Network Omok

김도형

목차

1. 개요

2. 사용자 요구사항

3. 화면 설명서

4. 서버 클라이언트 통신

5. Gson 사용

개발 목표

TCP/IP 통신을 이용하여 사용자 간에 오목게임을 가능하게 하는 소프트웨어 개발

개발 환경

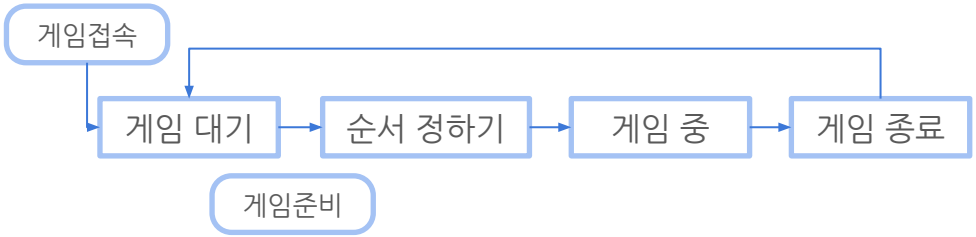
- Java jdk 1.8
- Processing API Library 3.5.3

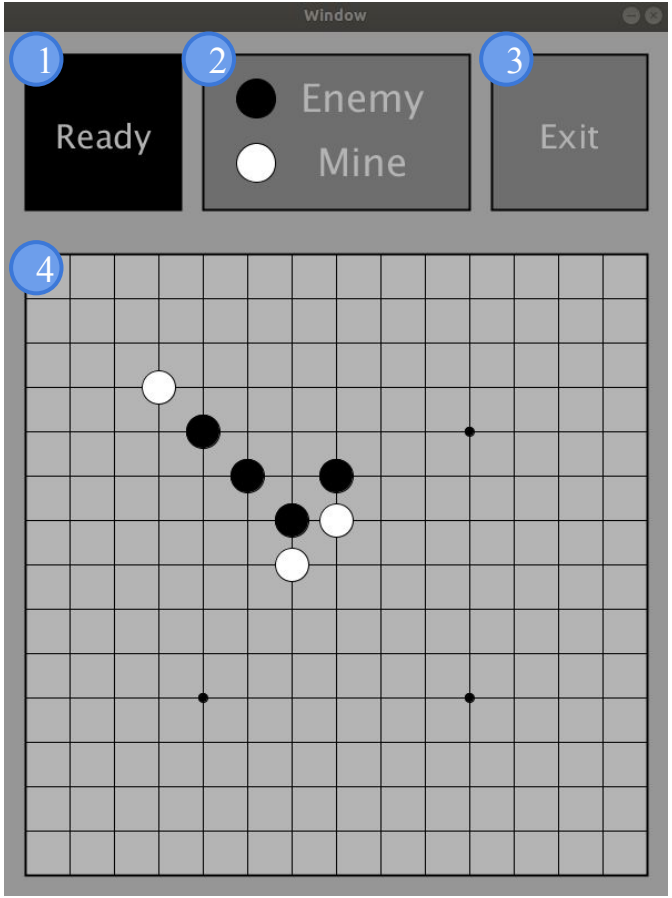
화면 설명서

와이어 프레임

1. “Ready” 버튼
2. 사용자 상태 창
3. “Exit” 버튼
4. 전체 오목판

플로우 차트





- **게임 중의 화면이고, 실제로 구현한 화면입니다.**

상세 화면 설명서 1

사용자

- 게임에 접속할 수 있다.

설명

- 사용자가 접속하면 Mine 이라는 문구가 나오고, 상대방이 접속하면 Enemy 이라는 문구가 나온다. 먼저 접속한 사용자의 문구가 위에 위치한다.
- 주사위를 던지기 전 단계이므로 플레이어의 오목돌 색은 아직 표시되지 않는다.

