Computer Networks: Programming Assignment #1

디자인 및 구현 보고서

Computer software

2020001658 이유민

1. 프로그램 설계 및 구조의 기술

(1) Class

- p2pChat: main함수가 포함되어 있는 메인 클래스로, 전체적인 프로그램의 실행을 다루고, 프로그램 실행을 위한 함수가 정의되어 있다.
- Read: Thread 생성을 위한 하나의 클래스로, 채팅방에 입장했을 때, 전송되어 오는 메시지들을 read하는 역할을 할 것이다.
- Write: Thread 생성을 위한 하나의 클래스로, 채팅방에 입장했을 때, 메시지를 계속해서 write하는 역할을 할 것이다.

(2) Class1: Read

- Runnable interface를 implements한다.
- 포트번호를 저장할 portNum, 입장한 채팅룸의 IP주소를 저장할 IpAddress, 사용자의 이름을 저장할 UserName을 가지고 있다.
- Runnable 인터페이스를 구현한 객체로 만들었기에, run() 메소드를 오버라이딩한다.

<run()>

- MulticastSocket으로 receive할 소켓(multicastSocket)을 생성한다.
- 생성한 소켓으로 채팅방에 joinGroup한다.
- 해당 채팅방에서 오는 메시지를 계속해서 받기 위해 while문으로 들어간다.
- 채팅 메시지를 chunk 단위(512 byte)로 나누어서 받기 위해 byte배열의 readcontent를 만든다.
- DatagramPacket으로 받을 메시지의 패킷(datagramPacket) 만든다.
- `multicastSocket.receive(datagramPacket);`
  : receive msg
- 화면에 받은 메시지 출력한다.

(3) Class2: Write

- Runnable interface를 implements한다.
- 포트번호를 저장할 portNum, 입장한 채팅룸의 IP주소를 저장할 IpAddress, 사용자의 이름을 저장할 UserName을 가지고 있다.
- Runnable 인터페이스를 구현한 객체로 만들었기에, run() 메소드를 오버라이딩한다.

<run()>

- while문에서 계속 메시지 write하기
- 만약 사용자가 입력한 String이 "#EXIT"이거나 "#"으로 시작하는 경우, 이는 메시지가 될 수 없다.
- "#EXIT"라면, 사용자가 채팅방을 나가고 싶다는 의미이므로, 사용자가 나갔다는 메시지를 출력한다.(ex. Yumin is out.)

  -> DatagramPacket으로 send할 패킷(datagramPacket)을 만든다. <- 파라미터로 메시지내용, 메시지의 길이, 보낼주소, 포트번호

  -> MulticastSocket으로 send할 소켓(multicastSocket)을 생성한다.

  -> 생성한 소켓으로 채팅방 join한다.

```
-> multicastSocket.send(datagramPacket);
```

  :send msg

  ➔ 소켓 close하고 시스템 종료하기.

- "#"으로 시작한다면, 메시지가 될 수 없기에 "# is only command"를 출력하고 재입력을 기다린다.(while문 처음으로 돌아간다.)

- 둘 다 아니라면(올바른 Msg), 사용자가 입력한 메시지와 사용자의 이름을 함께 출력한다.

  ➔ DatagramPacket으로 send할 패킷(datagramPacket)을 만든다. <- 파라미터로 메시지내용, 메시지의길이, 보낼주소, 포트번호

  ➔ MulticastSocket으로 send할 소켓(multicastSocket)을 생성한다.

  ➔ 생성한 소켓으로 채팅방 join한다.

```
➔ multicastSocket.send(datagramPacket);
```

  :send msg

(4) Clase3: p2pChat

<main>

```
-   String portN = args[0]; //multicast port number
    portNo = Integer.parseInt(portN);
```

: main - 프로그램 실행과 동시에 사용자에게 입력받은 포트넘버를 portNo에 넣어둔다.

- while문을 통해 계속해서 프로그램이 입력값을 받을 수 있도록 한다.

```
-   System.out.println("P2P Chatting room start!");
    System.out.println("What do you wnat? : ");
```

:프로그램을 시작하는 문구를 출력한다.

- 사용자의 입력값을 읽어오기 위해 Scanner 사용.
- 한 줄 단위로 읽어온 값을 command 변수에 넣고, 띄어쓰기를 기준으로 한 문장을 String배열로 바꾸어 commandSplit에 넣어준다.

