|  |
| --- |
|  |
|  |

**채팅 프로그램 보고서**

소속 : 한국폴리텍대학 성남캠퍼스 인공지능소프트웨어과

학번 : 2403110273

성명 : 박세연

1. 기본 정보

* 1.1 프로그램 제목

SY 채팅 프로그램

* 1.2 개요

자바 프로그래밍 언어를 사용하여Socket 통신 기반, 기본적인 send, receive 함수를 구현하였고 Thread를 설정하여 다수의 사용자가 참여 가능한 채팅 프로그램으로, 사용자 간 실시간 소통을 지원합니다. 또한 각종 기능을 추가하여 사용자가 원하는 행위를 할 수 있도록 제시합니다.   
본 보고서는 프로그램의 설계와 구현 과정을 설명합니다.

* 1.3 주요 프로세스

서버 구동 -> 클라이언트 구동 -> 클라이언트의 대화명 입력 -> 중복된 대화명이 아니면 채팅방 입장 -> 다수의 사용자와 실시간 채팅

2. 요구사항 분석

- 2.1 요구사항에 따른 과정 및 결과

- 2.1.1사용자는 하나의 채팅방에서 다수의 사용자와 채팅할 수 있습니다.

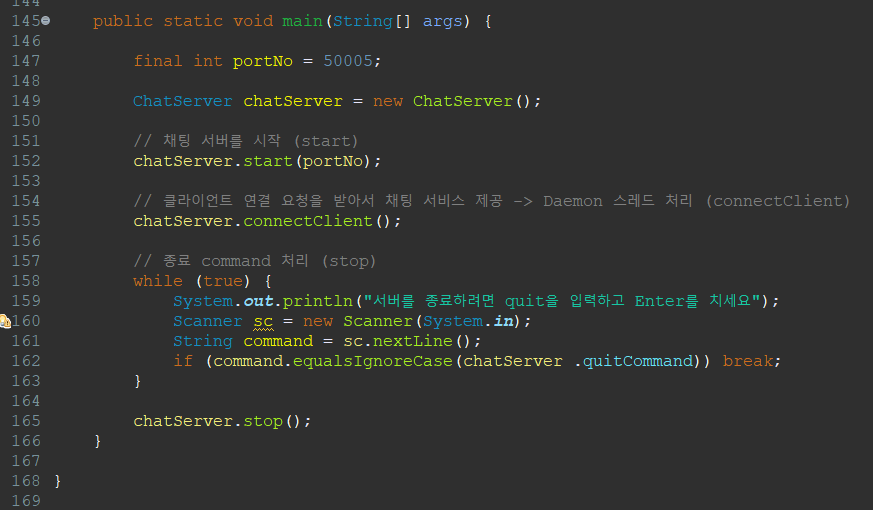


그림 1과정

서버의 start() 함수로 서버 소켓을 생성하고 connectClient() 함수 안에서 쓰레드를 활용하여 클라이언트가 접속할 때마다 클라이언트에 대응할 소켓 생성

