Final project report

Samuel Perez Escalante (培雷斯)
Student ID: 101062281
Department of Computer Science and Information Engineering

Taiwan, June 16, 2016

Abstract

In this document we present the report for the final project of the Software Studio Class for the spring term 2016. This class has been oriented to the *games with purpose* approach to solve problems, among the most commons are images labeling, text correction, audio recognition, etc. For this project we used the *Dance dance Revolution* game to make a fun yet challenging image labeling game with purpose.

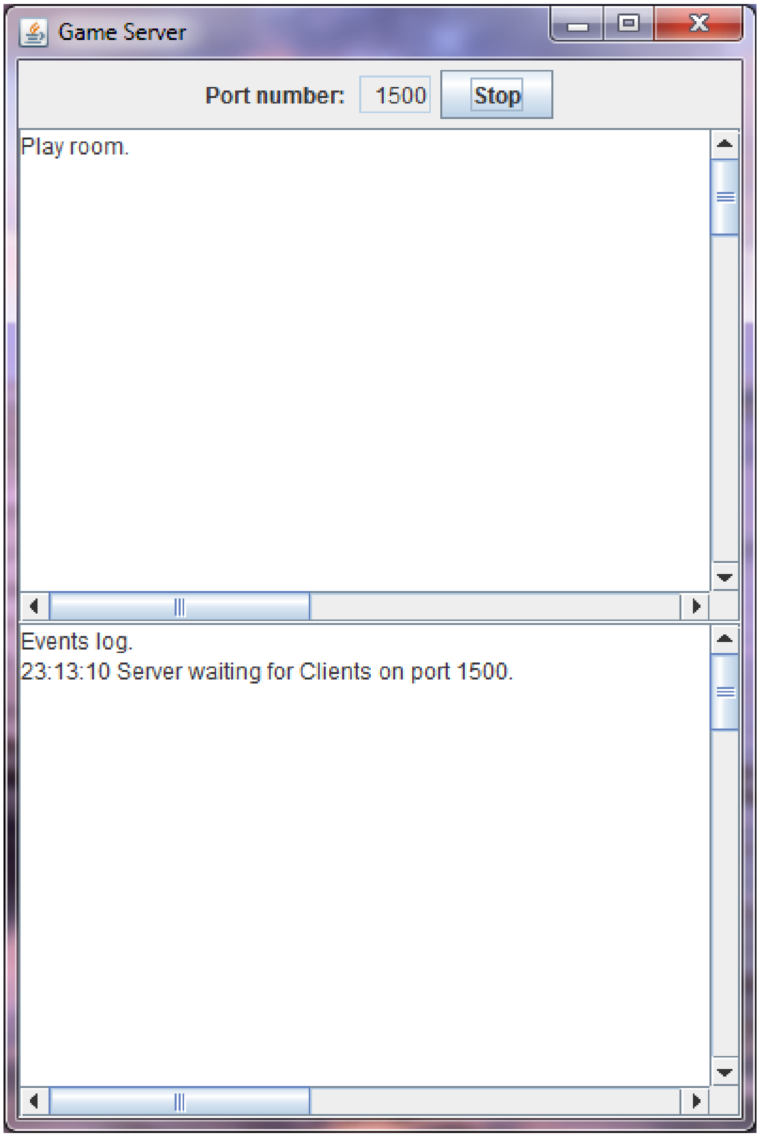
1 Introduction

This game's architecture was the traditional client / server architecture based on sockets. Each one is described as follows.

1.1 Server

The server basically is a sockets server who waits for exactly 2 players to connect in order to start the game. For our connections, we do not send raw String objects, but we serialize / deserialize a class called *ChatMessage.java* that we created that allows us to send different types of messages, such as log in signals, key pressed from each player, and the results of the image labeling processes. When the server is initialized (*ServerGUI.java*), we have 3 processes that occur in succession:

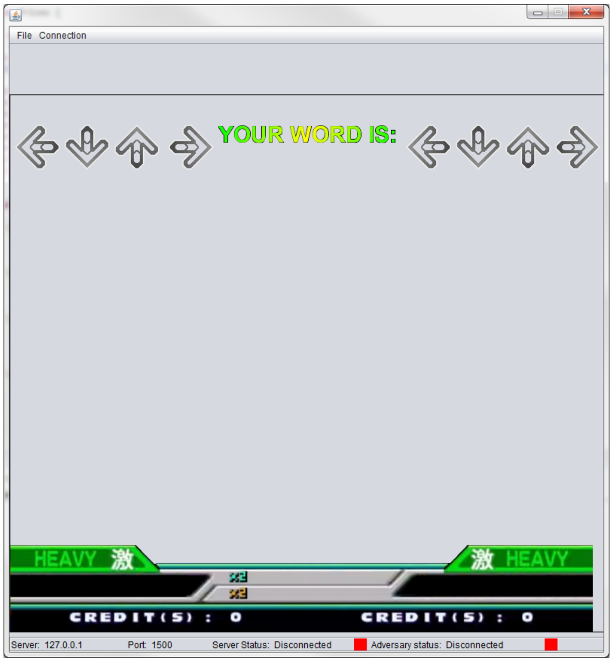
1. Input of the word to label throughout the images.



Play room.

