

CSC3002F Networks Assignment 2019
Socket programming project

By

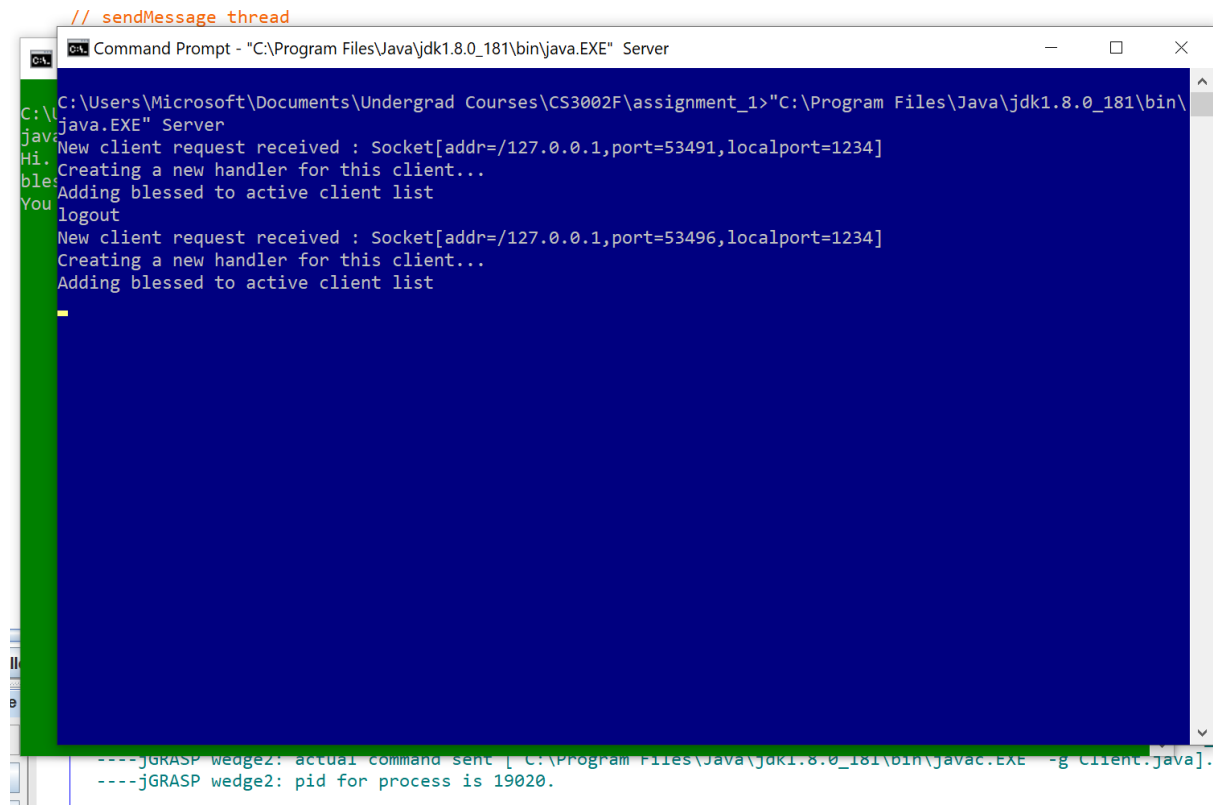
Rahul Deeljore, Philani Thwala and Blessed Chitamba

Introduction

The purpose of this assignment is to design and implement a client-server chat application using TCP sockets. The following sections in the report will show the protocol design with the help of sequence diagrams and the features of the application including screenshots of the working program.