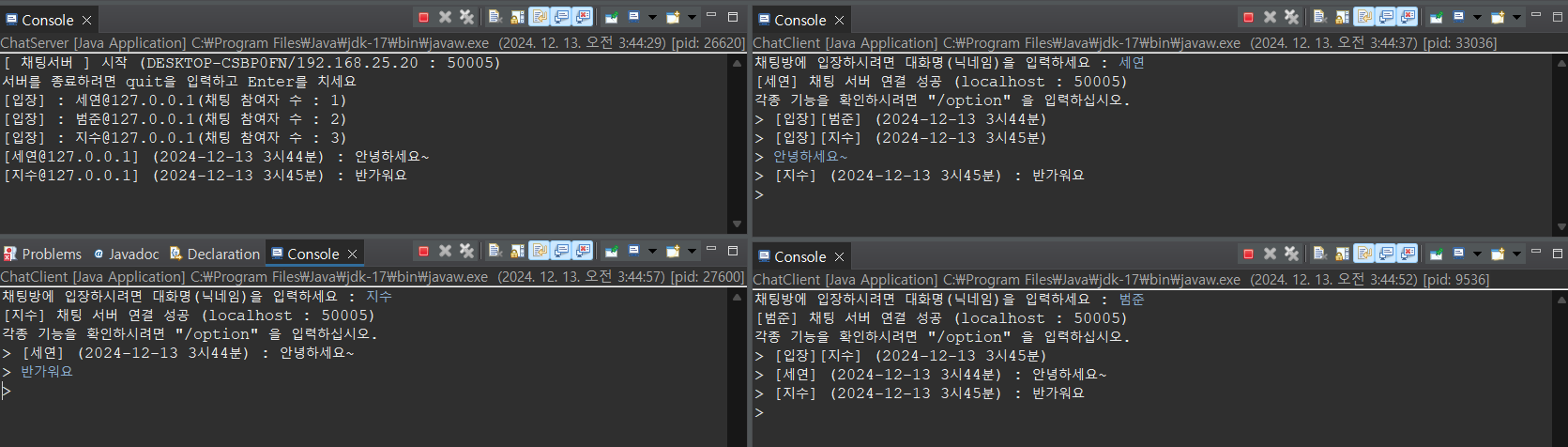


그림 2결과

서버는 한 대이며, 사용자 1은 사용자 2, 3 모두에게 채팅을 보낼 수 있으며, 사용자 2, 3이 보내는 메시지를 받을 수 있음

- 2.1.2받은 메시지는 발신자 이름과 전송 시간을 포함하여 표시됩니다. 예: [발신자이름] (시간): 메시지

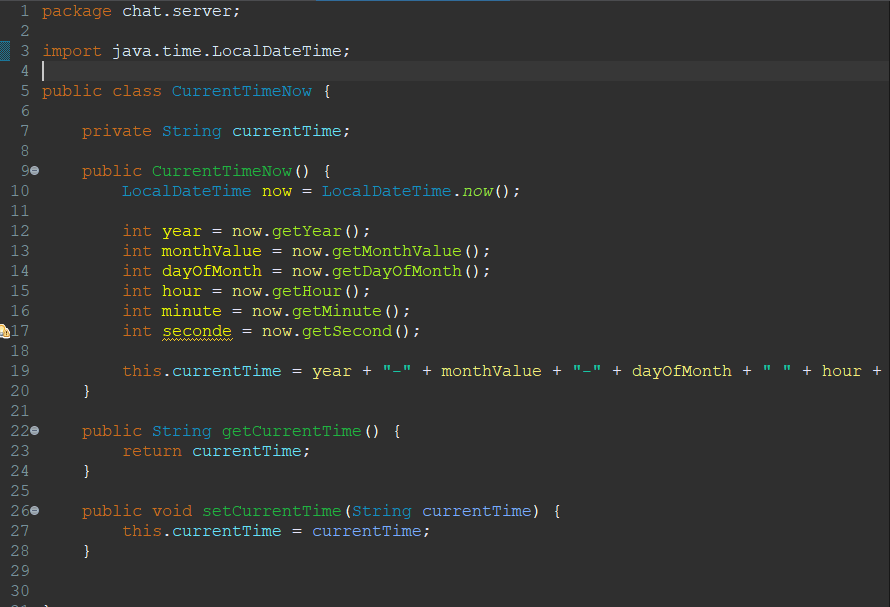


그림 3과정

CurrentTimeNow 객체(클래스)를 만들고 인스턴스를 생성할 때마다 현재 시각을 기록하고 getCurrentTime()으로 현재 시각을 가져옴

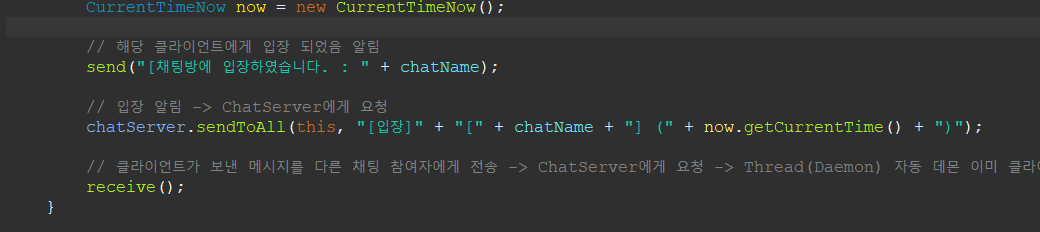


그림 4과정

서버에서 보낸 메시지는 클라이언트가 받으므로 보낼 때 대화명과 현재 시각을 같이 보냄

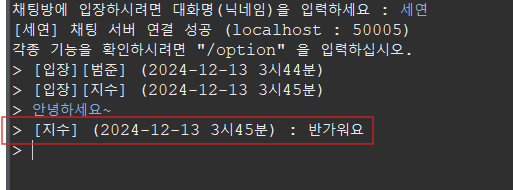


그림 5결과

닉네임 시간 메시지 순으로 표기

- 2.1.3사용자는 자신의 닉네임을 변경할 수 있으며, 중복된 닉네임은 사용이 불가합니다. (명령어: /rename 변경닉네임)

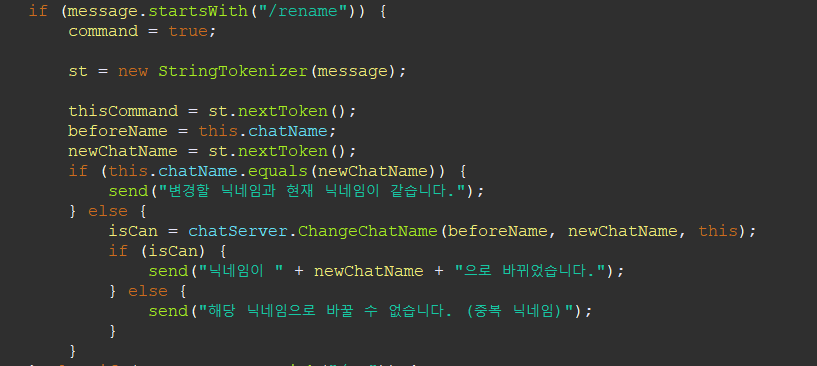


그림 6과정

Message.startsWith(“/rename”)으로 메시지가 /rename 으로 시작한다면 명령어로 받아들이고 command를 true로 바꿈(기본 false)

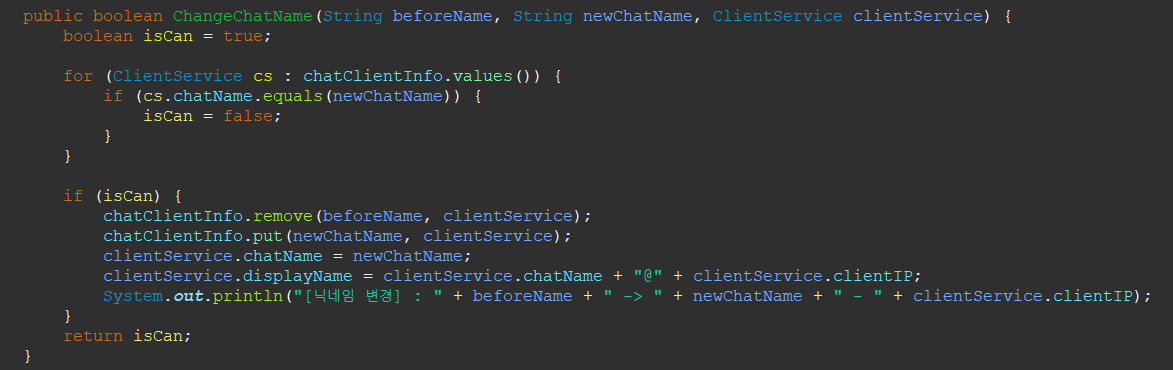


그림 7과정

Hashtable인 chatClientInfo를 순회하면서 바꿀 대화명과 기존 사용자들의 대화명과 중복된다면 return false;  
만약 중복되지 않는다면 Hashtable에서 기존 대화명 키:밸류를 제거 후 새로운 대화명으로 키:밸류 등록함