- 만약, commandSplit[0]이 "#JOIN"이라면 이미 존재하는 채팅방에 입장할 수도, 새로운 채팅방을 만들어 입장할 수도 있다.
- #JOIN (참여할 채팅방의 이름) (사용자 이름)의 형태로 입력이 들어올 것이기에, commandSplit[0] <- #JOIN

  commandSplit[1] <- (참여할 채팅방의 이름)

  commandSplit[2] <- (사용자 이름)

  이 순서대로 입력되어 저장된다.
- 사용자의 입력으로 받아온 참여할 채팅방의 이름과 사용자 이름을 각각 chatRoomName과 username 변수에 넣어둔다.
- 또한, 채팅룸에 입장하려면 참여할 채팅룸 이름에 해당하는 IP Address를 알아야 한다.

  ➔ getRoomAddress함수를 이용하여 chatRoomName에 해당하는 IP Address를 mapping해 roomIP 변수에 넣어준다.

```
-   Write write = new Write(portNo, roomIP, userName);
    Thread writer = new Thread(write);
    Read read = new Read(portNo, roomIP, userName);
    Thread reader = new Thread(read);
```

:이제 해당 채팅룸에서 메시지를 보내기도, 동시에 받기도 해야 한다.

이를 구현하기 위해, Tread를 이용한다.

➔ 포트넘버 portNo와 참여할 채팅룸의 IP 주소인 roomIP, 사용자 이름인 username, 총 3개의 파라미터를 입력받아 Write 객체(write) 생성.

➔ write를 통해 Thread 객체(writer)를 만든다.

➔ 포트넘버 portNo와 참여할 채팅룸의 IP 주소인 roomIP, 사용자 이름인 username, 총 3개의 파라미터를 입력받아 Read 객체(read) 생성.

➔ read를 통해 Thread 객체(reader)를 만든다.

```
writer.start();
reader.start();
writer.join();
reader.join();
```

: 2개의 Thread가 시작한다.

: Thread가 끝날 때까지 프로그램이 먼저 종료되지 않게끔 기다리기 위해 join메소드 사용.

```
else{
    System.out.println("Wrong Command");
    System.out.println("ReWrite Command! ");
}
```

: 만약, 처음 들어온 commandSplit[0]이 '#JOIN'이 아니라면 틀린 명령어임을 알리고, 다시 입력하게끔 적절한 문장을 출력한다.

<getRoomAddress(String chatRoomName)>

- 파라미터로 String타입의 chatRoomName을 받는다.
- chatRooomName에 해당하는 IP Address를 반환해주는 함수이다.
- chatRooomName <-> IP Address mapping을 위해, "SHA-256" 해시를 이용해서 Multicast address("225.x.y.z")로 변환해 줄 것이다.
- x, y, z 값을 구해야 한다.

- Return할 ipAddress변수는 null로 초기화한다.

```
MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
```

: "SHA-256"메시지에 해당하는 digest object 생성.

```
messageDigest.update(chatRoomName.getBytes());
```

: 채팅룸 이름을 byte data로 변환하고, 이를 사용해 digest를 갱신해준다. (update활용)

```
byte byteMessage[] = messageDigest.digest();
```

: byte배열로 해시를 반환한다.

```
- StringBuffer stringBuffer =  new StringBuffer();
  for(int i = 0; i < byteMessage.length; i++){
     String string = Integer.toString((byteMessage[i] & 0xff) + 0x100,
  16).substring(1);
     stringBuffer.append(string);
  }
```

: StringBuffer 자료형의 객체(stringBuffer)를 생성해서, 문자열을 계속해서 추가한다.

추가할 문자(byteMessage)를 HexString으로 변환하여 stringBuffer에 추가한다.

- stringBuffer의 내용을 String으로 변환해 ipAddress에 넣어준다.
- 이 ipAddress를 byte형으로 변환하여, 뒤 3자리를 뽑아 온다.

```
- byte hashAddress[] = ipAddress.getBytes();
  //64 개(0~63)니까 61, 62, 63 번째 가져와서 각각 x, y, z 에 넣어
  byte x = hashAddress[61];
  byte y = hashAddress[62];
  byte z = hashAddress[63];
  ipAddress = "225." + x + "." + y + "." + z;
```

: 채팅룸 이름에 mapping되는 ipAddress 생성완료.

- Mapping된 ipAddress를 return한다.

(5) 전체적인 디자인

➔ 프로그램 실행 시 포트넘버는 입력받는다.

➔ #JOIN명령어를 통해 각 사용자들은 원하는 채팅방에 자신의 이름으로 입장할 수 있다.

#JOIN (채팅방 이름) (사용자 이름)

➔ 입력받은 채팅방 이름의 IP 주소를 mapping해주는 데에는 getRoomAddress()함수를 사용한다.

