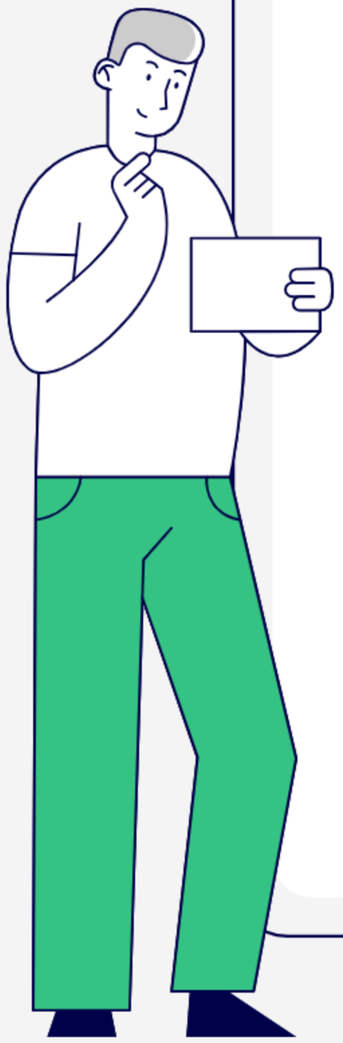
신연재

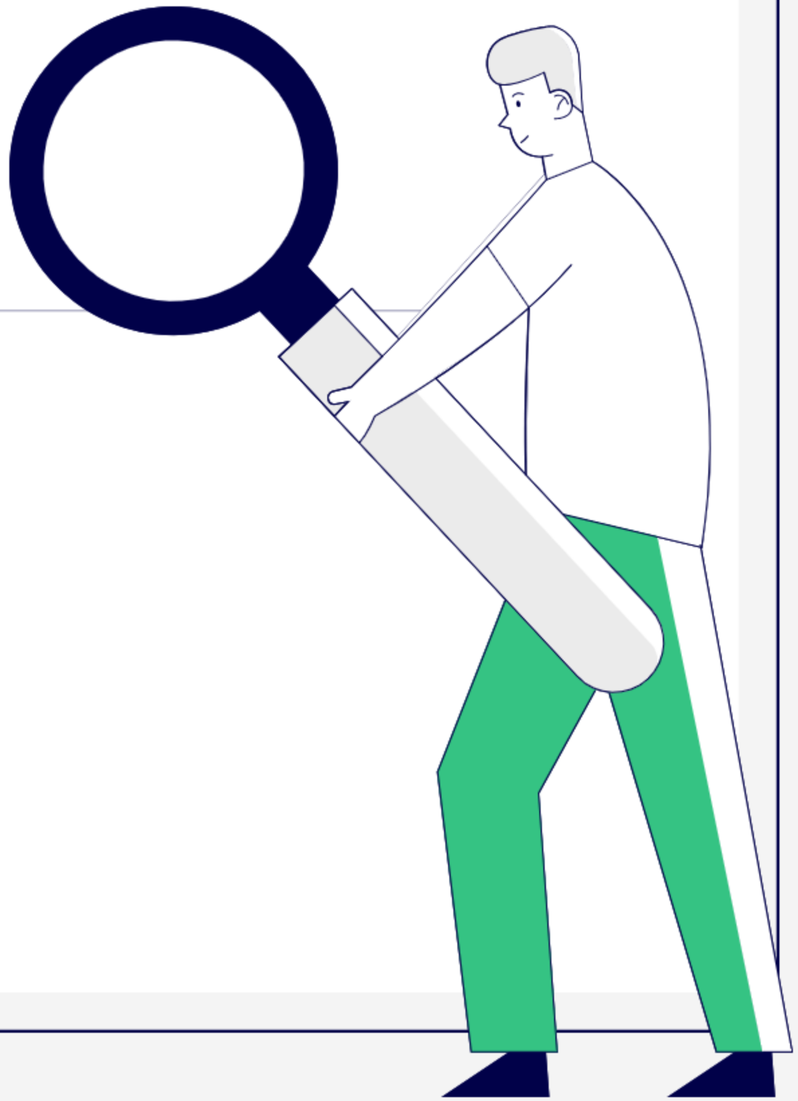


채팅 프로그램

교육과정 2개월차(2월15일~2월19일)