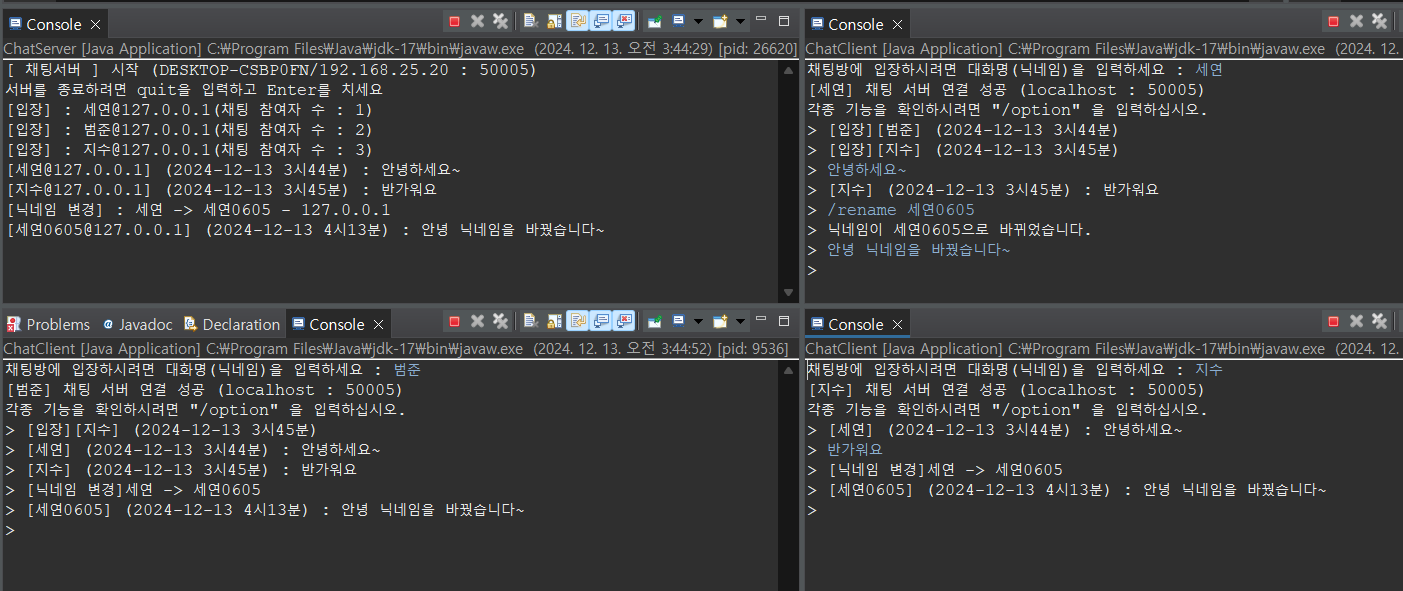


그림 8결과

서버에선 기존 닉네임이 새로운 닉네임으로 바꾼 것을 출력하고

대화명을 바꾼 사용자를 제외하고 모든 사용자에게 닉네임을 바꾼 것을 공지

- 2.1.4사용자는 특정 사용자에게만 메시지를 보낼 수 있습니다. (멘션 기능: /to 전송닉네임 메시지)

