자바 채팅 프로그램 보고서



학 과 : 인공지능소프트웨어

과 목 : 자바응용실습

학 번 : 2403110265

이 름 : 김해준

제출일 : 2024-12-12

### **1. 서론**

* **프로젝트 개요**

본 프로젝트는 자바 소켓 프로그래밍을 활용하여 채팅 프로그램을 개발하는 것입니다. 이 프로그램은 자바의 소켓 객체와 입출력 스트림을 이용하여 두 대 이상의 컴퓨터 간에 데이터를 송수신할 수 있는 실시간 채팅 환경을 제공합니다.

이 프로젝트의 주요 목표는 사용자가 텍스트 기반의 메시지를 송수신할 수 있도록 하는 채팅 프로그램을 구현하는 것입니다. 서버와 클라이언트 간의 연결을 설정하고, 데이터를 주고받는 과정에서 발생할 수 있는 다양한 예외 처리와 오류를 관리하는 방법을 학습할 수 있습니다.

프로젝트의 핵심 기술은 다음과 같습니다:

1. **자바 소켓 객체(Socket)**: 클라이언트와 서버 간의 연결을 관리하고, 데이터 송수신을 담당합니다.
2. **입출력 스트림(Input/Output Streams)**: 데이터를 전송하기 위한 채널로, 클라이언트와 서버 간의 메시지를 처리합니다.
3. **멀티스레딩**: 여러 클라이언트와 동시에 채팅을 할 수 있도록 서버 측에서 멀티스레딩을 활용하여 각각의 클라이언트와의 통신을 관리합니다.

이 프로그램을 통해 자바 소켓 프로그래밍의 기본 개념을 실습하고, 실제 네트워크 환경에서 클라이언트-서버 구조의 작동 방식을 이해하는 데 중점을 둡니다.

* **주요 기능 나열**

1. 다수 사용자 채팅 지원
2. 메시지 발신자 및 전송 시간 표시
3. 닉네임 변경 기능
4. 멘션 기능
5. 이미지 전송
6. 이미지 다운
7. 채팅 기록 저장
8. 오프라인 상태 알림
9. 온라인 사용자 목록 표시

* **자바 소켓 프로그래밍 설명**
* **ChatServer**

서버는 여러 클라이언트를 관리합니다. ChatServer는 여러 ClientService를 처리하고, 각각의 클라이언트와의 연결을 유지합니다.

* **ClientService**

클라이언트 서비스는 서버와 상호작용하며, 클라이언트의 요청을 서버로 전달하고 응답을 받습니다. ClientService는 ChatServer와의 통신을 담당합니다.

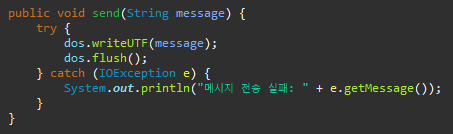
클라이언트 서비스는 ChatClient로부터 메시지를 받아서 처리하고, 서버와 연결된다른 클라이언트에게 메시지를 전달합니다.

* **ChatClient**

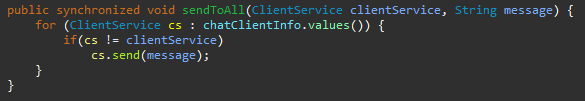
클라이언트는 서버에 연결하여 데이터를 송수신합니다. ChatClient는 ChatServer 에 연결하여 메시지를 보낼 수 있습니다.

### **2. 주요 기능**

1. 다수 사용자 채팅 지원
   * **클라이언트**: 메시지 입력 시 send() 메서드를 통해 클라이언트 서비스에 메시지를 전달합니다.

