Object-Oriented Software Engineering

Final Report

Group 3K

Deniz Alkışlar

Ekinsu Bozdağ

Eliz Tekcan

Serhat Aras

Selen Haysal

1. Changes in the implementation

Most significant change in our implementation is including game management classes and functionalities in our User Interface subsystem. Our design has only two subsystems but it was designed to have three: User Interface, Game Management and Game.

There are not much changes in the Game subsystem, the most notable change is that our design forced the items in the player's backpack to affect only one stats of the player however it is implemented to affect all four of the stats: health, money, sociality and grades. Also we decided that some functions in the system are redundant or misplaced, redundant ones were removed and misplaced ones were correctly placed. There are also minor changes in return types of the functions of the Game subsystem.

2. Current status of the project

2.1 Completed Parts and Bugs

The activity transaction is completed. Most of the layout designs are completed for the activities and fragments. Most of the buttons has listener and some of them is also implemented.

MemoryManager class is also implemented however there are some bugs when it is used. HouseActivity can't load values which LoginActivity saves. This bug is going to be fixed in the next iteration.

2.2 Parts Needed to be Done

Graph class is fully implemented although linking of events are not done yet since the flow of events are not determined yet. Our current game graph includes only 5 events and our graph consists of 5 unconnected events. Our aim is to improve the graph and determine the flow of events.

Game management subsystem is planned to be seperated from the User Interface classes.

The listener for the MotionSensor is implemented in order to capture the Shake motion but needed to be tested and linked to the NightClub Fragment.

The Listener for the MediaRecorder is implemented in order to capture the Decibel Level of the current sound in the given interval but the listener's interaction and Amplitude calculation must be done correctly. Therefore needs further research in order to the convert the double Amplitude value to the decibel. Also needed to be tested and linked to the Library Fragment.

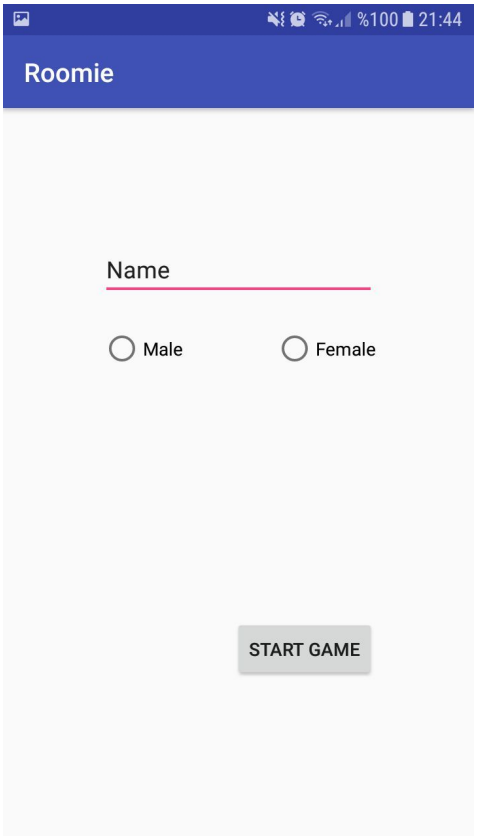
3. User's guide

3.1 Requirements and installation

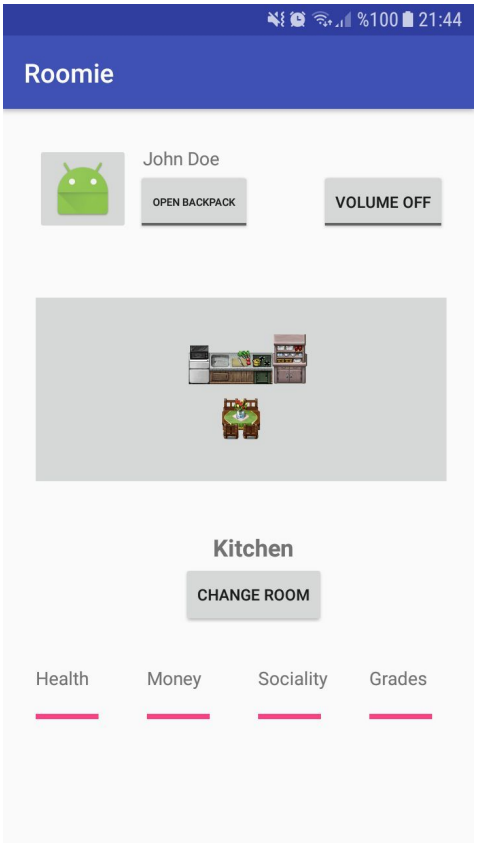
Roomie is an application which requires Android 6.0 API for installation. It is implemented in Java.

The project was implemented by using Android Studio which uses Intellij based IDE. The application demands permissions for the microphone and motion sensors. Also it uses Android's SharedPreferences class to save/load some key-value sets which uses internal storage of the device. However, Roomie requires special hardwares such as Gyro Sensor(Motion Sensor) and Microphone(MediaRecorder), these are mostly included by the manufacturers as default in most of the current phones of the Market. The application launches from the login screen and moves to the house activity where game is played. Source code of the project could be seen from master branch of the project file and the individual updates from the separate branches.

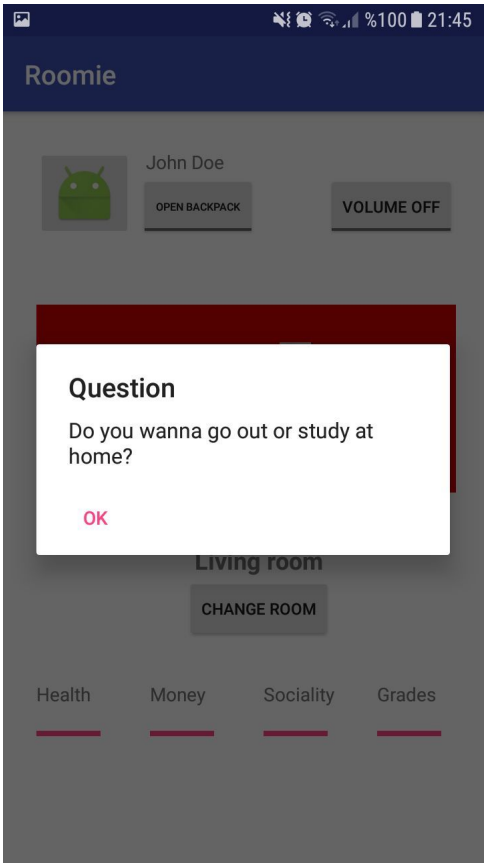
3.2 Overview of the game



- This is the opening screen of Roomie, the user is asked to create its own character by entering its name and choosing it's gender. Start game button is used to start the game.



- Android button represents the user and it is used for showing current event's question.
- Open backpack button is used to view the items in the backpack.
- Volume off button is used for muting.
- Change room allow the player to visit different rooms of the house such as bedroom, kitchen, living room etc.
- The four bars below represent the current stats of the player. In this screen shot, they are set to their maximum values. They change in the flow of the game with respect to the users choices.



- In this screen the application shows a dialog. The dialog shows current event's question. After closing the dialog, the user presses one of the item to make his/her choice. The game makes calculations based on this choice and shows the next question.