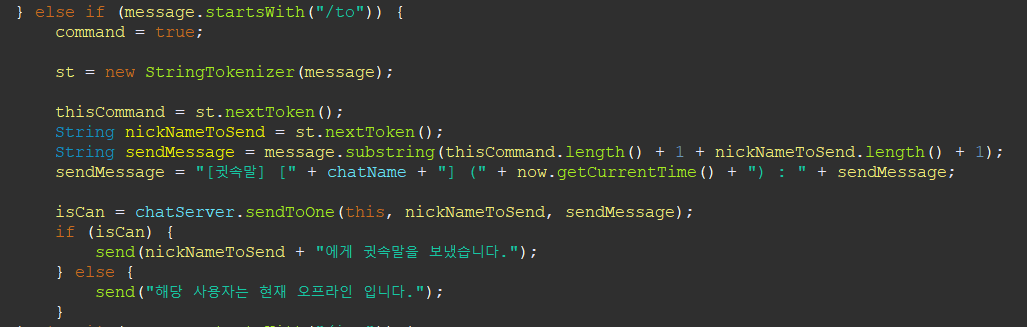


그림 9과정

Message.startWith(“/to”)를 사용하여 명령어를 확인하고, StringTokenizer로 보낼 사용자와 보낼 메시지를 할당 받음

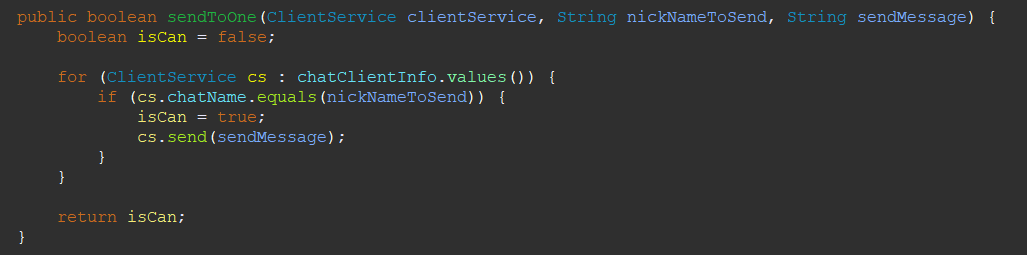


그림 10과정

Hashtable Value값을 순회하며 보낼 닉네임과 같은 ClientService 객체를 찾고 해당 객체의 send() 함수를 실행시킴

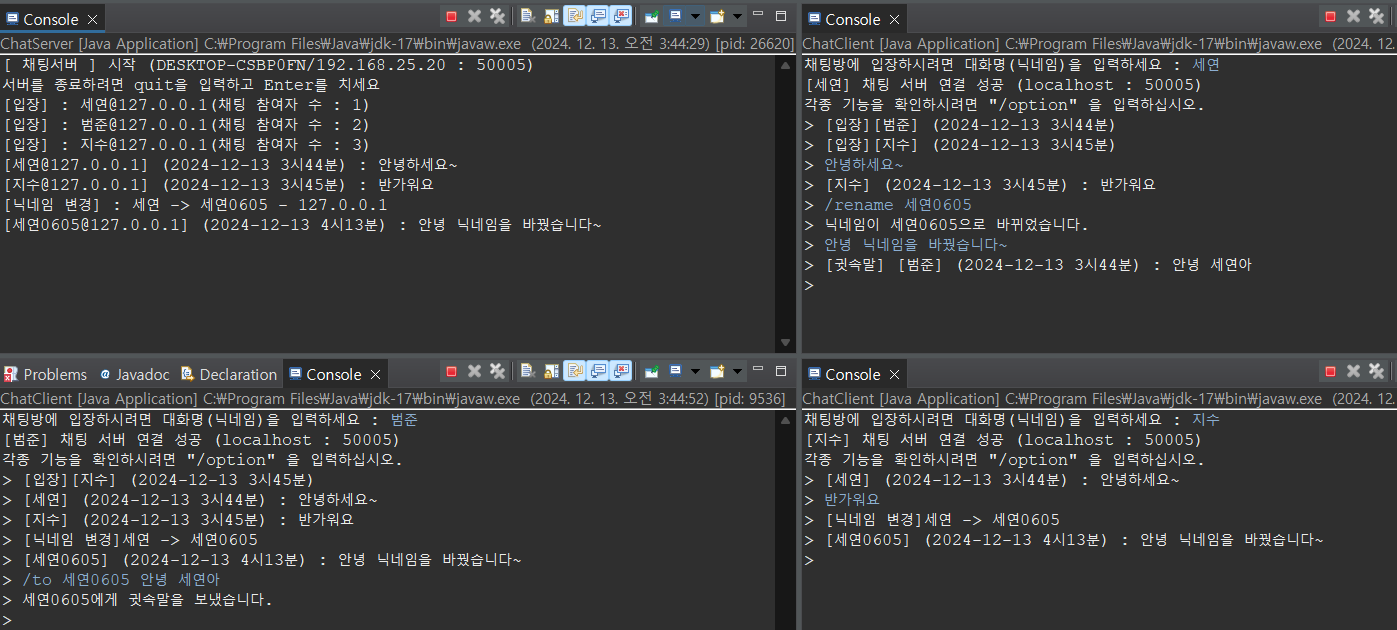


그림 11결과

사용자 범준은 “/to 세연0605 메시지” 기능을 통해 세연0605 대화명인 사용자에게만 메시지를 보내고 다른 사용자는 확인이 불가능함

- 2.1.5사용자는 이미지를 전송할 수 있습니다. (명령어: /img 전송할파일명)

