

디자인 및 구현

실행 방법

`javac Peer.java` 를 통해 실행파일을 만든다. (이 과정에서 경고 메시지가 출력될 수 있음)

`java Peer portNo` 를 입력하여 실행한다. `portNo`는 16비트로 표현될 수 있는 부호가 없는 정수이다.

프로그램이 실행되었으면 `#JOIN` (참여할 채팅방의 이름) (사용자 이름)을 한줄에 입력하여 실행한다. 만약 조건과 다르게 입력하였으면 처음부터 다시 입력한다. 사용자 이름은 509바이트보다 작아야 정상 작동한다.

채팅방 입장에 성공했으면 메시지를 입력해 채팅을 시작할 수 있다. `#`으로 시작되는 한 줄은 메시지 전달을 하지 않으며, `#EXIT` 를 입력받으면 프로그램이 종료된다.

프로그램 설계

class Peer

프로그램 실행 인자로 포트 번호를 받는다.

`#JOIN` (참여할 채팅방의 이름) (사용자 이름) 이라는 입력이 들어올 때 까지 계속해서 한줄씩 입력을 받는다.

만약 올바른 입력이 들어오면 “SHA-256” hash를 이용해서 IP주소를 구한다.

실행 인자로 받은 포트 번호와 해시를 통해 구한 IP주소를 이용해서 멀티캐스트소켓의 그룹에 들어간다.

`send` 스레드와 `receive` 스레드를 실행한다.

class send

생성자로 `MulticastSocket`, `IPAddress`, `String` (사용자 이름), `Int` (`portNo`)를 받는다.

`while`문을 통해 계속해서 한줄씩 입력을 받는다.

만약 `#`으로 시작하는 입력이 들어오면 그 입력은 전송하지 않는다. 그 입력이 `#EXIT`라면 멀티캐스트 그룹을 떠나고 `while`문을 종료한다.

`#`으로 시작하지 않는 입력이 들어오면 그 입력에 사용자 이름을 식별할 수 있도록 앞부분에 “(사용자 이름): “을 추가해준다. ...(1)

`DatagramPacket` 클래스를 이용하여 멀티캐스트 그룹 내의 유저들에게 메시지를 전달한다.

만약 Srtng의 길이가 512바이트를 넘어간다면 String의 512바이트를 기준으로 String을 두 부분으로 나눈다. (String이 1512바이트라면 512바이트와 1000바이트로 나눈다.) 앞 512바이트는 전송하고 뒷부분에 대해서는 (1)부터 다시 반복한다.

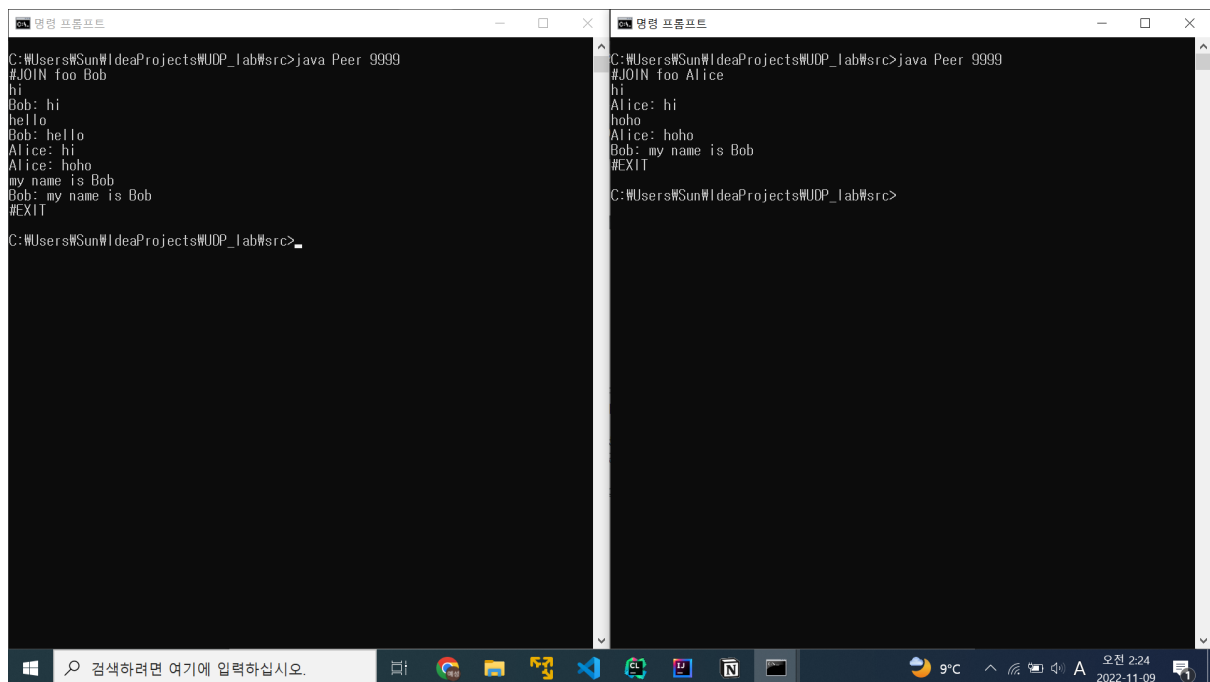
class receive

while문을 통해 1과 2를 반복한다.

1. 멀티캐스트 그룹으로부터 DatagramPacket을 받는다.
2. 패킷을 문자열로 바꿔주고 화면에 출력해준다.

SocketException이 발생하면 반복문을 종료한다. (SocketException은 send 스레드에서 #EXIT을 입력받았을 때 발생하게 된다.)

실행화면 예시



```
C:\Users\Sun\IdeaProjects\WUDP_lab\src>java Peer 9999
#JOIN foo Bob
hi
Bob: hi
hello
Bob: hello
Alice: hi
Alice: hoho
my name is Bob
Bob: my name is Bob
#EXIT
C:\Users\Sun\IdeaProjects\WUDP_lab\src>_

C:\Users\Sun\IdeaProjects\WUDP_lab\src>java Peer 9999
#JOIN foo Alice
hi
Alice: hi
hoho
Alice: hoho
Bob: my name is Bob
#EXIT
C:\Users\Sun\IdeaProjects\WUDP_lab\src>
```