Project

Networkprogramming ID1212

Diaco Uthman diaco@kth.se

8/1-2018

1 Introduction

The goal of this project was to develop a chosen idea, or task. The chosen idea is a chat application, that allows a user to enter a conversation, using his/her own chosen username.

2 Literature Study

The literature study consisted of watching the tutorial videos provided on Canvas, along with discussion with other students, in how to solve the problems that occurred. Furthermore, online browsing also helped when encountering problems using the websocket classes.

3 Method

To solve this project, NetBeans IDE was used mainly. Namely using a Maven Java Application type. This made it a bit easier to use the different dependencies that were needed. Instead of downloading them, Maven made it possible to import links instead.

Before starting the project, it was important to have an idea of the design. The videos that were given, along with looking at the source files of the provided codes, gave a good understanding of what was to be done.

Because the requirements of this project were quite unclear, a set of requirements were made up. They will be shown in the Result section of the report, and met.

4 Result

The source codes have been submitted on GitHub in the repository given in the link below:

Figure 1: https://github.com/FlyHighXX/ID1212-Networkprogramming

Link to the source codes of the application. The project will be under the Project folder

Some of the requirements will be shown and proven below

- **The client must be able to have a username when communicating in the conversation.**

To fulfil this requirement, each user will be asked to pick a username, when starting the application. Figure 1 will show this scenario in the command prompt.

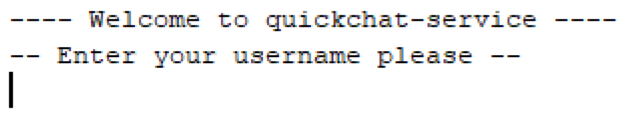

```
---- Welcome to quickchat-service ----
-- Enter your username please --
```

Figure 1: Shows when a user is asked to enter his/her username.

When the username is entered, the answer will be sent to the server and stored as a property in the client's session variable. From that point, the session will then have the username stored, and it can be accessed whenever needed. Hence, the requirement has been met.

- **The client must not store any data. All data entered by the user must be sent to the server for processing, and all data displayed to the user must be received from the server.**

All data that is used in the client view, are received by the server-side classes. To achieve this, it was important to make sure that all chat-related data was stored on the server. This could of course be done in many ways, for example in a database, or as variables in a Java class. Because of the simple nature of the chat application, no database was implemented. Also, since the conversation is not stored anywhere, but only broadcasted to all the users that are present, there was no need to save the messages. However, the username was something that needed to be saved, as well as the state of the user. The state, meaning whether the user is in the conversation or not. The username is as previously described, saved in the session variable, and not on the client. The state of the user is saved on the server-side, namely in the Conversation class. The conversation stores all users that are currently in the Conversation, and each message will be broadcasted to only this set of users.

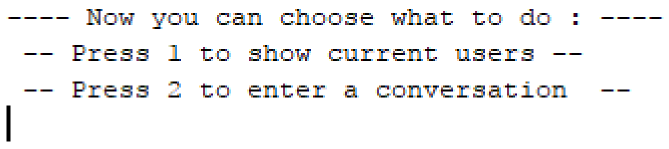
- **The communication between client and server, is provided using Websockets.**

This requirement was met, by using the javax.websocket library. Two Endpoint classes were implemented with the appropriate annotation methods, such as @onOpen, @onMessage and @onClose, that handles what happens when the Endpoint, for example receives a new message.

- **The user interface must be informative. The current state of the program must be clear to the user, and the user must understand what to do next.**

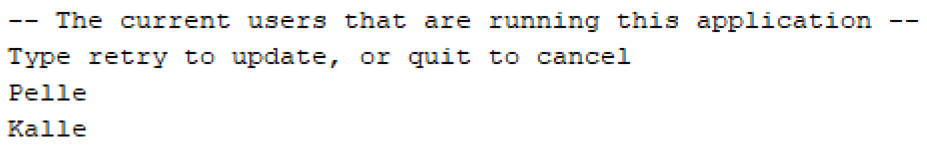
All the user interface is in the command line, and no graphical interface was implemented. If this project was to be developed further, then a graphical interface would be a much more user-friendly solution to use.

When the client starts the program, he/she will be asked to provide a username. This username will be the alias of that client, in the Conversation. Figure 1 shows this scenario. After that, the program will go into a menu, where the client can choose two options. Option one is to show all the current users, that are using the application. The second option is to enter the conversation. Figure 2, 3 and 4 show these views.



```
---- Now you can choose what to do : ----
 -- Press 1 to show current users --
 -- Press 2 to enter a conversation  --
|
```

Figure 2: The menu that is shown after the user has provided a username.



```
-- The current users that are running this application --
Type retry to update, or quit to cancel
Pelle
Kalle
```

Figure 3: The view that is shown when the user wants to see current users.

5 Discussion

The set of requirements that were set up before developing the application were checked at the end. In my opinion, all the requirements that were set up, were met. There are many ways to improve the application, and it is far from finished. For example, the application uses only the command prompt to communicate with the servers. Because of the architecture of the application, it is however easy to develop a graphical interface, that the clients can use.

A problem that occurred during the development of the application, was problems with the Websocket connection between the client and the server. The used libraries were not working properly together. Also, when using JavaFX, it was not possible to start multiple Threads, which became a problem. In the end, some tutorials online, made things a bit easier, however the JavaFX approach was abandoned. The command line was used instead as the interface of the program.

6 Comments About the Course

The project did not take too long to complete, because I decided to develop something that I already had somewhat knowledge about. Therefore, the task took about 40 hours in total, including literature studies, development and report writing.