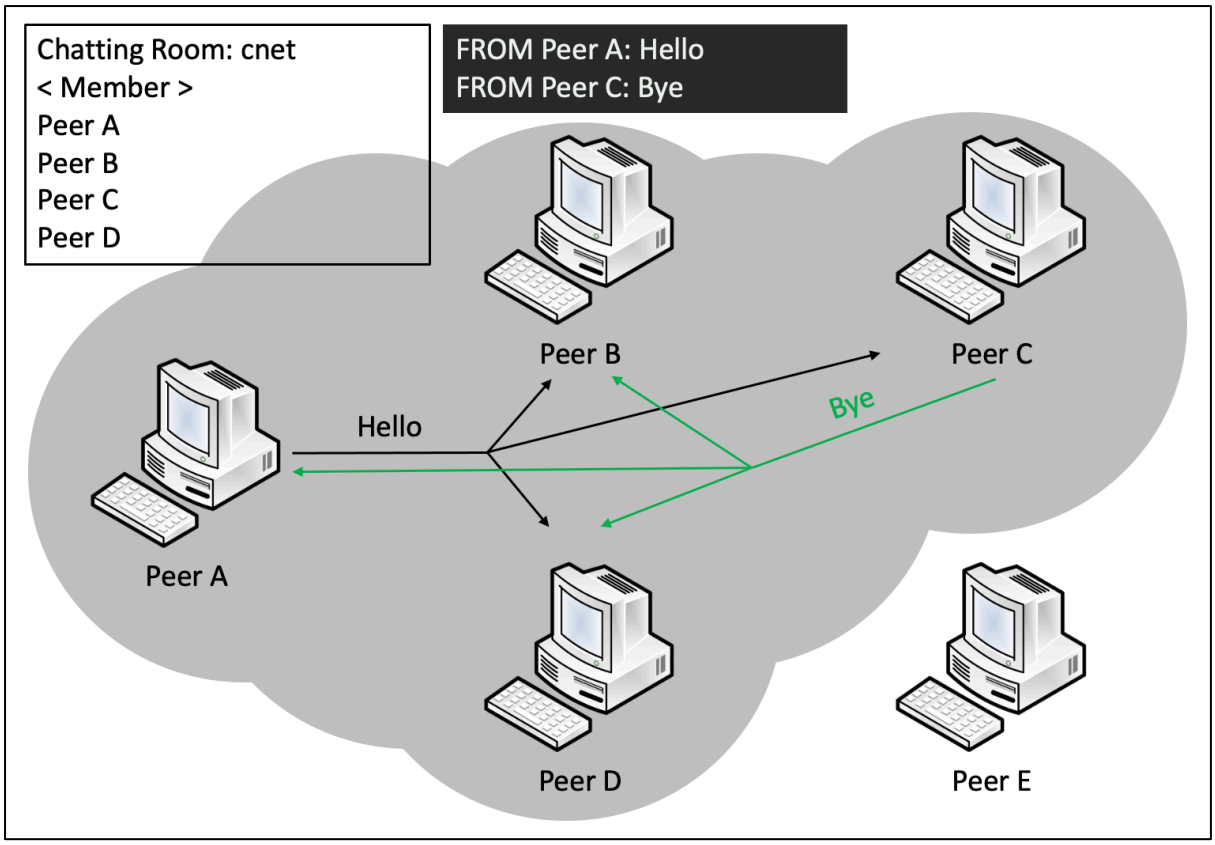
Computer Networks: Fall 2022
Programming Assignment

< 과제 기술 >

소켓 프로그래밍 API 에 대한 이해를 바탕으로 하여 본 문서에서 기술되는 방식으로 동작하는 채팅 프로그램을 구현한다. 문서에서 명시되지 않은 프로토콜의 세부적인 사항에 대해서는 자체적으로 결정할 수 있으나, 그 결정은 합리적이어야 하며 또한 구현 코드와 함께 제출할 디자인 및 구현 문서에 명시적으로 기술되어야 한다. 구현 프로그래밍 언어는 Java 로 한정됨을 유의한다.

➢ **Assignment #1** – UDP 의 Multicast 를 이용한 P2P 방식의 오픈 채팅 프로그램을 구현한다.

