.KeyEvent	Receive keyboard inputs from user

java.awt.event.KeyListener	Receive keyboard inputs from user
java.awt.image.RescaleOp	For transparent blue effect on map and objects in draw methods
javax.swing.JFrame	For the window screen of the game
java.awt.Font	Renders different fonts

Code Quality Measures

To ensure that our code maintained the highest quality level, we took advantage of the following principles and practices, some of which were taught in class, and others learnt from past experiences. One of our commonly used principles was the Open/Closed Principle, which we applied to ensure adaptability and extensibility, particularly in dealing with similar objects. At the same time, we used concurrency threads extending the SuperObjects class, allowing us to branch into various types of objects and entities present around the map. Moreover, we used encapsulation along with the Separation of Concerns principle to reduce the coupling present and maximise cohesion throughout our game. Finally, to enhance our source code's comprehensibility, we included various comments, some highlighting the roles of methods, and others describing the functionality of one single line. All together, these principles and practices allowed us to complete our game without finding hours of housekeeping and refactoring left for us upon completion.

Challenges

One of the major challenges we encountered during the development of our game was effective time management. We had planned a variety of features for our game which resulted in some of the design changes listed above due to time constraints. The coordination among team members was another challenge. For example, certain classes within the game depended on one another, requiring communication and coordination to ensure each person completed their assigned tasks. Additionally, the merging of our individual branches into the main branch presented some challenges as we needed to make sure what lines of code we needed to change or remove. Overall, although we encountered many hiccups along the way, we worked around them as a team, communicating whenever necessary.