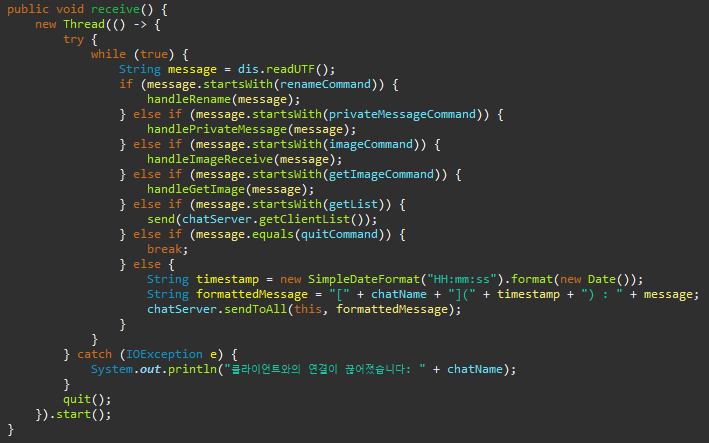


* + **서버**: sendToAll() 메서드는 서버에 연결된 모든 클라이언트에게 메시지를 전달하는 기능을 담당합니다.



파라미터로 clientService를 입력받아 자신을 제외한 나머지 키(클라이언트 닉네임)에게 모두 전송

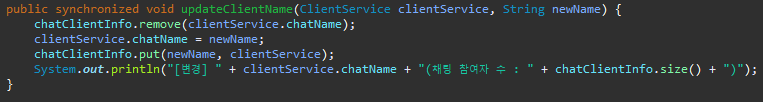
* + **클라이언트 서비스**: 메시지를 수신하고 출력하는 기능을 담당합니다. 클라이언트가 서버로부터 오는 메시지를 출력하는 별도의 스레드를 통해 처리됩니다.



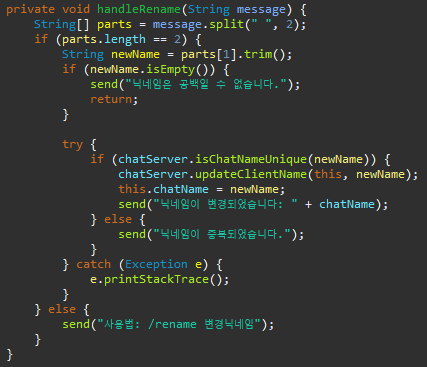
1. 메시지 발신자 및 전송 시간 표시
   * **클라이언트 서비스**: 수신한 메시지에 발신자와 시간을 출력합니다.



1. 닉네임 변경 기능
   * **클라이언트**: 사용자가 닉네임을 입력하고 서버로 전송합니다. /rename 변경명
   * **서버**: 클라이언트의 닉네임을 업데이트합니다.



* + **클라이언트 서비스**: 변경된 닉네임을 클라이언트에 반영합니다.

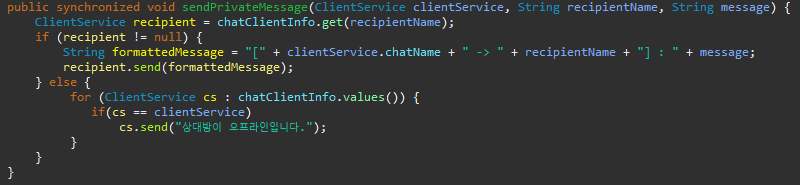


닉네임 중복을 처리하기 위해 chatServer 쪽에서

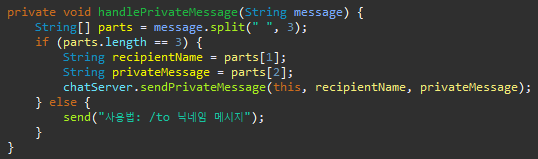


IsChatNameUnique 메서드 선언후 호출

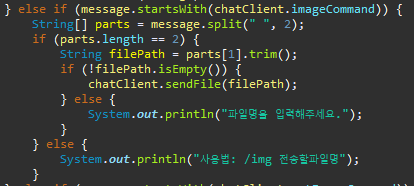
1. 멘션 기능
   * **클라이언트**: 멘션을 포함한 메시지를 작성하여 서버로 보냅니다. /to 상대방 메시지
   * **서버**: 멘션된 사용자가 온라인 상태일 경우 해당 사용자에게만 메시지를 보냅니다.



* + **클라이언트 서비스**: handlePrivateMessage 메서드로 멘션된 사용자의 메시지를 처리합니다.