과제 1 에서는 P2P(Peer to Peer)방식의 오픈 채팅 프로그램을 구현한다. 채팅방에 참여한 피어 간 채팅은 Multicast 를 이용해서 해당 채팅방의 모든 구성원들에게 전달되도록 한다.



세부 구현 사항

1. 채팅 프로그램 실행

Peer	• 프로그램 실행 인자로 port number 를 입력 받는다. • 실행 예시: java Peer portNo

피어 프로그램 수행 시에 명령어 라인 인자로 multicast port number(portNo)를 지정한다. 피어는 프로그램이 실행되면 원하는 채팅방에 들어가거나 새로운 채팅방을 개설할 수 있다.
채팅 내에서 "#"으로 시작하는 문장은 아래에서 설명하는 동작을 수행하기 위한 용도로만, 즉 명령어로만, 사용됨 ("#"으로 시작하는 메시지는 전달할 수 없음).

2. 채팅 연결

Peer	• 채팅에 참여하기 위해서 #JOIN 명령어를 구현
	
	#JOIN (참여할 채팅방의 이름) (사용자 이름)
	• 채팅을 시작하는 Peer 는 네트워크 내 중복되지 않는 Multicast address 를 이용해서 새로운 채팅방을 생성 또는 기존 채팅방에 참여
	  • 포트 번호는 프로그램 실행에서 입력 받은 번호 사용
	  • Multicast address 는 225.0.0.0 ~ 225.255.255.255 범위
	  • 입력 받은 채팅방 이름을 "SHA-256" 해시를 이용해서 Multicast address 225.x.y.z 로 변환(x, y, z 값을 구함)
	  • java.security.MessageDigest 참고
	  • 해시 값(byte 배열)의 가장 마지막 3 개의 byte 를 이용해서 값을 구함
	  • 한정된 범위 내에서 다른 채팅방 이름 사이에 같은 Multicast address 가 발생할 가능성이 존재하지만 본 과제 구현에서는 무시함
	• 해당 채팅에 참여하고 싶은 Peer 는 채팅방 이름을 동일한 방법으로 구한 Multicast address 를 이용해서 채팅방에 참여할 수 있음
	• 사용자 이름은 메시지 시작 부분에 포함되어서 전달되어서, 수신한 피어가 메시지를 화면에 출력할 때 송신자를 알 수 있도록 한다.

3. 채팅

Peer	• Multicast 를 이용해서 채팅방 내에 있는 모든 Peer 에게 메시지를 전달 • 채팅 메시지를 chunk 단위(512 byte)로 나누어서 전송

출력 예시)
피어 B 는 "cnet" 채팅방에 조인했을 때 아래와 같이 피어 A 와 C 로부터의 메시지 수신을 화면에 출력한다. 아래의 출력 예는 피어 A 의 사용자 이름은 "PeerA"이고 피어 C 의 사용자 이름은 "PeerC"인 경우이다.

PeerA: Hello
PeerC: Bye

4. 채팅방 나가기

Peer	• 채팅방에서 나가기 위한 #EXIT 명령어를 구현 #EXIT • 현재 채팅방에서 떠남

구현 시 주의 사항

• Peer 는 채팅 메시지를 전송하는 동작과 채팅 메시지를 읽는 동작이 동시에 동작할 수 있도록 Thread 를 이용해서 구현해야 한다.

< 과제 제출 >

과제는 HYON LMS 에 구현 소스 코드와 함께 "디자인 및 구현 문서" 파일을 포함하여 제출한다. 이 파일에는 제출 과제를 수행하는 방법이 정확하게 설명되어야 하며, 기술된 방법에 따라 수행하였음에도 불구하고 수행이 되지 않는 경우에는 미제출로 간주한다. 제출 파일명을 Assignment #1 은 "Assignment1_학번_이름.[jar|zip]", Assignment #2 는 "Assignment2_학번_이름.[jar|zip]"으로 하여야 한다. java 파일을 반드시 포함해야 하며, class 파일 혹은 executable 파일이 포함되지 않도록 유의한다.