➔ getRoomAddress()함수는 "SHA-256" 해시를 이용해서 Multicast address("225.x.y.z")로 변환해 줄 것이다.

➔ 또한 P2P 멀티캐스트 채팅 프로그램으로, client와 server가 따로 존재하는 것이 아닌 peer가 client와 server 역할 모두를 해주어야 한다.

➔ 따라서 Thread를 활용해 메시지를 보내는 writer와 메시지를 받는 reader, 두 Thread를 만들어 두 역할을 동시에 수행하도록 구현한다.

2. 프로그램 컴파일 방법(과제 수행 방법)

(1) 원하는 폴더로 가서 압축을 풉니다.

(2) CMD창 실행합니다.

(3) "p2pChat.java"파일이 있는 주소(압축을 푼 폴더)로 이동하여 아래 명령어를 실행합니다.

```
D:₩HANYANG₩2-2_2₩compute₩ass1test>ls
Assignment#1_2020001658_leeyumin

D:₩HANYANG₩2-2_2₩compute₩ass1test>cd Assignment#1_2020001658_leeyumin

D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>ls
Assignment1_??????????????????.docx  Read.java  Write.java  p2pChat.java

D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>javac p2pChat.java -encoding UTF8
Note: Some input files use or override a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
```

(4) javac p2pChat.java -encoding UTF8 명령어를 실행합니다.

- p2pChat.java 컴파일을 위해 실행하는 명령어
- java 파일에 포함되어 있는 한글 주석으로 인한 오류로 반드시 위의 명령어로 컴파일해주어야 합니다.

(5) 프로그램을 실행합니다. 방법은 아래와 같습니다.

- java p2pChat (포트번호) 명령어를 실행합니다.

  Ex. Java p2pChat 1500

```
D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>javac p2pChat.java -encoding UTF8
Note: Some input files use or override a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>java p2pChat 1000
P2P Chatting room start!
What do you wnat? :
#JOIN cNet yumin
```

(6) 프로그램이 실행되면, 아래의 행동을 할 수 있습니다.

- #JOIN 명령어를 통해 채팅방 입장하기.

  : #JOIN (채팅방 이름) (사용자 이름)