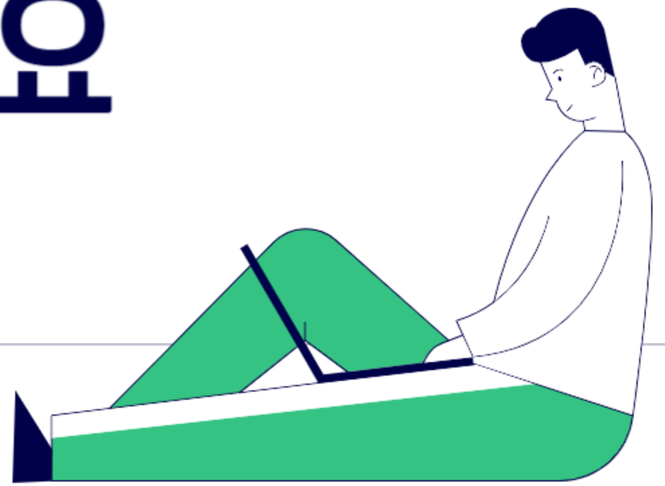
신연재재

CONTENTS



신연재

프로젝트 개요



본 프로젝트는 JAVA교육 2개월차에 만든 간단한 채팅 프로그램 입니다.
채팅방에 들어온 유저와 텍스트를 이용한 간단한 채팅이 가능하고
채팅방에 들어온 유저의 닉네임을 확인가능하며 귓속말 기능을 포함하고 있습니다.

주요 기술

- JAVA Socket,DataOut/InputStream 사용
- TCP/IP 기반의 멀티스레드 채팅 서버와 채팅 클라이언트를 구현합니다
- 화면 구성은 AWT의 Frame을 이용하여 구현하였습니다.