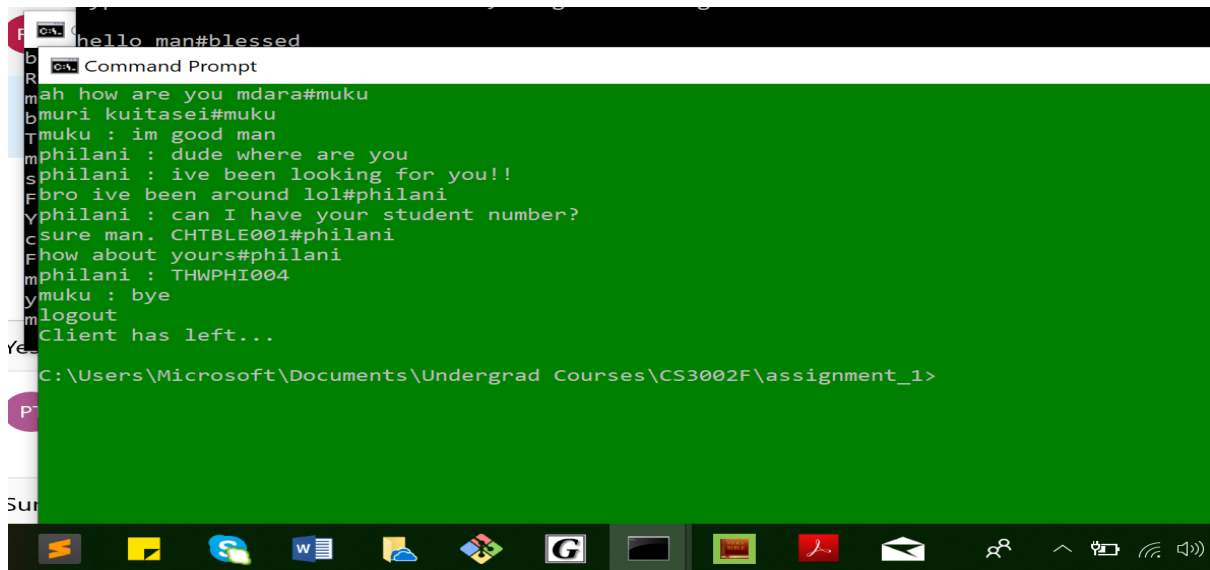
Server

Before launching the application on the client side, the server must be running. On execution, the server will notify the user that it is running. The server will then listen to connection requests and will be able to accept multiple clients. Below is a screenshot of the server accepting a client.



```
// sendMessage thread
C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Server
C:\Users\Microsoft\Documents\Undergrad Courses\CS3002F\assignment_1>"C:\Program Files\Java\jdk1.8.0_181\bin\
java.EXE" Server
New client request received : Socket[addr=/127.0.0.1,port=53491,localport=1234]
Hi. Creating a new handler for this client...
ble: Adding blessed to active client list
You logout
New client request received : Socket[addr=/127.0.0.1,port=53496,localport=1234]
Creating a new handler for this client...
Adding blessed to active client list
-
----jGRASP wedge2: actual command sent [ C:\Program Files\Java\jdk1.8.0_181\bin\javac.EXE -g Client.java].
----jGRASP wedge2: pid for process is 19020.
```