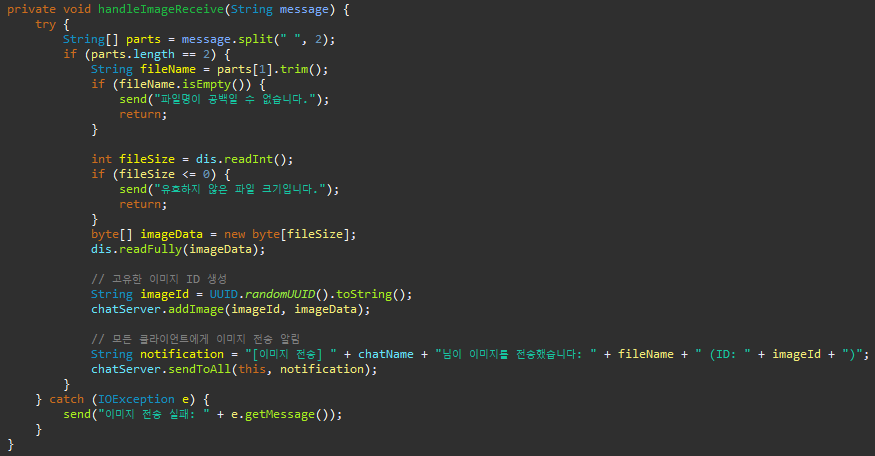
1. 이미지 전송
   * **클라이언트**: 이미지를 서버로 전송합니다.



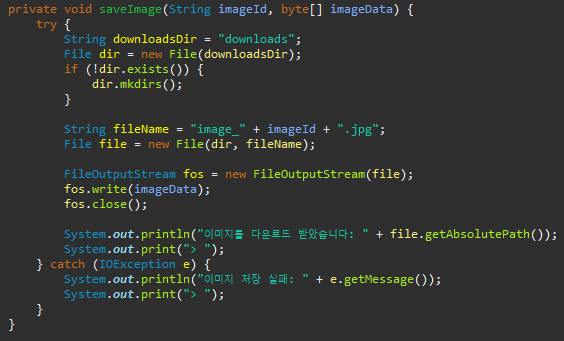
* + **서버**: 이미지를 수신하고, 연결된 모든 클라이언트에게 전송하거나 다운로드할 수 있도록 서버에 저장합니다. 저장할 때 식별하기위해 ImageID를 key로 저장



* + **클라이언트 서비스**: handleImageReceive 메서드로 이미지 입출력 예외 처리 후 서버에 전달합니다.



1. 이미지 다운
   * **클라이언트**: saveImage 메서드로 전달받은 이미지를 downloads 폴더에 다운받습니다.



* + **서버**: 저장되있는 이미지를 클라이언트에게 전달합니다.



* + **클라이언트 서비스**: handleGetImage 메서드로 이미지를 전송합니다.



1. 채팅 기록 저장
   * **클라이언트** : 접속한 이후부터의 채팅 내용을 messageHistory 에 저장해둔다.

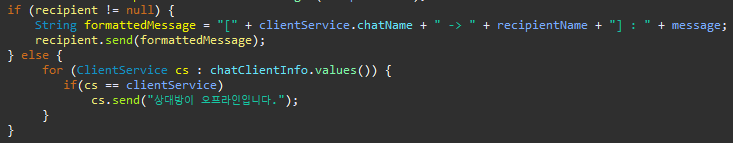


SaveChatHistory 메서드가 호출되면 지금까지의 내용을 txt파일로 저장한다.



서버와 입출력이 있을 때마다 addMessageToHistory 호출해서 messageHistory 에 저장한다.

1. 오프라인 상태 알림
   * **서버**: 클라이언트가 /to 해서 보내려고하는 상대가 오프라인인 경우 메시지를 출력해준다.