코드 구현

화면 구현

신연재

요구 분석

간단한 채팅기능

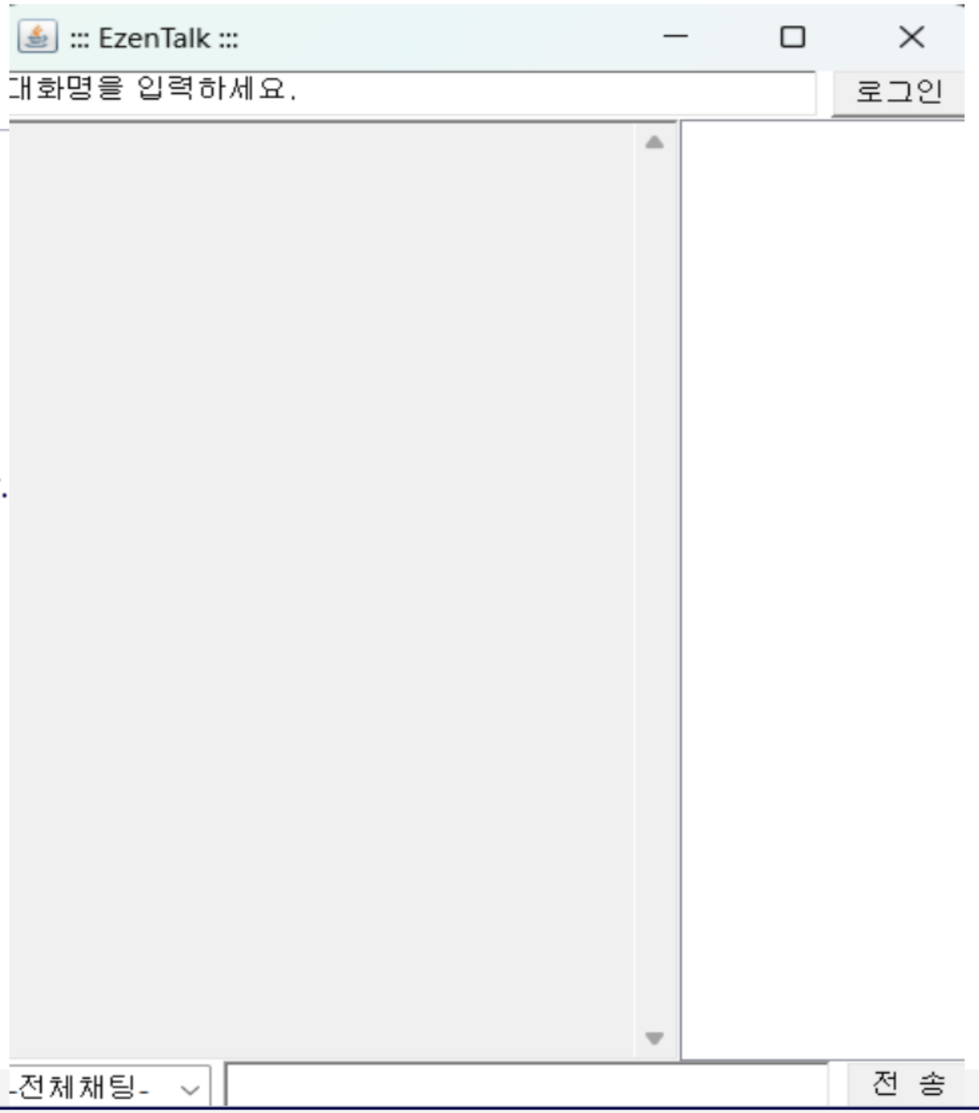
닉네임으로 입장

여러명과의 채팅

귓속말 기능 구현

실행 화면

초기 실행 화면



처음 프로그램을 실행하면 상단에 대화명을 입력 할 수 있습니다.

대화명을 입력하고 로그인을 누르면 왼쪽에 대화방에 참여했단 사실을 알 수 있습니다.

하단에는 원하는 채팅을 입력하고 전송 버튼을 누르면 채팅이 입력이 됩니다.

채팅종류에는 전체채팅과 귓속말 채팅이 있습니다.

실행 화면

입장 화면

사용자 닉네임을 입력 후 로그인 버튼을 통해 입장을 하게되면
채팅창에 사용자가 입장하였다는 글이 나오게 됩니다.
receive() 메서드는 새로운 스레드를 생성합니다. 서버로부터 지속적으로 메시지를 수신하고 처리하는 역할을 합니다. switch 문을 사용하여 명령어에 따라 다른 작업을 수행합니다.
CONNECT-새로운 사용자가 입장했음을 알리는 메시지를 화면에 출력합니다.
CONNECTION_LIST-현재 접속 중인 사용자 목록을 갱신합니다.
MULTI_CHAT-다중 사용자에게 전송된 채팅 메시지를 화면에 출력합니다.
DM (Direct Message)-특정 사용자에게 전송된 직접 메시지를 화면에 출력합니다.
DIS_CONNECT-사용자가 퇴장했음을 알리는 메시지를 화면에 출력하고 사용자 목록을 갱신합니다.


```
public void receive() throws IOException {  1 usage
    Thread thread = run() → {
            try {
                while (true) {
                    String jsonMessage = in.readUTF();
                    JSONObject jsonObject = new JSONObject(jsonMessage);
                    String command = jsonObject.getString( key: "command");
                    String nickName = jsonObject.getString( key: "nickName");
                    switch (command) {
                        case "CONNECT":
                            chatFrame.appendMessage("---------------------------");
                            chatFrame.appendMessage("♡♡♡[" + nickName + "]님이 입장하였습니다.♡♡♡");
                            chatFrame.appendMessage("---------------------------");
                            break;
                        case "CONNECTION_LIST":
                            JSONArray jsonArray = jsonObject.getJSONArray( key: "list");  // SocketClient 참고
                            chatFrame.NickNameList(jsonArray);
                            chatFrame.ChoiceNickNameList(jsonArray);
                            break;
                        case "MULTI_CHAT":
                            String chatMessage = jsonObject.getString( key: "message");
                            chatFrame.appendMessage("[" + nickName + "]" + chatMessage);
                            break;
                        case "DM":
                            String dmMessage = jsonObject.getString( key: "message");
                            String senderName = jsonObject.getString( key: "nickName");
```