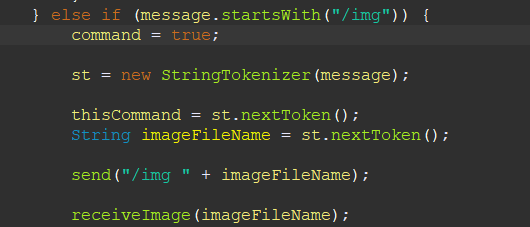


그림 12과정

Message.startWith(“/img”) 함수로 명령어를 확인하고 잘 받았으면 다시 클라이언트에게 /img 파일명 구조로 메시지를 보냄

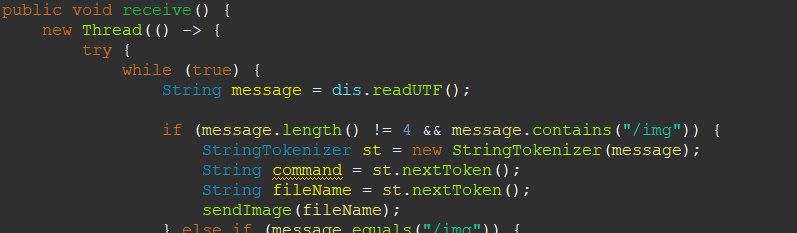


그림 13과정

메시지를 받은 클라이언트는 서버에서 /img 파일명 구조로 보낸 것을 확인하여 sendImage()함수 호출

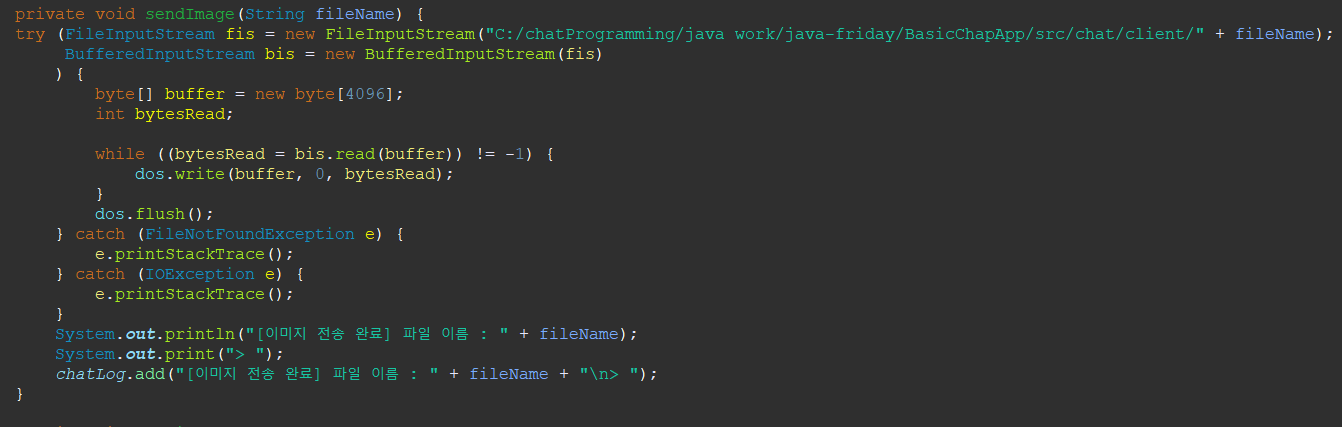


그림 14과정

미리 지정해둔 경로의 파일명을 찾고 FileInputStream으로 데이터를 읽어들여오고

버퍼로 한번에 받을 수 있게 BufferedInputStream으로 받으며 while문을 반복하여

읽어 들여올 수 없을 때 까지 읽은 후 소켓에 출력함

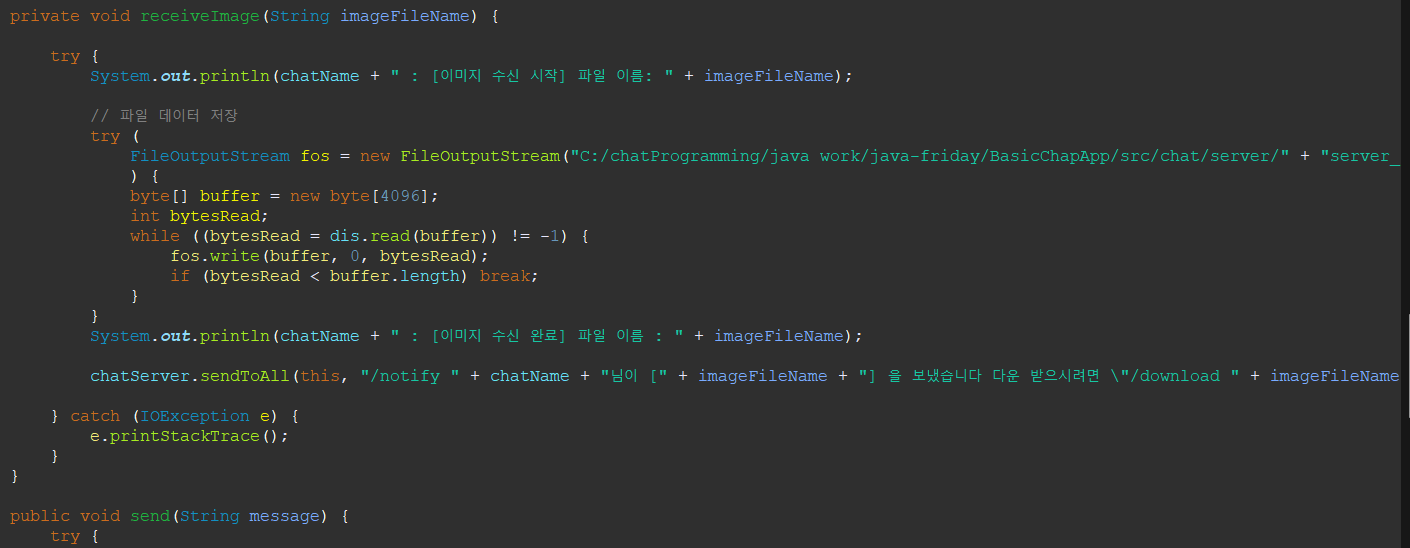
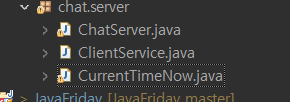


그림 15과정

다시 서버는 클라이언트가 보낸 파일 데이터를 DataInputStream에서(소켓 스트림) while()문으로 더 이상 읽어들여올 수 없을 때 까지 읽어서 fos에 지정한 경로에 저장 후 모든 사용자에게 /notify 구조로 메시지(/download)를 보냄



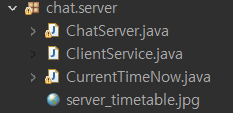


그림 16결과

- 2.1.6사용자는 다른 사용자가 보낸 이미지를 다운로드 여부를 선택할 수 있습니다.

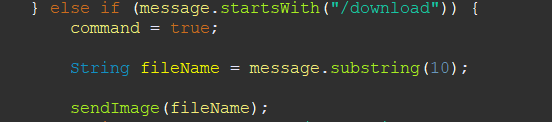


그림 17과정

클라이언트가 /download 파일명을 입력하면 서버에서 message.startWith(“/download”) 로직으로 확인 받고 sendImage() 호출



그림 18과정

sendImage() 함수를 호출하면 서버에 저장된 해당 파일을 찾아 fis로 읽어들이고 buffer 단위로 읽을 수 있게 BufferedInputStream을 사용하며, 먼저 클라이언트에게 /img을 보내고 파일명을 보낸 후 파일 데이터를 전송함

