CMPT 276 Phase 2 Report - Group 12

Overall approach to implementing the game:

We aimed to keep our game design highly modular. We created well-defined packages for different aspects of the game and divided the concerns further within those packages. For instance, our “Actions” package has two classes which separate user interface inputs from in-game inputs. Our design aimed to have minimal coupling and reliance on other classes. This was accomplished by having a single person in charge of their package, which could be compiled and tested independently. For packages where coupling was unavoidable, we followed robust communication practices, including descriptive comments in the code and regular meetings. The midway progress goal was to have a working user interface and a completed map, with around half of the core mechanics of the game engine being completed. Due to our group misunderstanding the deadline, an extension was provided by Dr. Alimadadi. This extension allowed us to submit the final implementation of the game by March 14th.

State and justify the adjustments and modifications to the initial design of the project:

Our group has two people with extensive experience in making games; they were at the helm of our game design. Because of that, our overall design was implemented with only a few major changes. Changes to the UI package include the removal of the “Button.java class,” which was to be used in the Menu superclass as an array of buttons, which would also send state changes to the controller. Instead, JButton was used in the two classes which inherited the Menu; PauseMenu and MainMenu. This was done to make the code easier to understand and implement into the game. It also made creating future buttons more efficient as you can directly create JButton in whichever inherited class required it. In addition, the GameOverMenu was also added to the UI package to display the time and final score the user earned.

Furthermore, changes to the player HUD from our use cases include the renaming of the ESC button to pause. This was done to be consistent with most 2D tile games. The renaming of stamps to pages was done to be consistent with the Slenderman game.

The state machine also underwent some minor changes. Instead of using strings exclusively, it was modified to take enum generics. This adjustment made the project type safety and easier to refactor. The Player class was also improved by adding additional helper methods.

The map also changed. A new map generator was made, which changes the look of the map for every game. This benefits the user and extends the replayability of the game. In the design, we had unlimited levels, however, we set the cap to a maximum of 3 so the user can feel satisfied by completing the game. Another adjustment our group made was not including the vision disruption in the final implementation of the game. This was done because we felt it was beyond the scope of the requirements. Another change from our use case: bonus rewards also increase the player's score along with freezing the enemy. This was done to meet the phase 1 requirements of bonus rewards increasing the player's score.

Explain the management process of this phase and the division of roles and responsibilities:

Our group made a Discord server during Phase 1. During the design stage of this project, Cordell was made the group moderator and put in charge of integrating all classes and methods into a working game, as he had the most experience. He was also in charge of the core mechanics of the game. Sikij and Jordan were responsible for handling GameObjects consisting of Player, Enemy, map creation, implementation and interactables. Tanvir was responsible for the User Interface consisting of Menu, MainMenu, PauseMenu, PlayerHUD and GameOverMenu. To avoid merge conflicts, we committed to git regularly. We also had weekly meetings to update the group on individual progress and our goals for the coming week.

List of external Libraries you used:

No external library was used.

Describe the measures you took to enhance the quality of your code:

The quality of our code was enhanced by consistently using JavaDoc comments and appropriately naming variables and methods. Having a good UML design was necessary for writing good code. It allowed our group to visualize the process of how their classes will be integrated into the game. Another measure was subdividing tasks by creating additional methods. This removed repeated lines of code and allowed refactoring to be more seamless.

Discuss the biggest challenges you faced during this phase:

There were a few challenges. Two of our group members had not coded in Java before this class, and there was a slight learning curve for them. These same members also did not have game development experience, which was another learning curve for them. Committing to weekly meetings was sometimes difficult due to the different schedules of group members. Our group's biggest challenge came because we misinterpreted the Phase 2 deadline. We assumed that it was to act as a guideline for us, and because of that, we received a 2-day extension. To meet this deadline, we worked tirelessly for two days until the game was fully implemented.