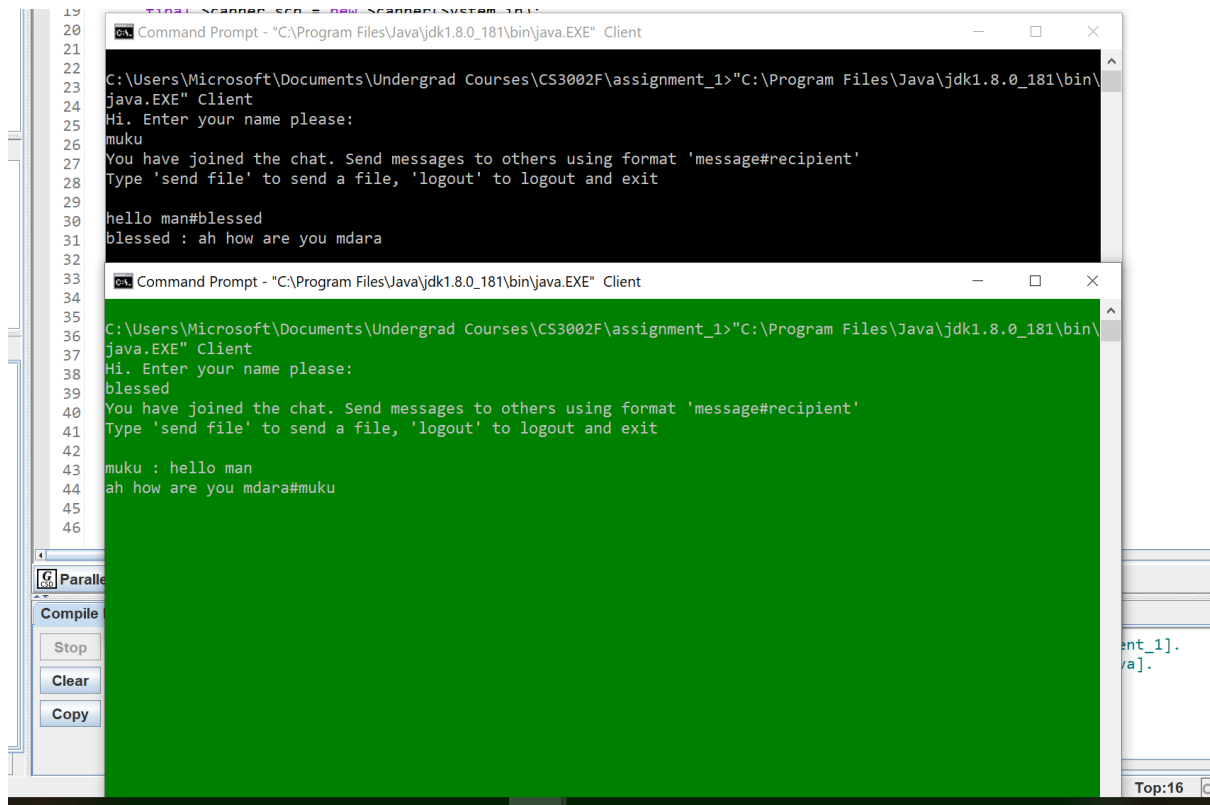
The screenshot below shows the server when a client has disconnected.



Application features

Username

On connection with the server, the client will be prompted to enter a username which will be used as an identifier for that particular client. The username will be stored in a database and no two clients can have an identical username. The screenshot below shows 2 clients joining the chat using 2 different terminals.