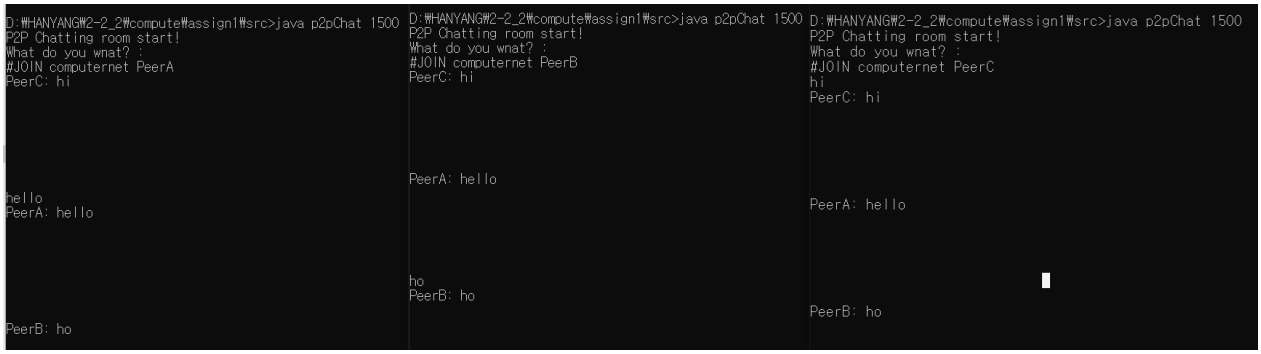
- 메시지 전송하기

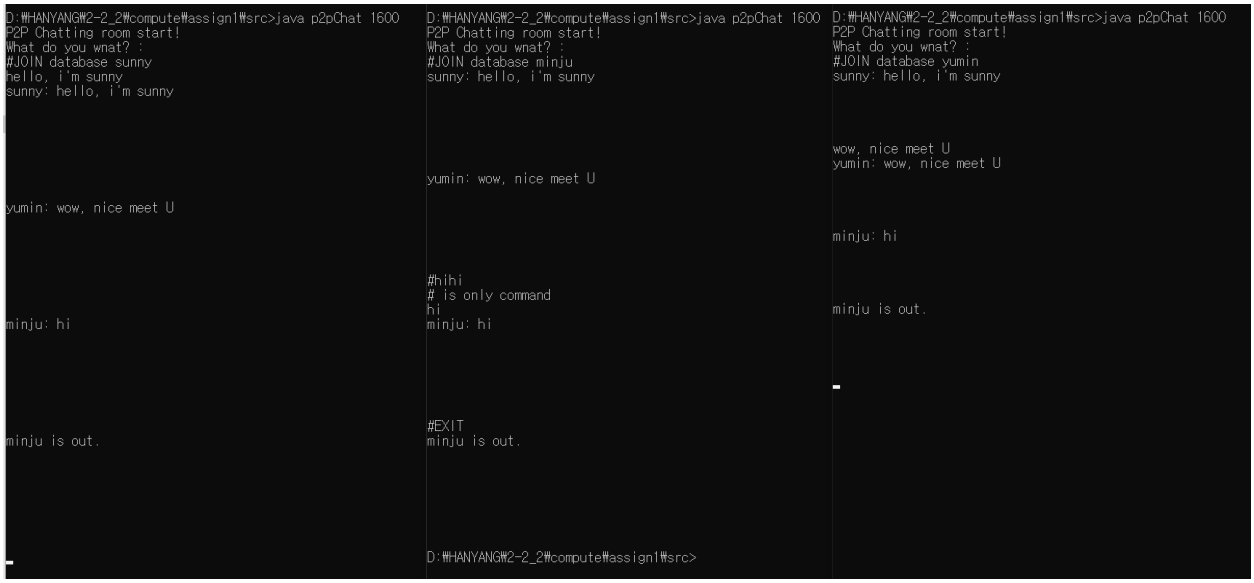
  * cannot contain "#"

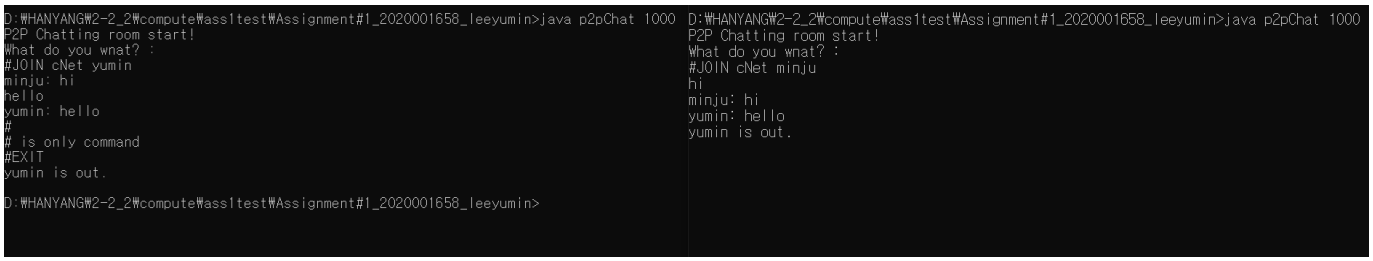
- #EXIT 명령어를 통해 채팅방 퇴장하기.

  : #EXIT

3. 실행 화면 예시


D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1500
P2P Chatting room start!
What do you wnat? :
#JOIN computernet PeerA
PeerC: hi
hello
PeerA: hello
ho
PeerB: ho
PeerB: ho
D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1500
P2P Chatting room start!
What do you wnat? :
#JOIN computernet PeerB
PeerC: hi
PeerA: hello
PeerB: ho
D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1500
P2P Chatting room start!
What do you wnat? :
#JOIN computernet PeerC
hi
PeerC: hi
PeerA: hello
PeerB: ho


D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1600
P2P Chatting room start!
What do you wnat? :
#JOIN database sunny
hello, i'm sunny
sunny: hello, i'm sunny
yumin: wow, nice meet U
minju: hi
#hihi
is only command
hi
minju: hi
#EXIT
minju is out.
D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1600
P2P Chatting room start!
What do you wnat? :
#JOIN database minju
sunny: hello, i'm sunny
yumin: wow, nice meet U
minju is out.
D:₩HANYANG₩2-2_2₩compute₩assign1₩src>
D:₩HANYANG₩2-2_2₩compute₩assign1₩src>java p2pChat 1600
P2P Chatting room start!
What do you wnat? :
#JOIN database yumin
sunny: hello, i'm sunny
wow, nice meet U
yumin: wow, nice meet U
minju: hi
minju is out.


D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>java p2pChat 1000
P2P Chatting room start!
What do you wnat? :
#JOIN cNet yumin
minju: hi
hello
yumin: hello
#
is only command
#EXIT
yumin is out.
D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>
D:₩HANYANG₩2-2_2₩compute₩ass1test₩Assignment#1_2020001658_leeyumin>java p2pChat 1000
P2P Chatting room start!
What do you wnat? :
#JOIN cNet minju
hi
minju: hi
yumin: hello
yumin is out.