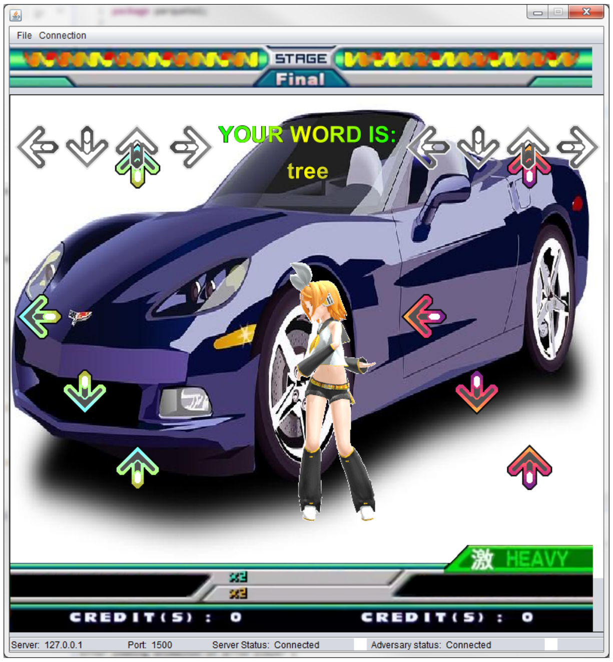
Events log.
23:13:10 Server waiting for Clients on port 1500.

Figure 1: Server waiting for players.



Server: 127.0.0.1 Port: 1500 Server Status: Disconnected Adversary status: Disconnected

Figure 2: Client waiting to connect.



Server: 127.0.0.1 Port: 1500 Server Status: Connected Adversary status: Connected

Figure 3: Gameplay with one image.

2. Input of the amount of pictures to use during the game.

3. Input of the pictures' files paths[1].

As a special note we need to mention two important facts:

1. The game starts only when there are **exactly** 2 players connected. If a third player connects to the server he is not allowed to join the actual game.

2. If one of the players closes his game or gets disconnected, the game stops because it cannot give an accurately result due to the absence of a player.

1.2 Client

For the client part (*DDRMain.java*), we also serialized the *ChatMessage.java* class in order to detect when a user pressed his keys.

For each user, the keys that they need to use are:

1. For first user:

   (a) Left: Left arrow

   (b) Up: Up arrow

   (c) Down: Down arrow

   (d) Right: Right arrow

2. For second user:

   (a) Left: Key *a*

   (b) Up: Key *w*

   (c) Down: Key *s*

   (d) Right: Key *d*

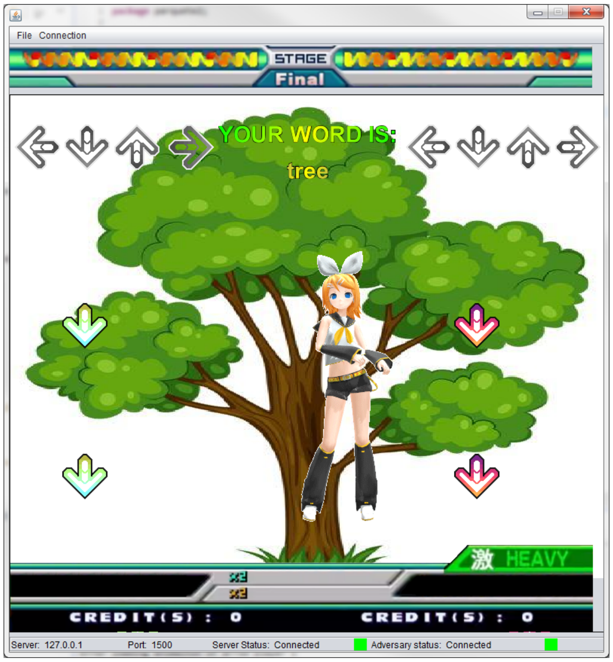
1.3 Game ending

After the game is done, it creates a txt file in the server stating which pictures correspond to the requested label in server setup.

1.4 Demo

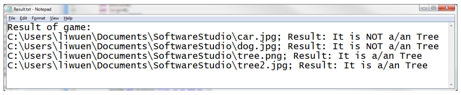
A demo of the game can be found in `https://www.youtube.com/watch?v=isWEFhtLJ4c`

[1] We get this one as many times as the amount of pictures previously input.



Server: 127.0.0.1 Port: 1500 Server Status: Connected 🟩 Adversary status: Connected 🟩

Figure 4: Gameplay with different image.



```
Result of game:
C:\Users\liwuen\Documents\SoftwareStudio\car.jpg; Result: It is NOT a/an Tree
C:\Users\liwuen\Documents\SoftwareStudio\dog.jpg; Result: It is NOT a/an Tree
C:\Users\liwuen\Documents\SoftwareStudio\tree.png; Result: It is a/an Tree
C:\Users\liwuen\Documents\SoftwareStudio\tree2.jpg; Result: It is a/an Tree
```

Figure 5: Results of the test.

2 Conclusions

For the conclusions of this report for this project, we have:

1. A different, two player, highly competitive game will be developed as an alternative solution for image labeling.

2. A client/server architecture approach was implemented for this project.

Samuel Perez, international student original from Guatemala. Born in 1991, came to Taiwan on August, 2011. Like to work with Java EE projects, have experience with Android and databases development.