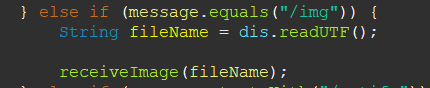


그림 19과정

클라이언트는 message.equals(“/img”)를 통해 해당 단계를 확인하고 receiveImage() 함수를 호출

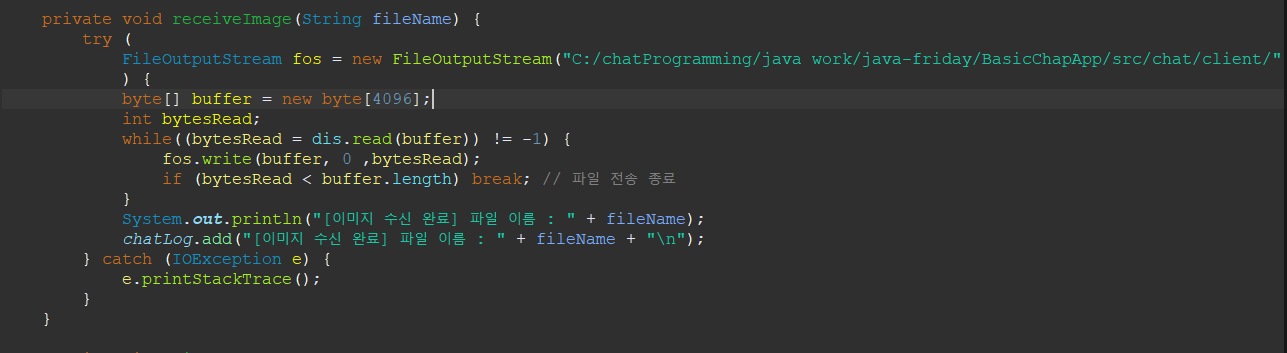


그림 20과정

receiveImage() 함수를 호출하면 해당 경로로 파일을 생성할 준비를 하고 dis(소켓 스트림)에서 읽고 파일을 지정 경로에 저장함

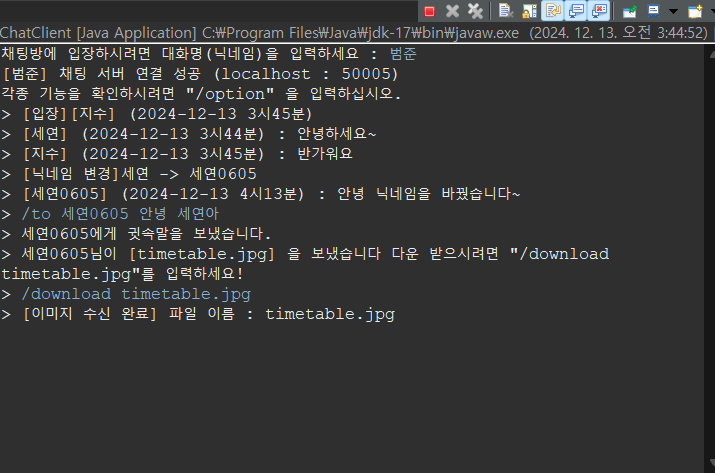


그림 21결과

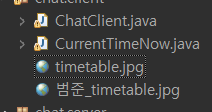


그림 22결과

범준 사용자는 세연 사용자가 보낸 파일을 확인하고 /download 파일명 을 입력하여 해당 파일을 다운로드 받음

- 2.1.7채팅 로그를 저장할 수 있습니다.