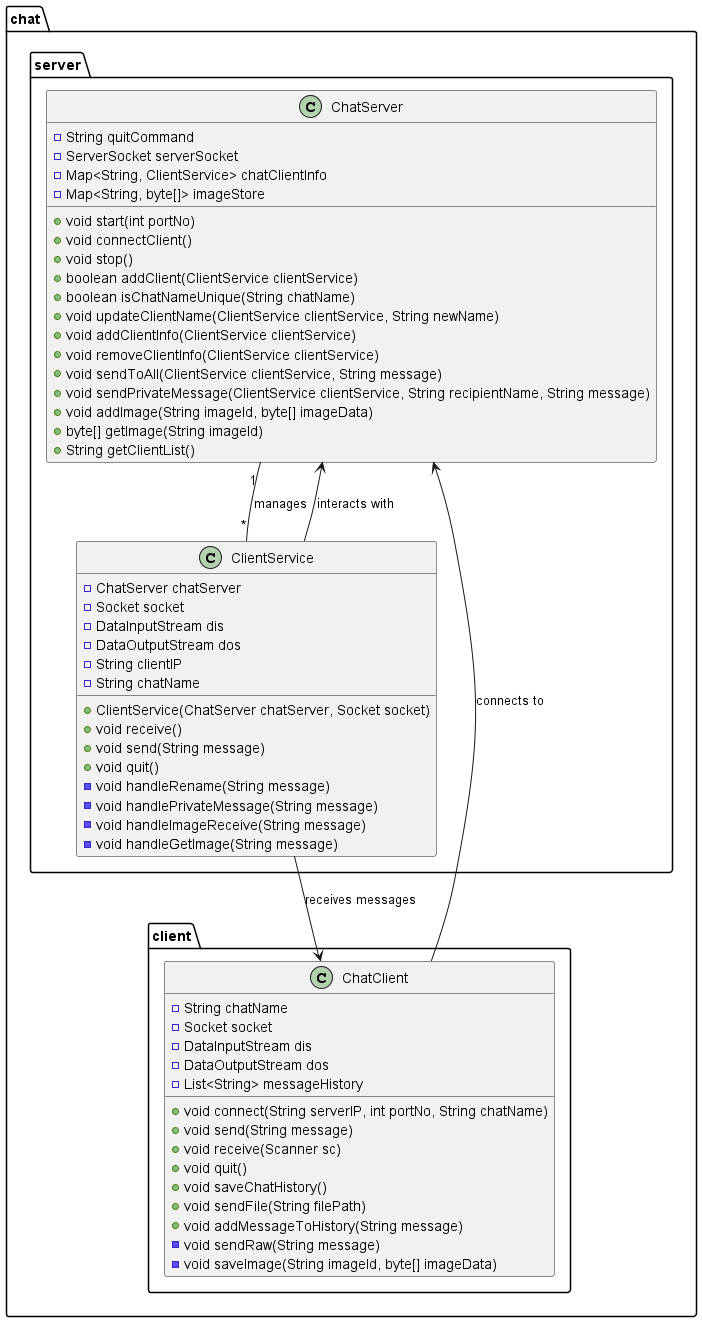


1. 온라인 사용자 목록 표시
   * **서버**: 현재 온라인인 사용자의 목록을 관리하고, 클라이언트가 요청할 때 이를 제공하는 기능을 구현합니다. 서비스에서 호출 시 키(닉네임)을 반환해줍니다.

