**Useful links:**

* <http://www.dreamincode.net/forums/topic/259777-a-simple-chat-program-with-clientserver-gui-optional/>
* <http://www.cn-java.com/download/data/book/socket_chat.pdf>
* File Transfer
  + <http://www.dreamincode.net/forums/topic/300837-file-transfer-in-a-simple-chat-program-with-clientserver/>
  + <https://www.youtube.com/watch?v=EFQ5gAjM7Qg> (Youtube tutorial)
  + <http://www.rgagnon.com/javadetails/java-0542.html>
  + <http://stackoverflow.com/questions/9520911/java-sending-and-receiving-file-byte-over-sockets>
  + <http://stackoverflow.com/questions/4687615/how-to-achieve-transfer-file-between-client-and-server-using-java-socket> ( best one )
  + <https://gist.github.com/CarlEkerot/2693246>
  + http://stackoverflow.com/questions/16707816/ftp-client-server-model-for-file-transfer-in-java

View:

## Server option pane:

## Client option pane:



## Common chat-box:



# 

# **DOCUMENT:**

## How client connects with server?

1. Client object created => connect the first time to check if the account is already online by sending its username
2. Server sends a boolean:

* true:
  + Server sends a list of online accounts for the client to display
  + Thread in client starts
  + ServerThread for the client starts
  + Notify all the already online account so they will display the new account in their account list
* false: account already online => close input, output, socket, frame of both sides

## How to notify all the clients if an account is logged out?

1. When an account is logged out and its frame is closed, the run() function in the serverThread that contains the socket connected to this client throws an exception, which runs the function “closeAll()” making the input, output, socket of the serverThread close. The exception also runs a function called “removeConnection(...)”
2. “removeConnection(ServerThread serverThread, String username)” do:

* Remove the serverThread in the users\_list
* Remove the name of the client in server’s JFrame using removeOnline(...)
* Notify all other clients that the account is off => All other clients remove its name in JFrame using removeOnline(...)

## How to notify all the clients if the server is logged out?

1. When the server’s frame is closed, all the clients connected to it throws an exeption, which closes all their socket, input, output. When user clicks “Send”, the message will not be displayed.

## How to send an emoticon?

1. A LinkedHashMap maps a particular String with an ImageIcon

Ex: “::1”: [icon1]; “::2”: [icon2]

2) When user clicks any emoticon, a function is invoked sending a message with this format “ emoticon [username] [String\_code]” to the server. The server updates its frame and send back to all clients.

3) The updateTextView look for the ImageIcon basing on the [String\_code] in the LinkedHashMap => Using textPane’s insertIcon function to display that icon

## What happens when an account changes status?

1. It sends a normal message with format “ status [username] [status]” to the server, who sends it to all clients
2. Based on the word “status” in the message, the account knows it is the notification that sb changed status and modify the display accordingly (invisible status make the account disappear in other accounts’ list)

## What happens when an account sends block signal

1. It sends a message with format “! [sender’s username]>[receivers’ username]>>” to the server
2. The server separates the receivers from [receivers’ username] and send each receiver a message with format “ block [sender’s username]”
3. The receiver adds the sender to the ‘blocked\_by’ list. The receiver cannot get any message/emoticon from the username in ‘blocked\_by’ list
4. The name of the blocker will become invisible to the one being blocked in account list

# String code for:

* Sending **normal message** for **all**:

“[Sender’s username] [message]\n”

* Sending **normal message privately**:

“![Sender’s username]>[Receivers’ usernames]>>[message]\n”

* Sending **emoticon** for **all**:

“ emoticon [Sender’s username] [emoticon\_code]”

* Sending **emoticon privately**:

“!![sender’s username]>[receivers’ username]>>[emoticon\_code]”

* Message sent by **server** when an account log out:

“ offline [username]”

* Message sent **by** **server** when an account log in:

“ online [username]”

* Message sent to **change status**:

“ status [username] available/busy/invisible”

* Message sent to server to **transfer file** for **all:**

“ file [sender’s username]”

* Message sent to server to **transfer file** **privately:**

“! [sender’s username]>[receivers’ username]>>”

* Sending block signal

“! [sender’s username]>[receivers’ username]>> ”

TASK DIVISION:

# Utilities package:

* Create class ChatboxView extends JFrame
* Create class LoginView, RegisterView (Van)
* Create class AccountOperation (Mai)
* Emoticon (Van)
* File transfer
* Status
* Video chat
* Create chat group

# Server package:

* Create class Server (Van)
* Create class ServerThread (Van)

# Client package:

* Create class Client (Mai)