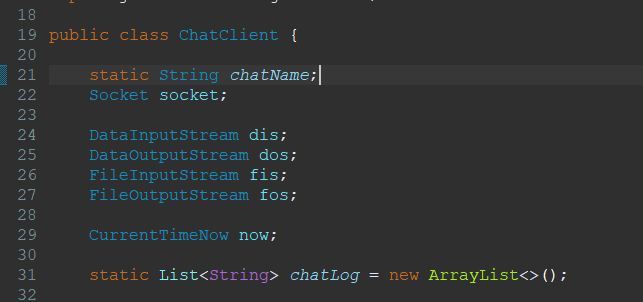


그림 23과정

클라이언트 클래스에서 static으로 지정하여 chatLog를 List로 지정

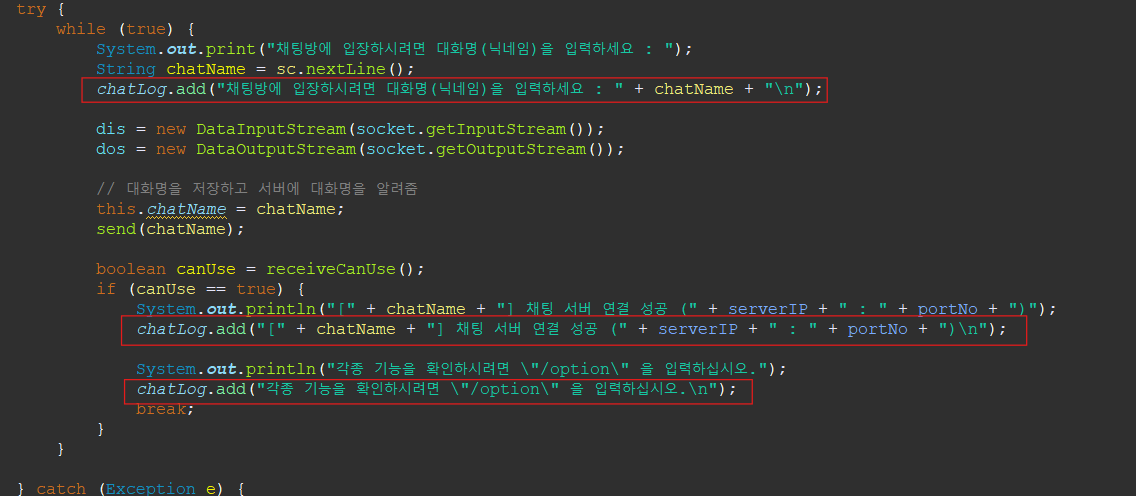


그림 24과정

클라이언트 클래스에서 출력할 때마다 chatLog에 add 시키는 방식

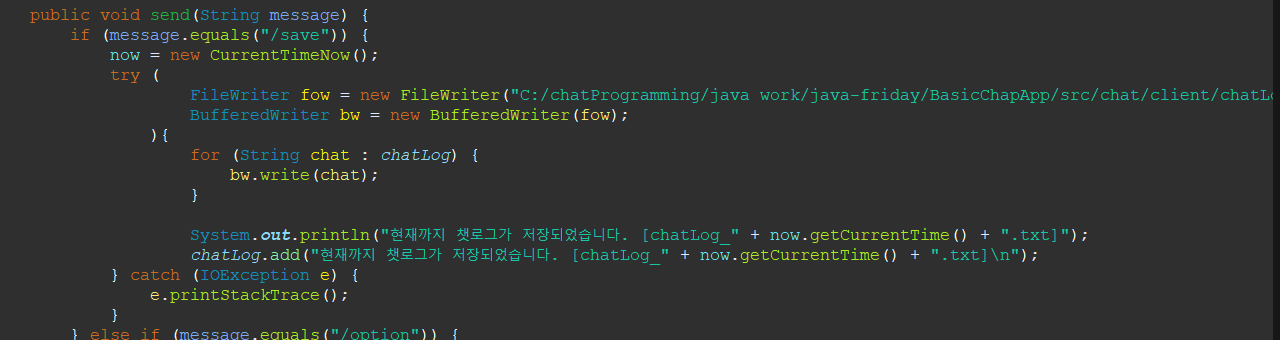


그림 25과정

클라이언트 클래스에서 메시지를 입력하고 send()를 호출할 때 /save 키워드를 검증하여 chatLog를 저장하는 형식

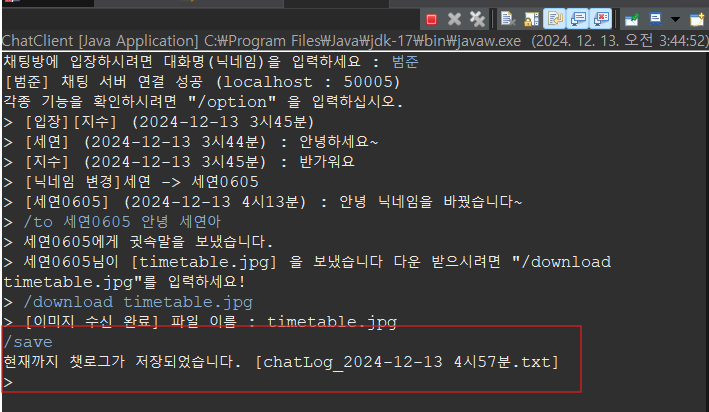


그림 26결과

범준 사용자는 지금까지 채팅 기록을 /save 로 txt형식으로 저장

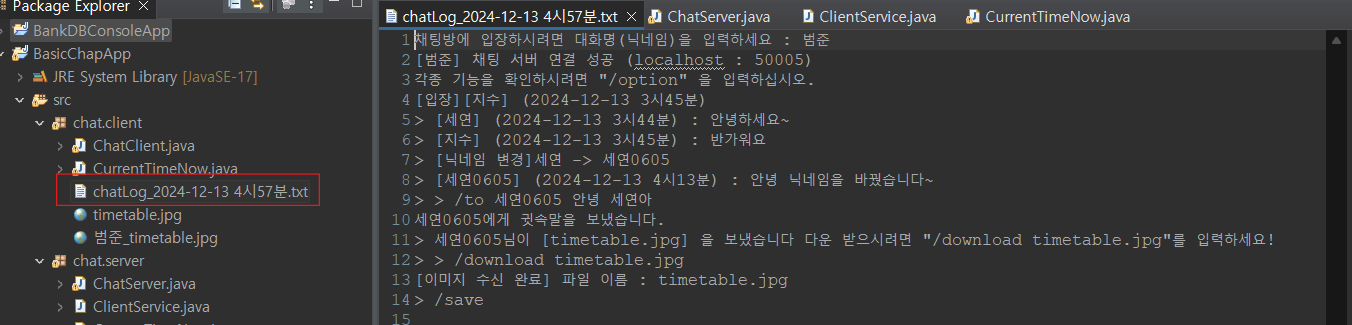


그림 27결과

chatlog\_저장시간.txt 파일이 저장됨

- 2.1.8메시지를 보낸 상대방이 오프라인일 경우 이를 알리는 기능을 제공합니다.

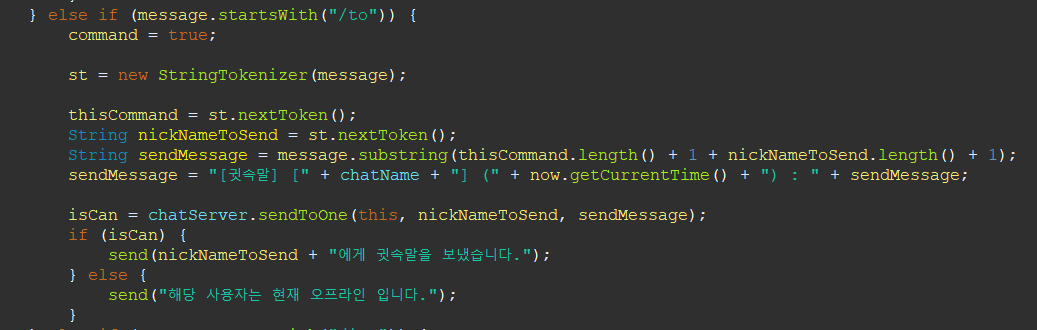


그림 28과정

메시지를 보낼 수 없는 상태, 즉 오프라인 상태에서는Hashtable에 보낼 닉네임의 사용자가 없다는 뜻이므로 isCan이 false일 경우 오프라인 메시지를 send()

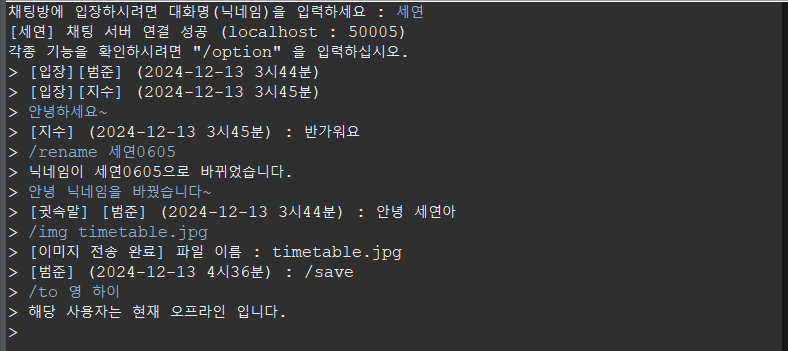


그림 29결과

영 사용자는 존재하지 않으므로 오프라인 메시지를 출력

- 2.1.9현재 메시지를 보낼 수 있는 상대방 목록을 확인할 수 있습니다.

