

# 컴퓨터네트워크 Assignment #2

## TCP Chattingroom

컴퓨터소프트웨어학부 2020068695 구범모

### 프로그램 설계 및 구조

과제의 명세조건에 맞게, client는 메시지를 같은 client에게 direct로 보내는 것이 아닌, 중간에 server를 거쳐서, server가 같은 채팅방에 속하는 client들에게만 메시지를 전달하도록 하였습니다. 자세한 설명은 각 class를 설명하며 이어가겠습니다.

### Class 설명

#### Client

CLI에서 client의 역할을 수행하는, main method를 가지는 class입니다. cmd에서 IP Address, port#1, port#2를 입력받아, port#1로 메시지를 송수신하는 clientsocket, port#2로 파일을 송수신하는 clientfilesocket을 생성하여 각각 server에서 accept될 수 있게 합니다.

또한, server에 메시지를 송신함과 동시에, 다른 client로부터의 메시지를 수신하기 위한 CListeningThread, CWritingThread를 생성해 주었으며, 처음에는 CListeningThread에서 clientsocket과 clientfilesocket으로 메시지와 파일 둘 다 받을 수 있게 해주려 했으나, 메시지만 수신할 때는 file을 안 받아주어 blocking이 발생하고, 반대로 파일만 수신할 때는 메시지를 안 받아주어 blocking이 발생하여, 따로 file만 전용으로 받아주는 CFileThread를 하나 더 만들어 주어, 총 3개의 thread로 메시지 송수신과 파일송수신을 할 수 있게 하였습니다.

(파일 송신은 CWritingThread내에서 처리하도록 구현하였습니다.)

또한 추후에 쓰일 #STATUS나 통신관련하여 편리하게 사용할 수 있도록 멤버변수로 유저이름을 저장하는 userName, 유저가 속해있는 채팅방인 chatRoom, 메시지,파일 송수신용으로 각각 clientSocket, clientfileSocket을 멤버변수로 설정했습니다.

Client method

setUserName(String userName) : userName을 setting해주는 setter method입니다.

setChatRoom(ChatRoom chatRoom) : chatRoom을 setting해주는 setter method입니다.

## CWritingThread

Client 입장에서 메시지와 파일을 동시에 송신할 수 있는 thread입니다. 메시지 송신 + #CREATE, #JOIN명령어를 보낼 때 이용하는 clientsocket, 파일송신에 이용하는 clientfileSocket을 생성자의 매개변수, 멤버변수로 설정해주었습니다.

일반적인 메시지 + #CREATE 혹은 #JOIN일 경우, 해당 client의 clientSocket을 이용하여 DataOutputStream객체를 이용해 server에 메시지를 송신하여 server측에서 그에 맞는 기능을 하도록 했으며(61 line.), #PUT일 경우만, 해당 클래스와 같은 디렉토리 내에 있는 파일을 server로 전송하도록 구현하였습니다. 또한, “#”은 명령어 전달에만 쓰이므로, 명령어가 아닌 기본 메시지이지만, “#”로 시작하는 메시지는 서버에 전달하지 않도록 구현했습니다.(74 line.)

세부적인 내용으로는, #PUT 명령어를 입력받을 경우, 크기가 64000바이트인 바이트 배열인 buffer를 생성해 주어 FileInputStream 객체로 해당 파일을 읽어주어, 64K바이트짜리 buffer로 몇번 송신해야 할지 while문으로 검사하여, 송신회수 측정이 끝나고, 해당 횟수를 data변수에 저장하여 server로 해당 명령문(#PUT filename), data변수, file의 크기를 순서대로 전송하려 했으나, 서버측에서는 읽는데 문제가 없었지만, client측에서 파일을 송신할 때 64000바이트 단위로 끊어서 전송할 때 파일을 읽어주는 read method에서 문제가 있어, 부득이하게 client측에서 파일을 송신할 때는 해당 파일을 readAllBytes method로 한번에 읽어서 buffer에 저장한 이후, 해당 buffer에서 파일을 data변만큼 읽어들이어 outputStream으로 전송토록 하였습니다.

이 때, file은 보통 64k단위로 전송하지만, 파일을 전송하다가 마지막에 전송할 때는 64k 미만으로 전송하는 경우가 있기에, 마지막 부분을 예외처리하기 위해서 해당 file의 실제 크기를 저장해 주는 filesize변수를 만들어 주어, 64k씩 파일을 전송하는 동시에 filesize에서 64k씩 빼주고, 해당 filesize가 64k 미만이라면 그만큼만 전송해 주어야 하기에 예외처리로 filesize만큼만 전송토록 하였습니다.(53~58 line.)

마지막으로 과제 요구사항인, 64K byte당 “#” 한개씩을 client화면에 출력하였으며, 파일 송신이 완료되면 추가적으로 file크기까지 출력되도록 구현하였습니다.(70 line.)

