

## Compiling instructions:

Open the project folder with IntelliJ or Eclipse and run the project (press on the play icon).

## How to run the client and server programs:

1. Open Cmd.
2. Navigate to the folder of the Client/Server program.
3. To run the Client program: run the command "java -jar Client.java".
4. To run the Server program: run the command "java -jar Server.java".

## Server and Client configuration files:

We used property file, which used to store the configurable parameters of an application in list of key-value pairs.

In the configuration file of the Server program we have only one pair:

key: "port", value: the port that the server listening on.

In the configuration file of the Client program we have two pairs:

key: "ip", value: the ip address that the client communicate with.

key: "port", value: the port that the client communicate with.

Each configuration file is stored in the same directory of the jar.

Server tool –


- Directory name: Server\_jar
- Tool name: Server.jar
- Configuration file name: server-config.properties

- Directory name: Client\_jar
- Tool name: Client.jar
- Configuration file name: client-config.properties

The Configuration file must be in the same location of the tool, both for server and client. For example, if you run the server tool at C:\Users\kuku, the configuration file serverconfig.properties must be in the same location (same for client). In addition, you can edit the configuration file at run time, before the connection.

## Client User interface and how it works:

Choose an option between 1 and 6:

- 1 – Connect – c • 2 – Register – r.
- 3 – Leave – l.
- 4 – Send – s.
- 5 – Disconnect – d.  – Quit – q.

To connect to a server, enter "1" **or** "connect" **or** "c", and press enter. If you are already connected you will get a message about that.

To register to a **new** topic, enter **"2" or "register" or "r"**, and press enter. After that enter the name of the topic you want to register to, and press enter. If you are already registered to this topic, you will get an error.

To leave a topic **that you are registered to**, enter **"3" or "leave" or "l"**, and press enter. After that enter the name of the topic you want to leave, and press enter. If you are **not registered** to this topic, you will get an error.

To send a message to a specific topic, enter **"4" or "send" or "s"**, and press enter. After that enter the topic you want to send a message to, and press enter. After that enter the message you want to send.

To disconnect from the server, enter **"5" or "disconnect" or "d"**. After that press enter.

To close the program, enter **"6" or "quit" or "q"**. After that press enter.

After you execute a command (except quit), you can keep executing commands until you decide to quit the program.

### Server User interface and how it works:

Choose an option between 1 and 3:

- 1 – Start.
- 2 – Stop.
- 3 – Quit.

To start listening on an IP address, enter **"1" or "start"**. After that, you will see a list of all IP addresses you can listen to (the last option is to write IP address that not on the list) and press enter. The server starts listening on the IP address.

To stop listening, enter **"2" or "stop"** and press enter. The server stops listening on the IP address. After the server stops listening, it can start over, and choose new IP address to listen to.

To close the program, enter **"3" or "quit"** and press enter.

### Location of log file:

The log file will be in current working directory. The log file name is ChatLog.log

### Thread safety strategy:

To lock access to shared resources we use the key word synchronized in all the critical sections.

In addition, for methods which must be synced, we added the keyword "synchronized" to the declaration of the method.

Furthermore, we used the Hashtable instead of HashMap because the Hashtable object guarantees synchronization and thread safety according to java documentation. This Hashtable object holds the topics of the clients, the key is the client socket and the value is a set of the client's topics. Therefore, because the table holds shared data it is important to guarantee synchronization.