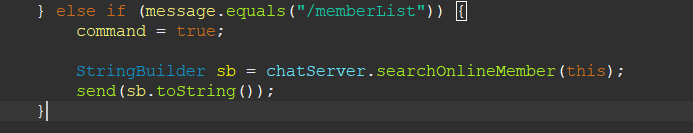


그림 30과정

chatServer에 구현된 searchOnlineMember() 함수 호출 (return 타입 : StringBuilder)

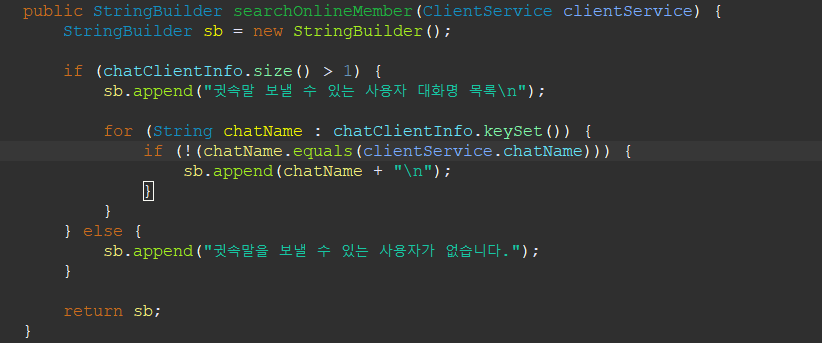


그림 31과정

만약 Hashtable의 크기가 1이라면 자신밖에 없는 상태이므로 size가 1보다 클 경우에만 순회하여 현재 등록되어 있는 클라이언트를 조회하고 해당 chatName(대화명)을 StringBuilder에 append()

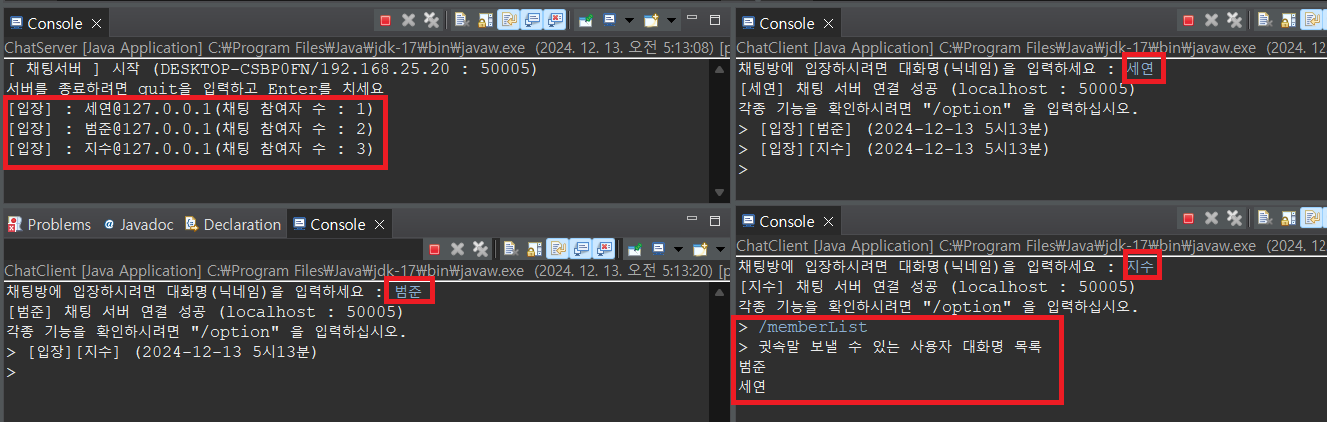


그림 32결과

지수 사용자 입장에서 본인을 제외하고 현재 세연, 범준 사용자만 있으므로 /memberList 입력 시 범준, 세연만 출력

3. 프로그램 설계 및 구현 3.1 시스템 구성

* 서버와 클라이언트 간 통신 구조로 설계되었습니다.
* 서버는 사용자 관리, 메시지 중계, 이미지 파일 처리, 로그 저장 등을 담당합니다.
* 클라이언트는 사용자 인터페이스 제공 및 서버와의 통신을 처리합니다.
* 3.2 주요 기능 상세
* 채팅 기능: 다수 사용자 간 실시간 채팅 가능.
* 닉네임 변경: /rename 명령어를 통해 닉네임 변경, 중복 닉네임 사용 불가.
* 멘션 기능: /to 명령어로 특정 사용자에게 메시지 전달.
* 이미지 전송: /img 명령어로 이미지 파일 전송 및 다운로드 기능 제공.
* 로그 저장: 채팅 기록을 파일로 저장.

4. 테스트 및 결과

- 4.1 테스트 환경

* 서버 환경: [서버 OS 및 스펙] – window11 / eclipse (JAVA)
* 클라이언트 환경: [클라이언트 OS 및 스펙] – window11 / eclipse (JAVA)
* 4.2 테스트 시나리오
* 다수 사용자 동시 채팅 테스트.
* 닉네임 변경 및 중복 처리 테스트.
* 멘션 및 이미지 전송 기능 테스트.
* 로그 저장 및 이미지 다운로드 기능 테스트.

4.3 테스트 결과

* [결과 요약: 성공/실패 및 주요 이슈]

5. 추가 기능 구현

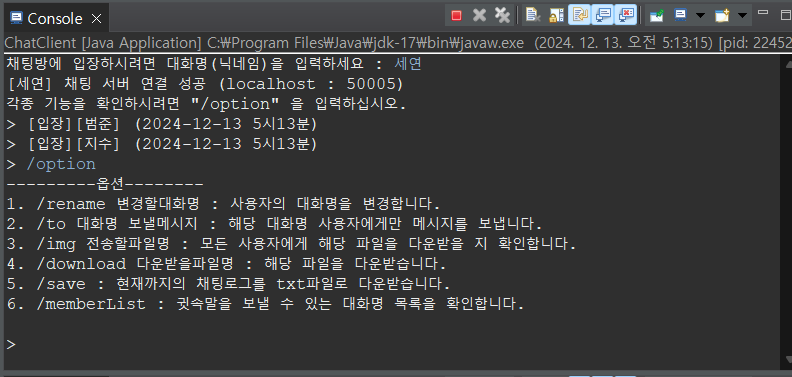


그림 33추가 기능

* /option 기능을 추가하여 사용자가 어떤 기능을 쓸 수 있는 지 쉽게 확인 가능합니다.