## CListeningThread

Client 입장에서 순수히 메시지 수신에 이용되는 thread입니다. 메시지만 수신하기 때문에 멤버변수와 생성자의 매개변수에는 clientSocket만 설정해 주었습니다. 단순히 server에서 도착한 메시지를 clientSocket의 inputStream으로 받아주어, client의 화면에 출력해 주기 위해 BufferedReader 클래스로 읽어주었습니다. 또한 server에서 따로 메시지를 보내지 않을 때, client에게 null메시지가 도착하여, null메시지 출력을 막아주기 위해 if문으로 예외처리 해주었습니다. (19 line.)

## Server

#PUT으로 전송된 파일을 저장해 두는 fileList를 static 변수로 가지고, cmd창에서 입력받은 port#1, port#2를 각각 port로 갖는 welcomesocket, filesocket을 만들어 주어, thread내에서 해당 serversocket으로 accept할 수 있도록 WelcomeThread를 생성해주어 매개변수로 넣어주었습니다.

## WelcomeThread

Server 입장에서 매개변수로 받은 welcomesocket(메시지 전용 socket), filesocket(file송수신전용 socket)으로 각각 client측에서 생성된 socket을 accept만 전용으로 하는 thread입니다.

해당 thread에선 아직 client에게서 #CREATE, #JOIN 명령어를 받지 않으므로, 먼저 client 생성자에, accept된 clientSocket과 clientfileSocket만을 생성자의 매개변수로 넣어줍니다. (24 line.)

client 객체 생성이 완료되면, 이제 메시지 송수신 + 파일 송신전용 thread인 ClientThread에 client객체를 매개변수로 넣어주고, 파일 송신 전용 thread인 FileThread에 fileSocket만을 매개변수로 넣어줍니다.

(처음에는 ClientThread에서 메시지 송수신, 파일송수신을 동시에 해주도록 하려 했으나, CListeningThread에서와 비슷한 blocking 이유로, 파일 수신은 따로 추후에 설명 예정인 FileThread에서 가능토록 구현하였습니다.)

(ClientThread에선, 일부러 해당 client의 userName과, 속해 있는 채팅방인 chatRoom을 쓸 수 있도록, client객체를 매개변수로 넣어주었습니다.)

## ClientThread

Server 입장에서 매개변수로 들어온 client객체의 clientSocket, clientfileSocket을 이용해 메세지 송수신, 파일 송신을 해주는 thread입니다.

먼저, 메세지들은 client객체의 clientSocket의 inputStream으로 수신될 것이므로, inputStream으로 들어온 메세지를 BufferedReader 객체를 이용해 메세지를 읽어줍니다. 읽은 메세지를 parsing하여, parsing된 문자열 별로 분기를 나누어, 해당 문자열이 명령어라면, 그에 맞는 명령을 수행하도록 하였습니다. 분기별로 간단히 설명드리겠습니다.

#EXIT : 해당 client의 프로그램을 종료하는 것이 아닌, client가 속한 채팅방을 '나가기만'합니다. 해당 chattingroom의 userlist에서, 해당 client객체를 remove 하고, client의 멤버번호인 chattingroom을 null로 바꿔줍니다. 이후에 cmd창에서 다른 명령어를 이어서 입력할 수 있습니다.

#CREATE : 이어서 수신하는 chattingroom의 이름, user이름을 parsing합니다. chattingroom을 만들기 전, 해당 이름과 겹치는 이름을 가진 채팅방이 있는지 검사하고(31 line.), 중복되는 채팅방이 있다면 해당 client에게 실패 메세지와 다시 입력을 요구하는 메세지를 전송합니다.(33 line.) 중복되는 채팅방이 없다면, parsing된 채팅방 이름으로 하나의 chattingroom 객체를 만들고, 해당 room의 userlist에 client를 추가해주며, client의 username과 chattingroom도 setter method로 설정해줍니다. 동시에 해당 client에게 채팅방 생성 성공 메세지를 보냅니다.(36 ~ 43line.)

#JOIN : #CREATE와 같이, parsing을 진행해 주고, 해당 room의 이름을 갖는 채팅방이 존재하는지 검사합니다. 존재하지 않다면, client에게 실패 메세지를 전송합니다.(53 line.) 존재한다면, 해당 room의 userlist에 client를 추가하고, setter method로 client의 멤버번호도 업데이트 해줍니다. 최종적으로 client에게 채팅방 참가 성공 메세지를 보냅니다.(56 ~ 61 line.)

#STATUS : 해당 client가 들어가 있는 채팅방의 정보를 보내는 분기입니다. 먼저 room 이름을 String type의 info 변수에 저장해 주고, 해당 room의 userlist를 순회하면서, user들의 이름을 info변수에 추가적으로 저장해줍니다. 순회가 끝나면, 해당 info를 client에게 보내줍니다.