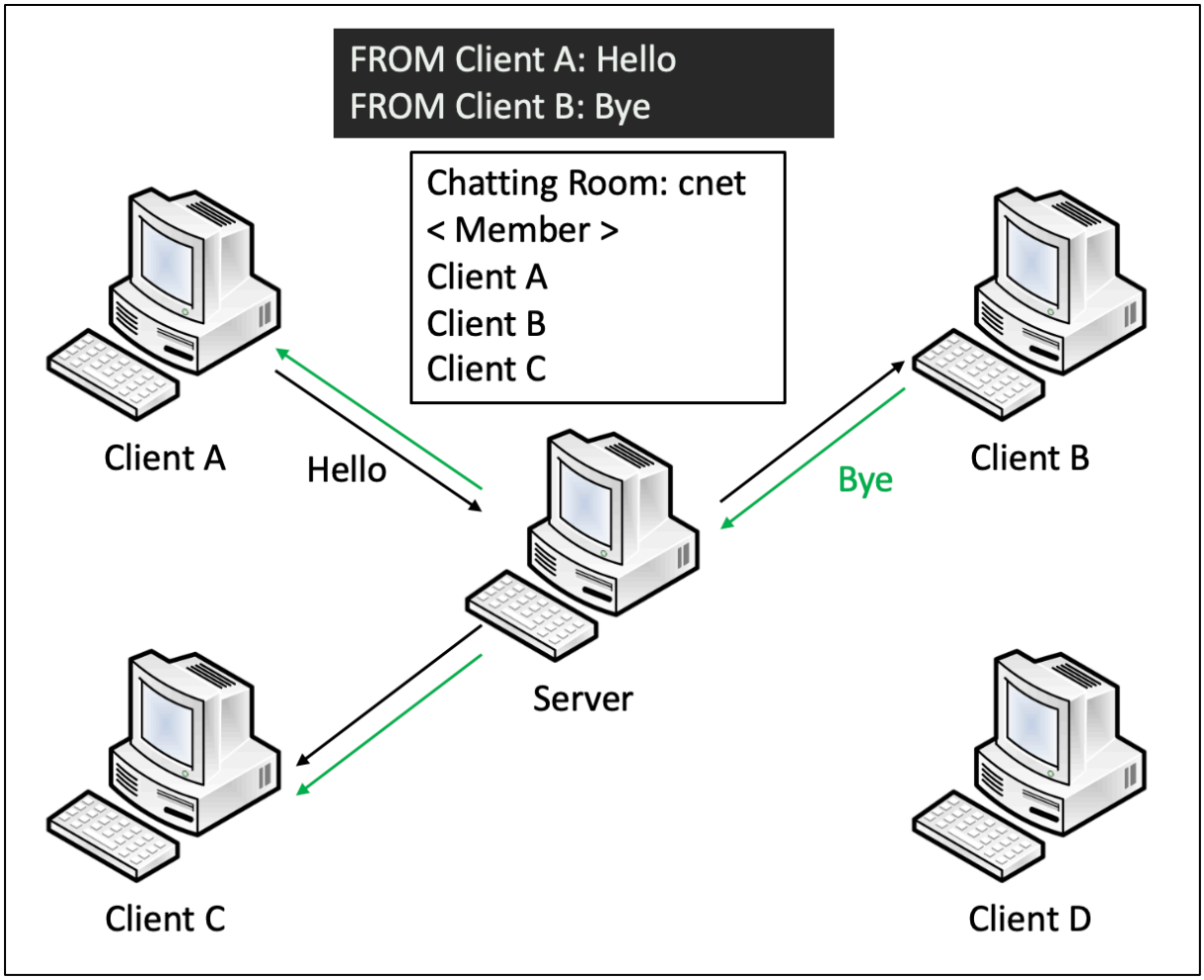
소스 코드 제출 시한은 Assignment #1 은 11 월 9 일 수요일 자정까지, Assignment #2 는 11 월 30 일 수요일 자정까지로 설정한다. 지연된 프로그램 제출의 감점은 제출 시한으로부터 24 시간 이내는 50% 감점, 48 시간 이내는 75% 감점이다. 제출 시한으로부터 48 시간 이후 제출은 점수가 주어지지 않는다.

