Personal Progress in Game Programming

Prepared by Andrei Martins

Role(s) in team: Programmer, Documentation, Designer and Tester.

Short Overview of completed tasks:

- Design of the entire map, weapons, objects and characters.
- Implementation of the soundtrack.
- Supported the implementation of different chat boxes for each NPC.
- Implementation of small tasks, like health recovery mechanics and breakable rocks leading to the boss.
- Code management, documentation and refactor.

Detailed description of completed tasks:

Design of the entire map, weapons, objects and characters. - The design of the map was done using Tiled software. I used free sprites, most of them collected from Devianart. For easier management, the map was built in layers (for example, the grass floor represents a layer, the trees another one, the houses another and so on). This facilitates how the map can be read and rendered in Java.

Implementation of the soundtrack. - The soundtrack management is implemented in a class called "Sound". This class uses the "javax.sound" library, which handles how to play sound files in Java. This class contains some methods in handling the background music from the game, such as playBackgroundMusic(), for the main music game and playSound() for the sword sound when the player attacks. Another useful method was the setVolume() for handling the volume of each sound and music in the game.

Supported the implementation of different chat boxes for each NPC - Originally all the NPCs of the game were loaded in a list of type NPC. This caused some problems because it didn't allow each NPC to be addressed individually, for individual interactions with the player and quests. A solution was factorial of the code, where each NPC would be in an individual class, inherited from the NPC class. These class were: BobNpc, BrutusNpc, CatNpc, DogNpc, OlgaNpc and SteveNpc

Implementation of small tasks, like health recovery mechanics and breakable rocks leading to the boss - After the player defeats the monsters and collects their bones, the health bar should be restored. This is done by adding the hit() method in the Avatar class, when there is a collision detection between the player and the bones and calculating how many health points the player should get back. Also creating the "Rock" class, which represents the wall of rocks separating the boss area from the map. These rocks have collision detection and can take damage from the player to liberate the path to the boss.