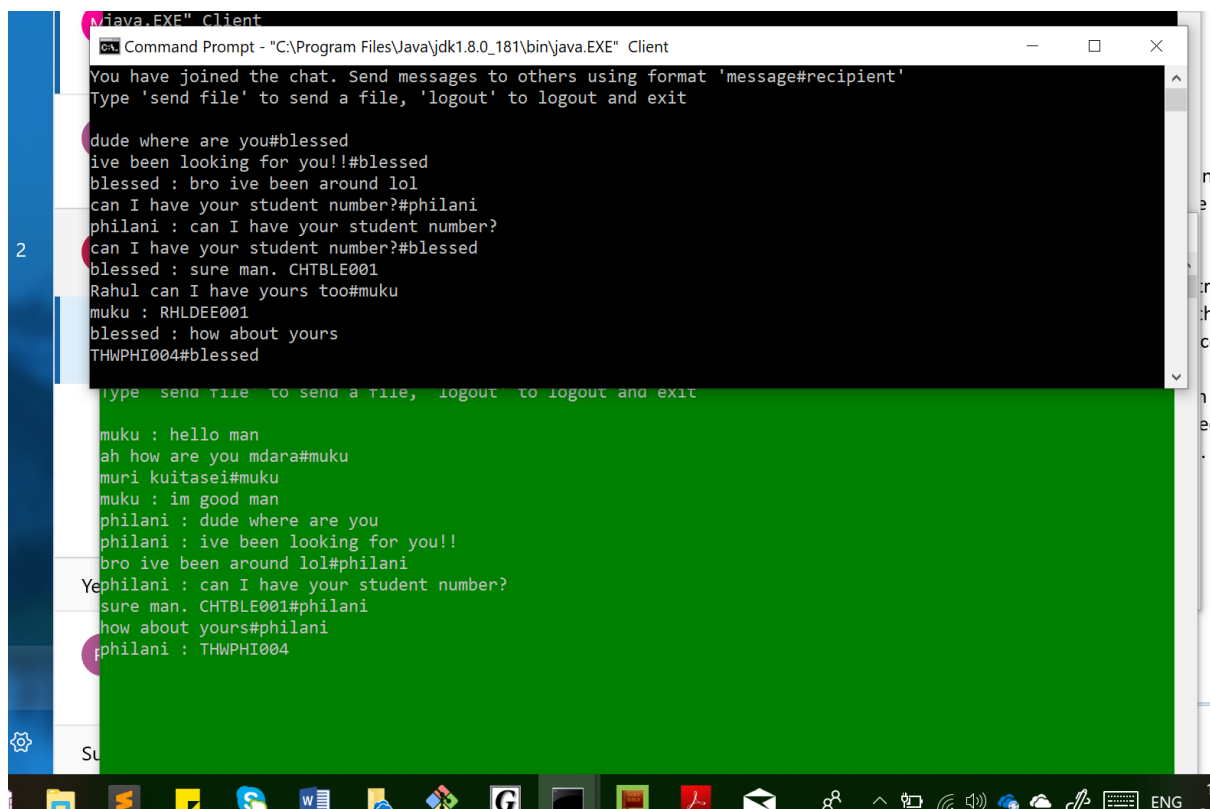


Message Transfer

The basic goal of this application is to enable client-client message transfer via a server. The message is then shown on the server and in the recipient's terminal in real time.

In the case when multiple clients are connected, a client will chat privately to another client in the format "message#recipient_name" where message is the message the client(sender) wants to send, and recipient_name is the name of the intended destination(recipient) client.

Below is shown a screenshot of the 3 of us sharing our student numbers using the chat.



```
Java EXE" Client
Command Prompt - "C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Client
You have joined the chat. Send messages to others using format 'message#recipient'
Type 'send file' to send a file, 'logout' to logout and exit

dude where are you#blessed
ive been looking for you!!#blessed
blessed : bro ive been around lol
can I have your student number?#philani
philani : can I have your student number?
can I have your student number?#blessed
blessed : sure man. CHTBLE001
Rahul can I have yours too#muku
muku : RHLDEE001
blessed : how about yours
THWPHI004#blessed

Type 'send file' to send a file, 'logout' to logout and exit

muku : hello man
ah how are you mdara#muku
muri kuitasei#muku
muku : im good man
philani : dude where are you
philani : ive been looking for you!!
bro ive been around lol#philani
Yes philani : can I have your student number?
sure man. CHTBLE001#philani
how about yours#philani
philani : THWPHI004
```

File Transfer

A Client will have the option to send local files to another client, with this process made possible by the server. This feature is susceptible to bandwidth

and connectivity limitations, therefore the client receiving the file will have the choice to accept or receive the file before it is sent.

When a client wants to send a file, they type "send file", they then receive an input prompt from the server to "Enter the name of the recipient". After entering the recipient's name, the server searches for the recipient client in the list containing client names, if it exists, it then sends a message to the recipient client "File coming, do you want to receive it? Y-yes, N-no". If the recipient types Y, the server then sends the file to the client in the recipient socket and writes "file saved in your directory" afterwards. The process of sending a file is shown in the screenshots below.