디자인 및 구현 문서에서는 프로그램의 설계 및 구조의 기술 외에도 제출된 프로그램을 컴파일 및 실행하는 방법을 단계별로 정확하게 기술하여야 한다. 또한, 소스 코드를 단순히 Copy&Paste 하여 의미없는 문서를 제출하는 경우 감점의 요소가 됨을 유의하여야 한다. 프로그램 테스트 시 기술된 절차를 그대로 따라 했음에도 불구하고 프로그램이 수행되지 않는 경우에는 제출 프로그램이 동작하지 않는 것으로 간주한다. 문서의 마지막 부분에는 간단한 예시 실행 화면 캡처를 포함하도록 한다.

과제 제출 관련하여 이세정 조교(lshizheng199@gmail.com)에게 문의할 수 있다.

➢ **Assignment #2** – TCP 프로토콜을 이용한 클라이언트-서버 방식의 오픈 채팅 프로그램을 구현한다.

과제 2 에서는 클라이언트-서버 방식의 채팅 프로그램을 구현한다. 서버는 클라이언트 간의 채팅을 중계하는 역할을 한다. 서버는 클라이언트와 클라이언트의 연결을 수행하고, 메시지를 중계 전달한다.



세부 구현 사항

1. 채팅 프로그램 실행

Client	• 프로그램 실행 인자로 Server IP address, port number 1, port number 2 를 입력 받는다. • 실행 예시: java Client 127.0.0.1 2020 2021

Server	• 프로그램 실행 인자로 port number 1, port number 2 를 입력 받는다. • 실행 예시: java Server 2020 2021

Port number 1: 서버와 채팅 메시지를 주고 받는 용도로 사용되며, #로 시작하는 명령어 전송에도 이용된다.
Port number 2: #PUT, #GET 의 동작만을 위해 사용

클라이언트는 프로그램이 실행되면 채팅방의 이름을 통해서 새로운 채팅을 시작하거나, 다른 클라이언트가 생성한 채팅방에 참여할 수 있다.

채팅 내에서 “#”으로 시작하는 문장은 아래에서 설명하는 동작을 수행하기 위한 명령어 용도로만 사용 (“#”으로 시작하는 메시지는 전달할 수 없음)

2. 채팅방 생성 및 참여

Client	새로운 채팅을 시작하기 위한 #CREATE 명령어와 기존의 채팅에 참여하기 위한 #JOIN 명령어를 구현
	
	#CREATE (생성할 채팅방의 이름) (사용자 이름)
	• 생성할 채팅방의 이름을 인자로 채팅방의 생성을 요청
	  • 채팅방 생성에 성공하면 채팅을 시작할 수 있음
	
	#JOIN (참여할 채팅방의 이름) (사용자 이름)
	• 참여할 채팅방의 이름을 인자로 채팅방에 참여를 요청
	  • 채팅방 생성에 성공하면 채팅을 시작할 수 있음

Server	• 채팅방 생성 요청을 받으면 채팅방을 생성하고 중계를 수행
	  • 채팅방 생성에 성공하면 성공 메시지를 전송
	  • 이미 존재하는 채팅방은 실패 메시지를 전송
	• 채팅방 참여 요청을 받으면 채팅방에 추가하고 중계를 수행
	  • 채팅방 참여에 성공하면 성공 메시지를 전송
	  • 존재하지 않는 채팅방은 실패 메시지를 전송

클라이언트는 한 번에 하나의 채팅방에 참여하여 채팅을 진행한다. 하지만, 서버는 복수 개의 채팅방을 동시에 지원할 수 있어야 한다.

3. 채팅

Client (송신자)	• 송신자는 채팅방에 전달하고자 하는 메시지를 서버로 전송

Client (수신자)	• 채팅방에 참여 중인 송신자를 제외한 모든 클라이언트는 서버로부터 메시지를 받음

Server	• 채팅에서 송신자의 메시지를 받아 모든 수신자에게 메시지를 중계 전달

출력 예시)
FROM ClientA: Hello
FROM ClientB: Bye

4. 파일 송수신

Client	채팅 내에서 파일을 주고 받기 위해서 #PUT 명령어와 #GET 명령어를 구현 파일 송수신은 프로그램 실행 시 입력 받은 port number 2 의 포트 번호를 이용해서 별도의 TCP 연결을 만들어 수행. 이때 클라이언트 화면에 파일 송수신 진척도를 표시하기 위해 64Kbyte 당 하나의 “#” 문자를 출력하도록 한다. #PUT (FileName) • 파일의 송신자는 #PUT 명령어를 이용해서 Server 로 파일을 전송 #GET (FileName) • 파일의 수신자는 Server 로부터 전달 받은 파일 이름을 통해 #GET 명령어를 수행하고 Server 로부터 파일을 다운로드

