Chat Application Report

Matvei Pavlov - Hugo Triolet

matvei.pavlov@etu.univ-grenoble-alpes.fr
hugo.triolet@etu.univ-grenoble-alpes.fr

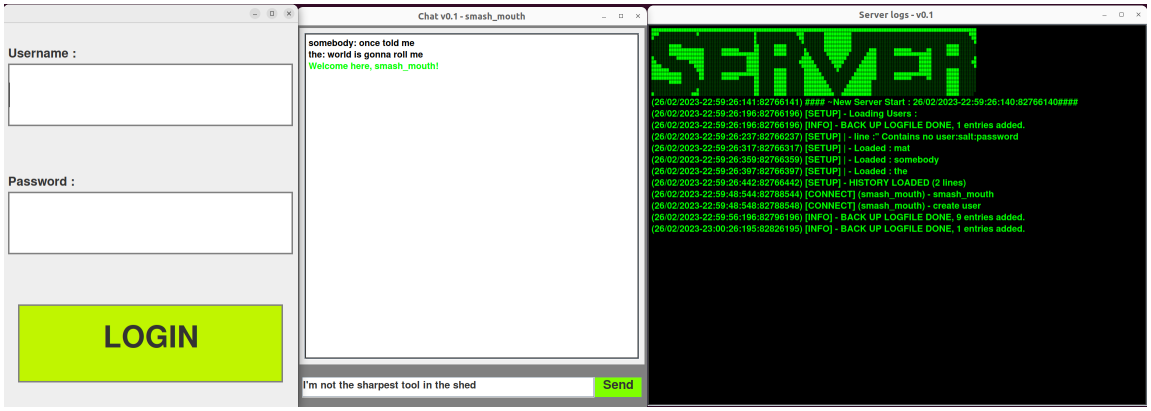
February 26, 2023

1 Introduction

The goal of this exercice was to build a chat application, where participants can dynamically join, leave, exchange messages and have a history of their conversations. Plus, the chat system is persistent (persistent history of messages). This means that if the server shuts down and restarts, it can recover the message history.
There is also a graphical user interface implemented for the chat system.

2 Architecture used for the app

For this app, it was supposed to a MVC model but as the GUI communictaes with the server, it's finally not. We implement our chat according to the architecture of an RMI-Client/Server application.



3 Organization of the code

Our code is stored in **Pavlov_Triolet_IDS/Labs_Without_Eclipse/ChatApplicationRMI**. First of all, there is a repository called **/docs** which contains all the javadoc of the projet.

The code is divided as followed :

- **exec.sh** : bash file to compile the .java and run the server (as weel as the client if stated so)
- **.classpath** : xml file generated
- **.history** : file where the history of the conversation is saved
- **.logfile** : All the log of the server
- **.project** : xml file
- **.userdata** : file where usernames and passwords associated are written encrypted
- **ChatApp.java** : GUI of a chat app client
- **ClientImpl.java** : Implementation of the client, this function is what is called when a client wishes to log in
- **ClientMessages.java** : Class used to receive messages from server and send them to the GUI

- **ClientMessagesInterface.java** : Interface of the ClientMessage, that is used by the server to send message to the client.
- **ColoredTextPane.java** : initialy found here and modified to suit our needs
- **ConnexionClient.java** : launches the connection client once the program exits, an actionlistener is triggered in ClientImpl and the execution continues
- **FileLoader.java** : loads the content of a file, makes it possible to append lines etc
- **HelloServer.java** : Server class, launching the server stub, and initializing it
- **Info_itf_impl.java** : Interface of the info class
- **Info_itf.java** : Class containing two strings, to send infos about the client easily.
- **Security.java** : Security class to hash strings in salted sha-256, for safer password storage Uses SecureRandom instead of Random
- **Server.java** : Server interface
- **ServerApp.java** : GUI of the server, to show the logs
- **ServerImpl.java** : Server Class, doing all the server skulduggery and creating links between all the users
- **Tools.java** : Tools set used around
- **UserData.java** : UserData class to manipulate easily user - salt - password

4 Functionality of the app

Basically, you can :

- launch a server with an interface
- Connect with username and password to the server
  - if your username is not in the database, registers you
  - else, if the password matches the one in the database, log you in
- talk to multiple people in the same channel
- leave the conversation
- be informed when the server stops
- retrieve the history of the conversation

5 Compile and run

The file exec.sh can be used to compile, clean and run the project. It works like a Makefile in C for example and you have access to those command :

```
bash exec.sh make // to compile only
bash exec.sh clean // to delete *.class
bash exec.sh client // to recompile and lauch a client
bash exec.sh kill_java // to kill all running instances of java. Can be used as a second
    argument too.
```

However, to run the server, we use a package called **xdotool** to automatically open the server in a new tab in the same window. So there is two cases :

If you already have xdotool or xdotool can be installed and will work normally on your machine :

```
bash exec.sh //to compile and lauch the server with xdotool
```

If xdotool doesn't work on your machine :

```
bash exec.sh noXdo // to compile and lauch the server without xdotool
```

In any case, you have to open a new terminal to create a new client that'll connect to the serveur (if it is running). To run it with localhost, you can use :

```
bash exec.sh client // to compile and lauch the server without xdotool
```

6 Git URL of the project

For further details and if you want to fork or just consult our project, see the github of the project.