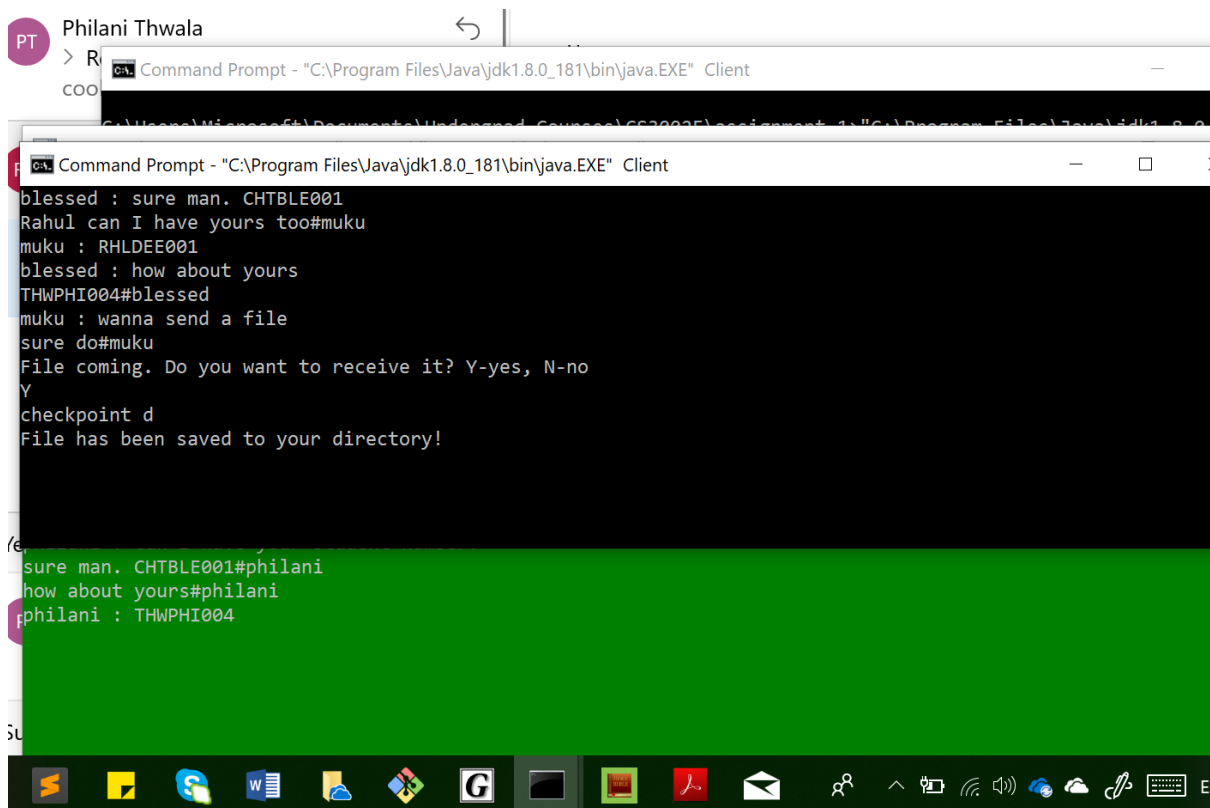
```
Command Prompt - "C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Client
C:\Users\Microsoft\Documents\Undergrad Courses\CS3002F\assignment_1>"C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Client
Hi, Enter your name please:
muku
You have joined the chat. Send messages to others using format 'message#recipient'
Type 'send file' to send a file, 'logout' to logout and exit
bhello man#blessed
Cblessed : ah how are you mdara
Pblessed : muri kuitasei
Cim good man#blessed
Bphilani : Rahul can I have yours too
RHLDEE001#philani
Wanna send a file#philani
Cphilani : sure do
Tsend file
MType name of receipient:
Sphilani
FEnter file name:
Rrubbish.txt

Yes
sure man. CHTBLE001#philani
how about yours#philani

Philani Thwala
> R
cool

Command Prompt - "C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Client
C:\Users\Microsoft\Documents\Undergrad Courses\CS3002F\assignment_1>"C:\Program Files\Java\jdk1.8.0_181\bin\java.EXE" Client
dude where are you#blessed
ive been looking for you!!#blessed
blessed : bro ive been around lol
can I have your student number?#philani
philani : can I have your student number?
can I have your student number?#blessed
blessed : sure man. CHTBLE001
Rahul can I have yours too#muku
muku : RHLDEE001
blessed : how about yours
THWPFI004#blessed
muku : wanna send a file
sure do#muku
File coming. Do you want to receive it? Y-yes, N-no
Y
sure man. CHTBLE001#philani
how about yours#philani
philani : THWPFI004

Su
```



Protocol design and specification

Username

The process for checking the username of a client attempting to enter the chat is shown below in Fig.1. The user enters his chosen username and the server checks for its availability. If the username is available, the client is connected to the chat, if not the client is told that his username is already in use. This process occurs every time a new user attempts to join the chat.