Server	• 파일을 송신하는 Client 로부터 #PUT 명령어를 통해 파일을 Server 로 전달받음 • 파일을 수신하는 Client 로부터 #GET 명령어를 통해 파일을 Client 로 전달

5. 채팅방 나가기

Client	채팅방에서 나가기 위한 #EXIT 명령어를 구현 #EXIT • 현재 채팅방에서 떠남

Server	• 채팅방을 떠난 클라이언트는 채팅방 멤버에서 제거

6. 채팅방 상태 확인

Client	채팅 구성원 확인을 위한 #STATUS 명령어를 구현 #STATUS • 현재 채팅방의 이름, 구성원의 정보(이름)를 출력

Server	• 채팅방의 이름, 구성원의 정보(이름)를 전달

구현 시 주의 사항

- Server 가 한 시점에서 복수 개의 채팅방의 요청을 처리하기 위해서 Thread 를 이용해서 구현한다.

< 예시 >
Client A, Clinet B, Client C 가 포함된 ChatRoom X 에서 대화가 진행되는 동시에,
Client D, Client E, Client F 가 포함된 ChatRoom Y 에서의 대화도 진행되어야 한다.

선택적 구현 사항

- 다음의 내용을 구현하면 추가 점수가 부여된다. 이 경우 코드를 "Assignment2_NIO_학번_이름.[jar|zip]"으로 추가 제출하여야 한다.
  - 과제 2 의 기본 구현에서는 thread 를 이용하여 채팅방을 구현하는 것이나, 이 선택 구현에서는 single thread 에 mon-blocking IO 를 이용해서 multiple chat room 을 구현한다. 이를 위해 아래의 class 를 사용한다.
    - java.nio.channels.Selector
    - java.nio.channels.ServerSocketChannel
    - java.nio.channels.SocketChannel
    - java.nio.channels.SelectionKey
    - java.nio.ByteBuffer

< 과제 제출 >

과제는 HYON LMS 에 구현 소스 코드와 함께 “디자인 및 구현 문서” 파일을 포함하여 제출한다. 이 파일에는 제출 과제를 수행하는 방법이 정확하게 설명되어야 하며, 기술된 방법에 따라 수행하였음에도 불구하고 수행이 되지 않는 경우에는 미제출로 간주한다. 제출 파일명을 Assignment #1 은 “Assignment1_학번_이름.[jar|zip]”, Assignment #2 는 “Assignment2_학번_이름.[jar|zip]”으로 하여야 한다. java 파일을 반드시 포함해야 하며, class 파일 혹은 executable 파일이 포함되지 않도록 유의한다.

소스 코드 제출 시한은 Assignment #1 은 11 월 9 일 수요일 자정까지, Assignment #2 는 11 월 30 일 수요일 자정까지로 설정한다. 지연된 프로그램 제출의 감점은 제출 시한으로부터 24 시간 이내는 50% 감점, 48 시간 이내는 75% 감점이다. 제출 시한으로부터 48 시간 이후 제출은 점수가 주어지지 않는다.

디자인 및 구현 문서에서는 프로그램의 설계 및 구조의 기술 외에도 제출된 프로그램을 컴파일 및 실행하는 방법을 단계별로 정확하게 기술하여야 한다. 또한, 소스 코드를 단순히 Copy&Paste 하여 의미없는 문서를 제출하는 경우 감점의 요소가 됨을 유의하여야 한다. 프로그램 테스트 시 기술된 절차를 그대로 따라 했음에도 불구하고 프로그램이 수행되지 않는 경우에는 제출 프로그램이 동작하지 않는 것으로 간주한다. 문서의 마지막 부분에는 간단한 예시 실행 화면 캡처를 포함하도록 한다.

과제 제출 관련하여 이세정 조교(lshizheng199@gmail.com)에게 문의할 수 있다.