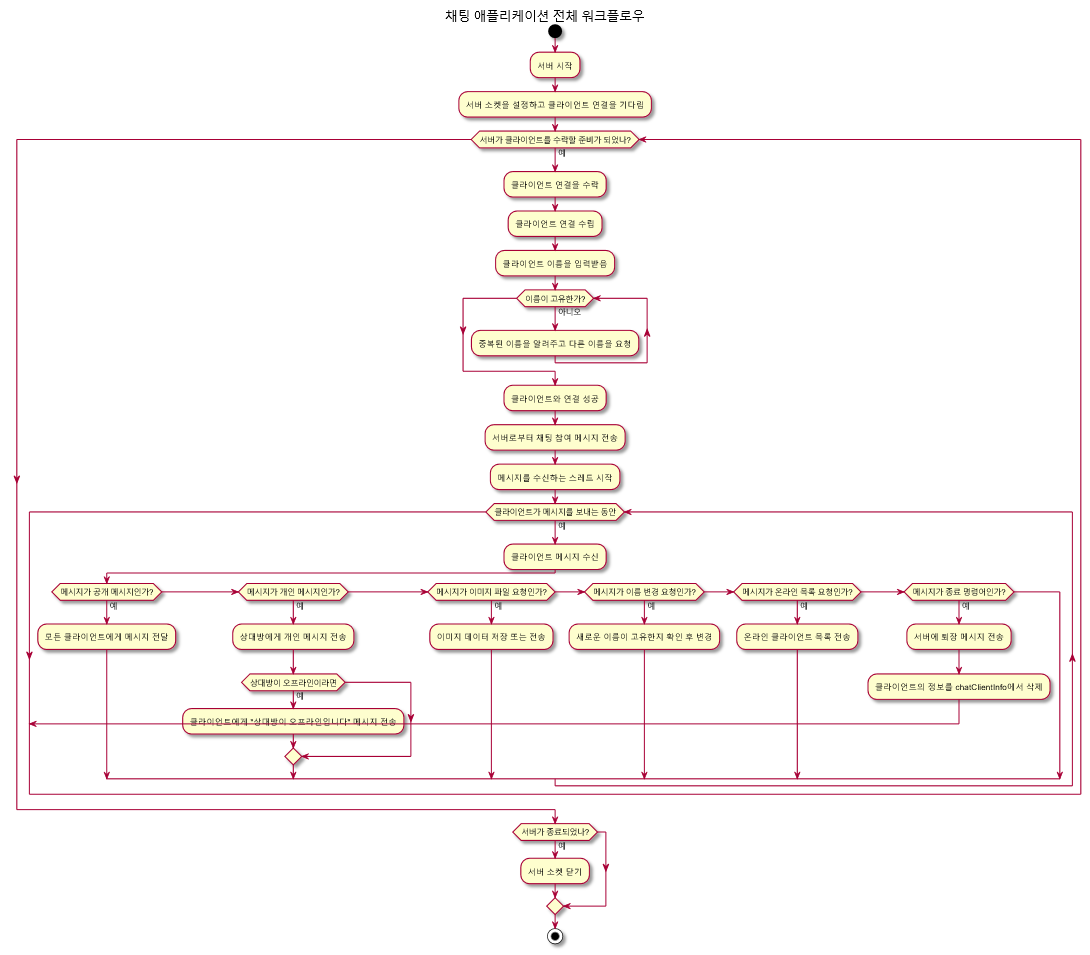


* + **클라이언트 서비스**: 온라인 사용자 목록을 전송합니다.





\*클래스 다이어그램

 \*위 기능들의 흐름을 다이어그램으로 나타낸 워크플로우

### **3. 결론**

본 프로젝트는 자바 소켓 프로그래밍을 활용하여 다수 사용자가 동시에 채팅을 할 수 있는 서버-클라이언트 기반의 채팅 프로그램을 구현하는 것을 목표로하였습니다. 프로그램은 서버와 클라이언트 간의 실시간 데이터 송수신을 위한 소켓 객체와 입출력 스트림을 사용하고, 멀티스레딩을 활용하여 여러 클라이언트와의 동시 연결을 처리하는 구조로 설계되었습니다.

프로젝트를 진행하며 주요 기능으로는 다수 사용자 채팅 지원, 메시지 발신자 및 전송 시간 표시, 닉네임 변경, 멘션 기능, 이미지 전송 및 다운로드, 채팅기록 저장, 오프라인 상태 알림, 온라인 사용자 목록 표시 등이 포함 되었습니다. 이를 통해 채팅프로그램의 기본적인 구조를 이해할 수 있었습니다.

이 프로젝트에서 아쉬운점이 있다면, 기능중에서 log기능이다. 일반적으로 생각했을 때, 로그를 남기고 저장하는 기능을 클라이언트가 사용한다하면, 서버에 저장된 로그를 불러와서 보내줘야 한다고 생각했다. 하지만 구현에 어려움을 겪어 어떻게든 동작하는 로그 저장 기능을 구현하다보니, ChatClient에서 각각 대화내용을 저장하고 파일로 저장하게 작성하였다.

나중에 아주 먼 미래에 기회가 된다면 서버, 클라이언트, 서비스의 구조에 맞춰 이상적으로 굴러가는 채팅 프로그램을 작성해보고싶다.