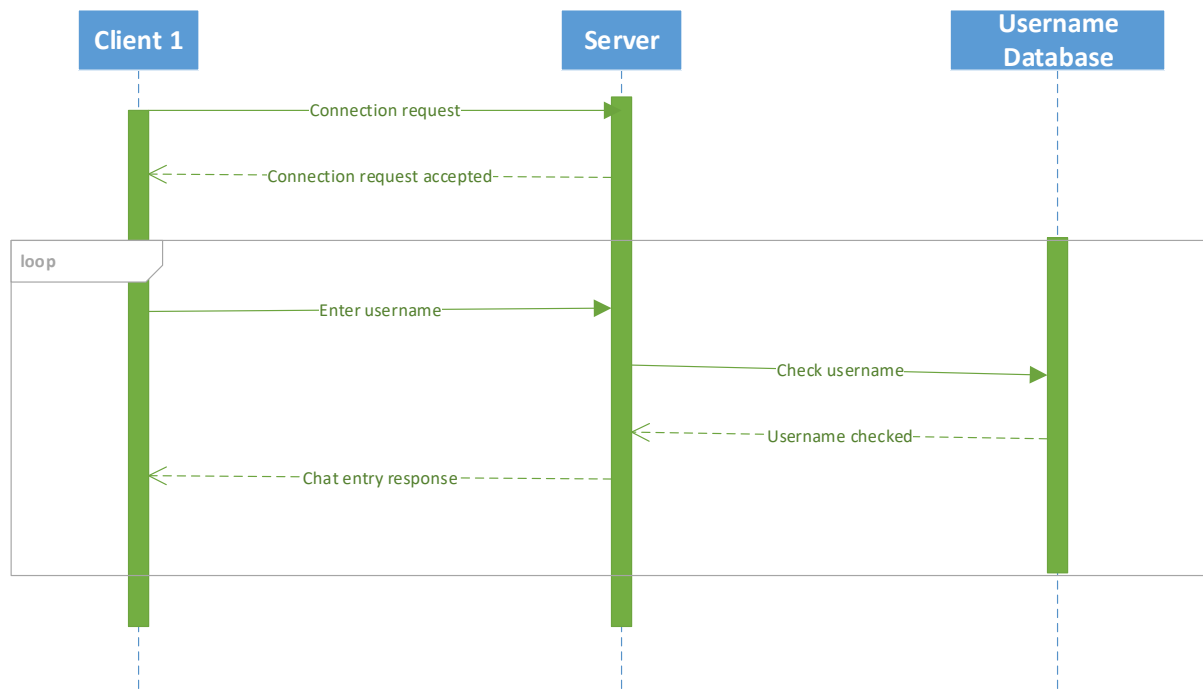


Fig.1

Message Transfer

The sequence of events when sending a text message are depicted using a sequence diagram.

