CMPT276 Phase 4 Report

Sean Chan - smc26
Kaj Grant-Mathiasen - kgrantma
Ashhal Vellani - ava47
Bavneet Hothi - bkh6

The Game

Our game theme involves a miner who has found himself in a deep dark cave. You must navigate through the cave and collect all the gold before making your way to the exit. Occasionally, a diamond will appear and if you are able to collect it before it disappears, you will be rewarded with extra score! Be careful however, as there are several dangers like bats and snakes within the cave.

Compared with our initial design, we made some heavy modifications to arrive at our final state.

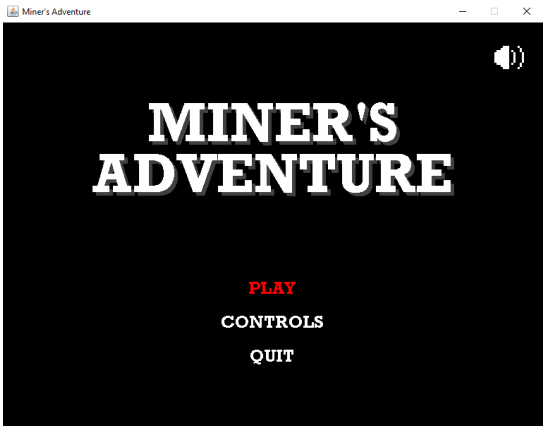
For the monster class, we deviated from our initial plan by making a single monster class as opposed to two different moving monster/static monster classes. This is because they shared all of the common functionality, with the exception of the deal damage and movement logic. Rather than clutter the project with unnecessary classes, we decided to make small adjustments in the logic for these areas. We also added a number of additional functions to the monster class so that there would be a more realistic feel to the ai. For example, we implemented the A* algorithm with an element of randomness so that the ai difficulty could be adjusted while still feeling like a realistic entity.

As collision is an integral part of our game, we created two collision classes to handle this functionality as every entity/object interacted with this logic. We also created a separate class to handle the instantiation of each entity and object. This way we could abstract the logic that was specifically only for instantiation and keep it away from our main gameConsole class so that it would not become a large blob class.

Additionally, for the objects, we changed the way they were spawned by randomizing the locations to maximize the game play experience. We also added randomization of time to the diamond class with cool time so that even if the player is unable to get the diamond the first time, they still have the opportunity to boost their score.

Finally, considering our current Tile implementation, we initially did not have any Tile implementation specified in our UML diagram. We chose to handle the generation of the maze in this manner as it seemed to be the easiest solution.

Tutorial



To play our game, first launch the game by executing the .jar file. Once you have started the application, you will be greeted with the main menu. Navigation is done with the WASD keys. If you navigate up, you will be able to select the mute button and toggle sounds by pressing enter to mute/unmute. If you navigate down, you will be able to select controls. If you press enter again, the controls for the game are displayed. Exiting back to the main menu, we can start the game by pressing enter on the play option.

Once you have done this, you will start on the default spawn point at the bottom center of the map. You can navigate around the map by pressing the WASD keys. Collect all 12 gold ingots by walking over them.

Occasionally, a diamond will randomly spawn on the map. The coordinates for the diamond spawn location will be displayed. If you are able to make it to the coordinates before the diamond despawns, you will be able to collect it and increase your score.

If you are not able to make it to the diamond in time, it will despawn and a message indicating the despawn will be displayed. Once you have collected all gold ingots, a message will display indicating that the exit is opened. Make your way to the exit which is at the center top of the map. Once you reach the exit the game will end and you will be prompted to return to the main menu.

Modifications

Since Phase 3, only minor changes have been made. There have been various updates to classes to add additional Javadoc comments to comply with the Javadoc standard. The pom.xml file was changed to include jar packaging and javadoc creation.