Escape Room Testing Document

Group 1: Fade Abdeljaber, Chris Lenell, Deonvell Shed, Luke Wittenkeller

Project Overview

The escape room game has players going through a 3-d space while solving puzzles and riddles with the objective of escaping the room. The game contains several of these rooms the player must escape from. These rooms consist of walls and sprite objects which the player can interact with. Players also have an inventory and can pick up certain objects in the game environment. These picked up objects can then be used for the various puzzles in the levels.

Code Descriptions

1. The class BackgroundMusic.java is the basis for how we can use sounds and music in this project. It uses Clip, AudioInputStream, AudioSystem, and FloatControl which are all from the Java Sampled library. The AudioInputStream takes a .wav file which can create the Clip that will play the sound. The volume is controlled by the gain variable which is passed to the FloatControl to adjust the decibel level of the clip.
2. The abstract class Level.java is the base used for creating the various levels in the game. It contains helpful functions for creating levels. It has methods to add and remove sprites to the level, add wall textures to the level, and adding objects to the player's inventory. Textures are added to an ArrayList of textures objects and sprites are added to an ArrayList of sprites objects. So far, two classes implement this abstract class -- Tutorial and LevelOne.
3. The class StartMenu.java is where the start menu ui components will be stored. It uses JPanels, JButtons, and images to build the start menu. Each button has an actionlistener attached to it so the page will change depending on what's clicked.
4. The class Texture.java contains all the textures that are used in the game. These textures are used for the walls, ceilings and 2D sprites.Every texture that is loaded when a new Texture object is created has to be a 64x64 in size. Each individual textures holds the location of the original file path and file name.

Test Suite Information

1. Before each test a BackgroundMusic class object is created. The first test for BackgroundMusic ensures that the class is properly initialized by checking using the functions inherited from the java object class. The next three tests relate to the gain variable and setGain function. They all set the gain and then check if the gain variable is set properly. One test uses a value between 0-100, another uses a value less than 0, and

the last uses a value greater than 100. The gain needs to be within the range of 0-100, so we expect that the function checks for this. There is one last test for the pause function. It uses the playMusic function on a .wav file then it waits for five seconds and calls the pause function. We then check the clipTime variable which should be set to about five seconds because of the pause function.

2. For this test, a new class named TestLevel is created. This class will implement the abstract class Level and test its methods that manipulate the textures and sprites. The first test will be for the textures. It will add textures to the textures array. It will then check that these textures render properly on the walls and on the floors. The second test is for the sprites. It will add sprites to the sprites array and ensure the sprites render properly. This includes making sure the correct image is shown and multiple sprites do not interfere with each other.
3. Before each test, an instance of StartMenu.java is created. The first test will ensure that the class is initialized. From there, another test is run to ensure that each component in the start menu is initialized and properly displayed. The next two tests ensure that when the button is pressed, it will load the level properly. This is checked by checking if an instance of a level was created. If it was created, then the level was loaded correctly. When a level is selected, the start menu is removed from the main JFrame.
4.

Code Inspection and Testing Results

1. After running these tests everything worked as expected except for the setGain function. It failed the two boundary tests that test values outside the correct range. Upon the inspection of the code, setGain does not check if the value inputted is greater than 100 or less than 0.
2. After running these tests, the textures were found to render correctly on the walls, but not the floor. The floor texture did not update to follow the players position. The floor was a static image instead of moving according to the player. The sprites rendered the correct image, but were not displayed in the correct order. Farther away sprites were rendered before closer ones.
3. After running these tests, everything worked as expected except for the start menu disappearing. When a level was selected, the game was created OVER the start menu, but the start menu didn't actually disappear from the JFrame.

Conclusion

After performing all the tests, if there was any problem then as a team we would review that problem and try to find a solution. Such as the test with the floor texture being static

and not moving with the player. Each team member would try to find their own solutions and talk to each other to see which works best. Most of the solutions were either fixed this way after finishing certain tasks of the game.