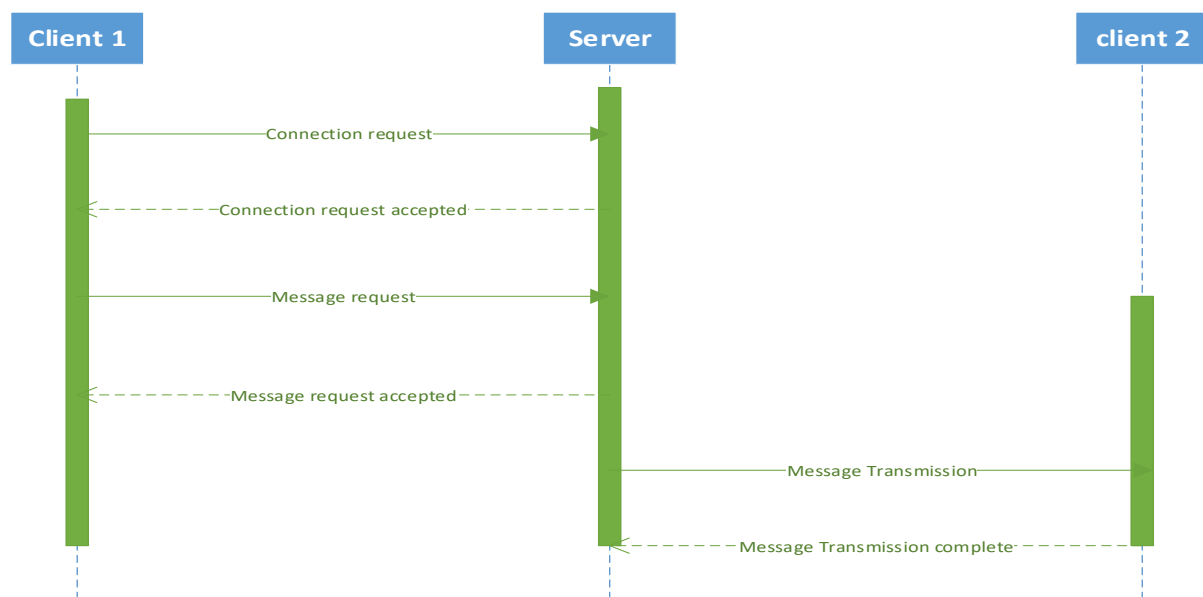


Fig. 2 Connection of a client and message transfer

We can see on Fig. 2 that the user needs to connect with the server before sending the message. In the above case, “Client 2” is already connected and so after “Client 1” has connected to the server, he can send the message.

File Transfer

The client can send a file to another client via the server. The receiving client has to accept the file download to initiate the file transfer. After accepting the file download, the file is decomposed into bytecode and sent to the client from the server. When it reaches the client, it is reconstructed into the original file.

If the receiving client chooses to refuse the file download, the file transfer is cancelled by the server and the sending client is notified.

The file transfer process is depicted below in Fig.3 and Fig.4.