#GET : 이어서 수신하는 fileName을 parsing하여, 미리 client들에게 받아냈던 file들을 저장하는, server class의 static arraylist인 fileList에서 순회하며 같은 이름을 갖는 file을 찾습니다.(찾는 file이 없는 예외는 과제 요구사항에 없어 따로 처리하지 않았습니다.) file을 찾은 후에는, CWritingThread에서 file을 보낼 때와 똑같이, buffer로 몇번 읽어야 하는지 먼저 측정하여 data변수에 저장하고,(108~110 line.) 먼저 client에게 filename, data, 보내는 file의 크기를 전송 후, data횟수만큼 해당 file을 buffer로 읽어들이 DataOutputStream객체로 해당 file을 보내주었습니다.(112 ~ 121 line.) 추가적으로, 파일 송수신에는 port#2에 해당하는 socket으로 file을 보내주어야 하므로, clientfileSocket으로 파일을 전송하도록 하였습니다.

위의 명령분기가 아닌, 일반 메시지일 경우에는, 해당 client가 속해 있는 chattingroom의 userList를 참조하여 반복문으로 한명마다 socket의 outputStream으로, printWriter 클래스의 객체를 이용해서, 보내는 client의 이름을 붙여서 최종적으로 메시지를 전송하도록 하였습니다.(125 ~ 133 line.)

## CFileThread

client 입장에서 #GET으로 요청한 file을 받는 thread입니다. (#GET으로 받은 file임을 구분하기 용이하게, client가 받으려는 file이름의 앞에 "copy"문자열을 붙여주었습니다. 14 line.)

Server에서 보내준, buffer로 읽어들여야 하는 횟수인 data변수, file크기를 저장해둔 filesize를 각각 수신해 주고, data횟수만큼 해당 file을 buffer 배열로 읽어줍니다. 또한 buffer로 한번 읽을 때 마다 64K Byte일 것이므로, 그때마다 "#"을 출력하도록 하였습니다. 최종적으로 file을 다 수신하면 file size와 함께 성공 메시지를 출력하도록 하였습니다.(35 line.)

## FileThread

client가 서버에 file을 전송하는 명령인, #PUT인 경우에만 서버에서 file을 수신하는 thread입니다.(blocking을 막기 위한 thread입니다.)

CFileThread에서와 같은 방법으로 file을 수신해주고, 다른 client들이 해당 file을 #GET명령어로 받아갈 수 있게 따로 선언해둔 Server class의 fileList에 해당 file을 add해서 저장해 두었습니다.

#PUT에 성공하면, server의 화면에 성공 메시지를 출력하도록 하였습니다.(42 line.)

## CLists

모든 채팅방을 저장해두는 static ArrayList인 chatRoomList와, String type의 채팅방 이름을 매개변수로 갖는 findRoom method로 이루어져 있습니다. findRoom method는, 해당 채팅방을 찾으면 성공 메시지와 함께 해당 room을 return해주고, 못찾는다면 실패 메시지와 함께 null을 return합니다.

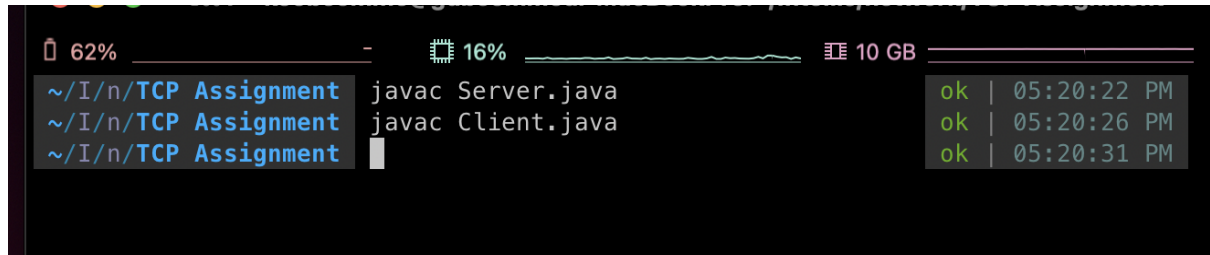
## ChatRoom

채팅방 이름인 chatRoomName, client객체를 저장해두는 userList를 멤버변수로 갖는 클래스입니다.

생성자로는 채팅방 이름만 매개변수로 가지며, #CREATE 혹은 #JOIN명령어가 있을 때 마다 addUser method로 해당 client를 채팅방의 userList에 추가해줍니다. (quitroom method의 기능은 위의 #EXIT분기 설명에 쓰여 있습니다.)

마지막으로, 과제 명세에 따로 프로그램 종료 명령어는 요구사항에 있지 않아, 따로 구현하지 않았습니다. 프로그램 종료는 command line에서 control + c 키로 종료해 주시면 됩니다. 또한, file 경로에 관해서도 나와있지 않아, 같은 디렉토리 내의 file을 송신한다는 것을 가정하고 코드를 구현하였습니다.

## 프로그램 실행방법



```
~ / 62% - 16% 10 GB
~/I/n/TCP Assignment javac Server.java ok | 05:20:22 PM
~/I/n/TCP Assignment javac Client.java ok | 05:20:26 PM
~/I/n/TCP Assignment ok | 05:20:31 PM
```

해당 파일들이 위치한 디렉토리에서 Server, Client java 파일을 컴파일하여 줍니다.(다른 java파일들도 같은 디렉토리 내에 위치해 있어야 합니다.)

## 프로그램 실행화면

