Grave Robber

CMPT 276 Group 15 Final Project

Dylan Rowsell - 301163100
Yuncong Chen - 301416502
Gray Keng - 301451848
Seenan Iftekhar - 301371941

April 16, 2022

Game Description

Overview

Grave Robber is a 2 dimensional top down game available for the PC where players are able to slink through graveyards as a grave robber seeking treasure. The grave robber must be cautious and avoid skeletons, zombies, and traps which will harm them. Using the keyboard, a player will be able to move up, down, left, or right by using the WASD keys. While playing Grave Robber, the player will be able to view their score and elapsed time playing the game at the top of the screen. Only once the grave robber is able to collect all the red hearts will they be allowed to leave the graveyard, however, the extra brave robbers will also be able to steal elusive golden hearts if they are lucky and fast enough.

Obstacles

When navigating the graveyard, the grave robber must be extra careful of the dangers that lie ahead. Scattered around the map are spikes which will hurt the player if they are stepped on. The robber only has a limited amount of health, so they must be careful to not get too harmed. The player must also avoid brainless zombies which run quickly in straight lines. They may not be smart, but they are very fast and will kill the robber if they catch him. A spooky skeleton will also be slinking around the graveyard. Be careful, this skeleton is much smarter than the zombies. If your health reaches zero then you will need to restart the game!

Goals

The grave robber must collect all the red hearts on the map to be able to leave through the exit at the top right of the map. Every time a heart is picked up the player will become a little stronger. In the event that the play picks up a golden heart, they will receive triple the score of a red heart. Golden hearts are not needed to complete the game.

Video Link

https://www.youtube.com/watch?v=KPb4RuY-ECs

Jar and Javadocs Location

An example JAR file can be found in the output folder.
An example of javadocs can be found in the javadoc folder.
Both JAR and Javadocs will be created in the target folder when running mvn clean package.

Game Design

While designing Grave Robber many things changed and some managed to stay the same. In this section it will be discussed how the game changed, and how it was still able to meet all the original requirements given in Phase 1.

What has Stayed the Same?

1. Overall design from mockup
2. Enemy mechanics
3. All characters move on ticks
4. All Red hearts must be collected to win
5. Bonus hearts as Bonus treasure
6. Trap will hurt character

What Has Changed?

1. Mummy enemy was changed to Skeleton
2. Exit tile was made separate from the entrance
3. Remove the Room
4. Remove the key
5. Remove the weapon
6. Remove the Boss
7. Remove the Door
8. Sound was added
9. Skeleton AI that would use DFS algorithm to find the hero character was changed to a basic closest euclidean distance heuristic

Justification of Changes

1. The mummy was changed to Skeleton because we were unable to find mummy sprite assets.
2. The exit tile was made separate from the entrance to meet the phase 1 requirement instead of our own.
3. The secret room is difficult to create, so it was converted to gold heart which is the bonus treasure.
4. The key was removed because the secret room is removed
5. The weapon was removed because it doesn't not meet the requirement of phase 1 which means it is more than the requirement and difficult to work on.
6. The Boss was removed because it was not needed
7. The Door was removed because the room was removed
8. The Sound will be played when the character was killed by enemies, picking up the rewards or entering the end interphase of the game.

9. Creating the DFS algorithm with the way our map worked was very hard to implement in the Java language and also made things very slow when it was attempted. Our heuristic of the skeleton looking for the hero does enough of a good job.

Lessons Learned

When designing the game the team learned to become more flexible. Because it can be difficult to estimate a development task, hard choices need to be made to achieve deadlines. In the future we would prefer to create unit tests as the system was being built opposed to after the fact. Our team believes that if this was done, then the structure of the project may have had lower coupling.

Working together as a team can be difficult when people have different schedules and course loads. Using the agile method of project organization was useful, but we mainly only used it in phase 2. It would have been better if we continued to use this method throughout phase 3 and 4.

We also learned that code should be reviewed and checked over by many team members more frequently throughout the development process. That way we can refactor important portions of code as we go instead of struggling to refactor all the code at the end.