실행 화면

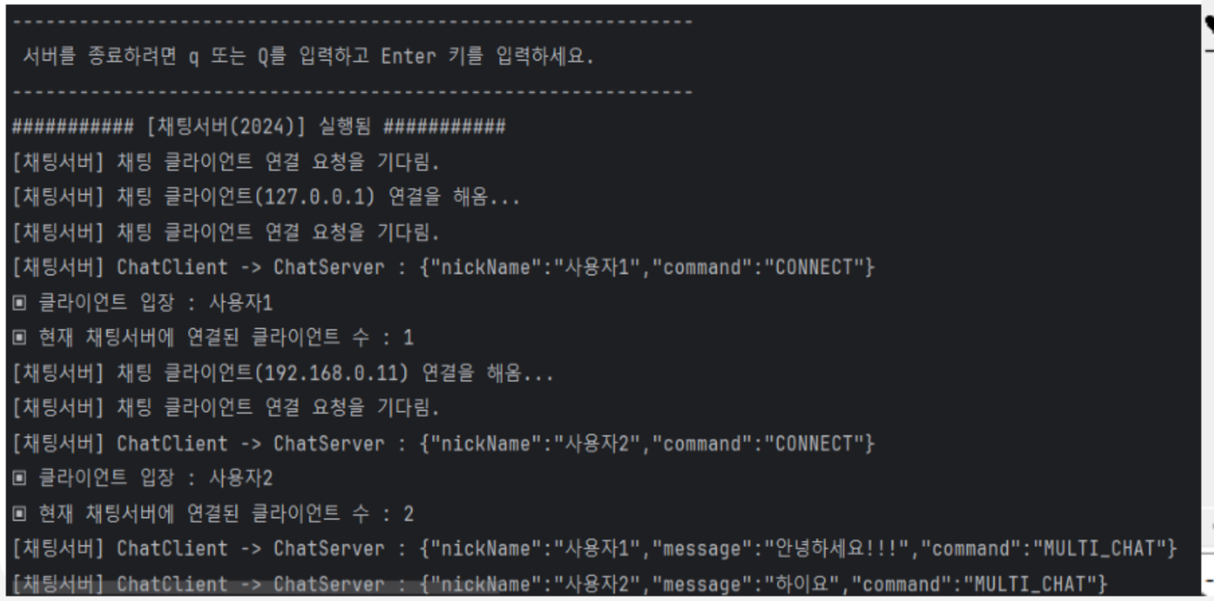
초기 실행 화면

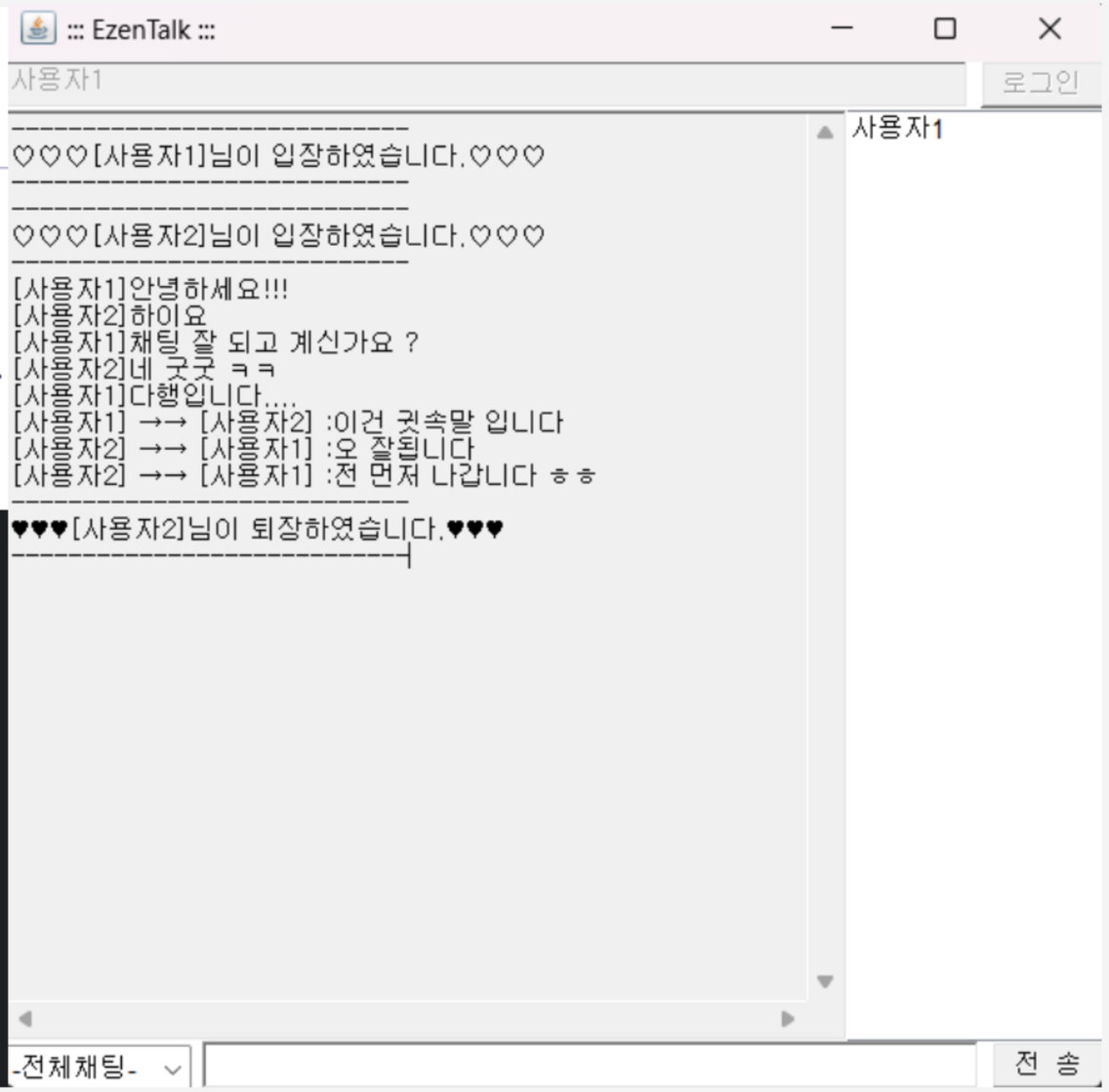
대화가 시작이 되면 서로 자유롭게 채팅을 주고 받을 수 있습니다.

귓속말 기능은 프로그램 왼쪽 하단에서 사용자를 선택 후 채팅을 보내면 됩니다.

사용자가 퇴장을 했을 경우 채팅창에 사용자가 퇴장 하였다는 메시지가 출력 됩니다.

채팅서버에서는 연결된 클라이언트 수와 사용자의 채팅내역이 나오게 됩니다.





신연재

프로젝트 후기

Java 수업을 두달가량 듣고 진행한 프로젝트 였습니다. TCP/IP 기반의 채팅 클라이언트를 Java로 구현한 것으로
처음에는 소켓의개념과 스레드 처리에 대해 잘 알지 못해서 어려움이 많았습니다.
특히 클라이언트와 서버 간의 통신을 원활하게 구현하는 데 많은 시간을 투자해야 했습니다.
프로젝트를 통해 소켓을 사용하여 클라이언트와 서버가 어떻게 데이터를 주고받는지, 그리고 다중 스레드 환경에서 어떻게 동시성을 관리하는지에 대해 깊이 있게 배울 수 있었습니다. 또한 JSON을 이용한 데이터 전송 및 처리 방법도 익힐 수 있었습니다.
채팅 클라이언트를 완성하고 나니 큰 성취감을 느낄 수 있었습니다. 특히, 여러 사용자가 동시에 접속하고 메시지를 주고받을 수 있는 기능을 구현했을 때 가장 보람을 느꼈습니다. 또한, 사용자 인터페이스를 만들어 실제로 작동하는 프로그램을 구현할 수 있어서 뿌듯했습니다.
다음에 비슷한 프로젝트를 진행한다면 예외 처리를 좀 더 철저히 하고 싶습니다. 또한 보안적인 측면도 고려하여 데이터를 암호화하는 방법도 추가해보고 싶습니다. 예를들어 아이디,비밀번호로 입장이라던지 채팅방 비밀번호 설정이 있겠습니다.
이번 프로젝트는 Java 프로그래밍 실력을 향상시키고 네트워크 프로그래밍에 대한 이해를 넓히는 좋은 기회가 되었습니다.