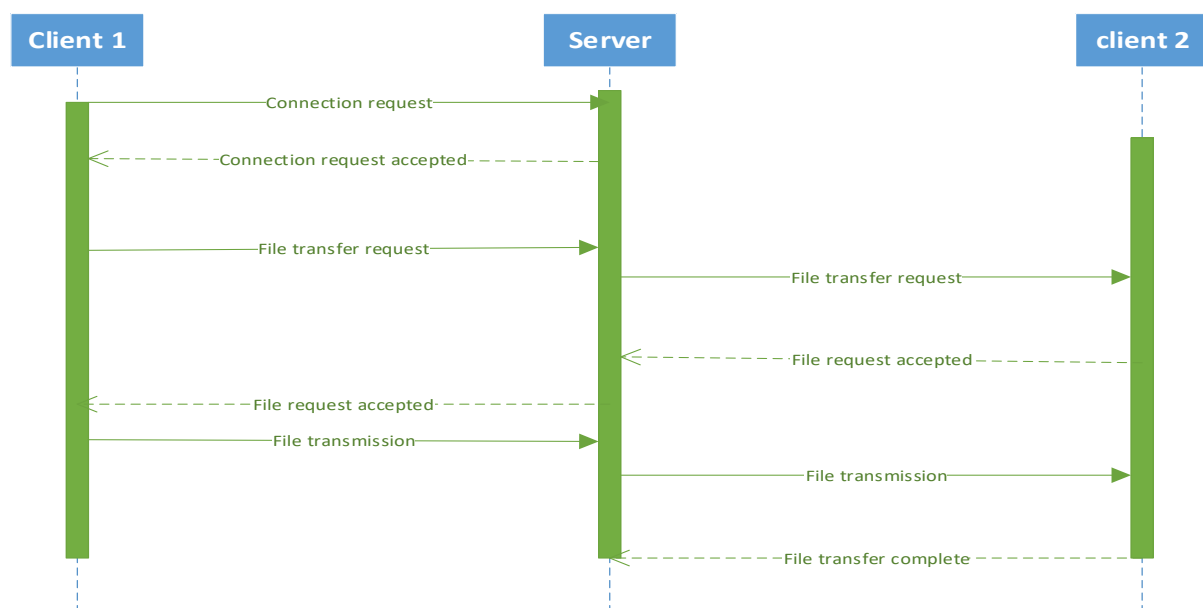


Fig.3 Successful file transfer

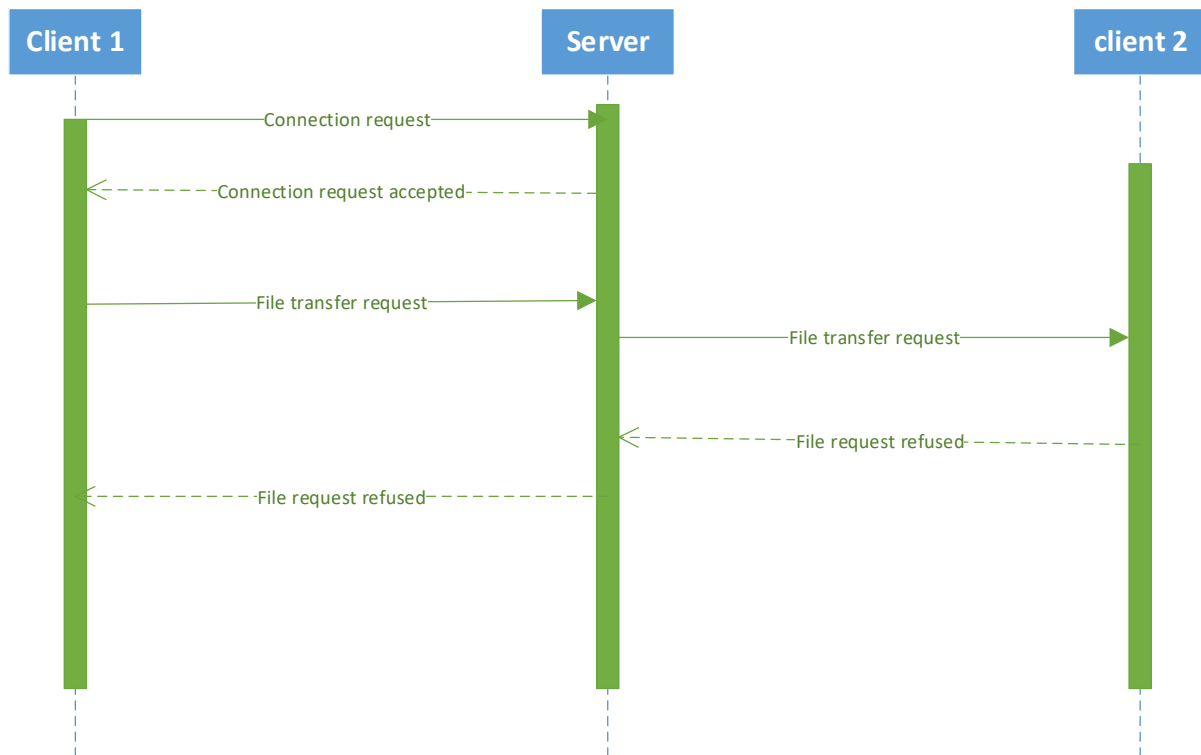


Fig.4 Unsuccessful file transfer

Implementation details

Client

Here we have separated the process of sending a message and receiving a message into 2 separate threads using 2 inner classes. 2 different threads are used to send and receive a message so that they can happen independent of each other.

Server

Along with the main thread, there is a clienthandler class that handles communication between each client and the server. Whenever a client is received and a socket is assigned to this client, a new clienthandler instance is created and runs on a separate thread to handle the processes associated with that client.