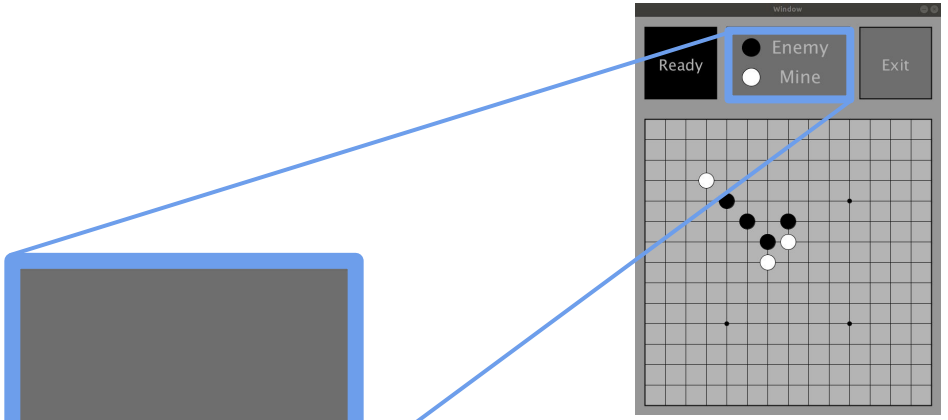


그림1. 전체화면 와이어프레임

사용자 접속하기 전

사용자 접속한 후

상세 화면 설명서 2

사용자

- 게임 준비를 할 수 있다.

설명

- 사용자는 접속한 후에 Ready 버튼을 누를 수 있다.
- Ready 버튼을 누르면 색이 변하고 (회색 => 검은색) 준비완료 상태가 된다.

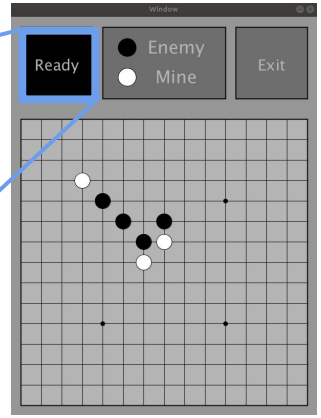


그림1. 전체화면 와이어프레임

Ready 버튼을 누르기 전

Ready 버튼을 누른 후

상세 화면 설명서 3

사용자

- 카운트 다운이 수행된다.

설명

- 준비 완료를 모든 사용자가 했다면, 게임이 시작되고 카운트 다운이 수행된다.
- 카운트 다운은 3, 2 ,1 순으로 수행된다.

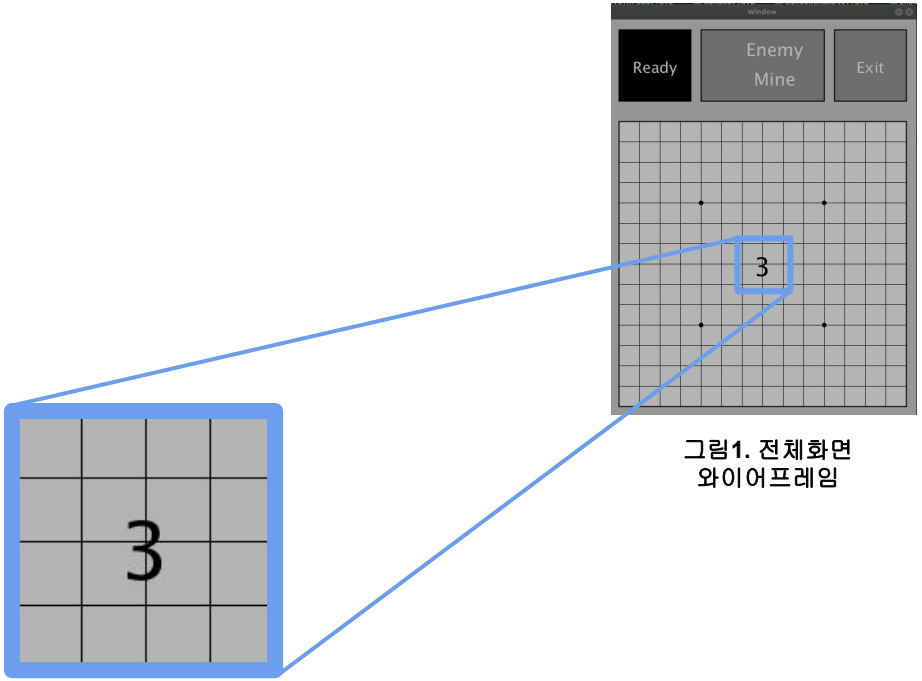


그림1. 전체화면 와이어프레임

상세 화면 설명서 4

사용자

- 주사위를 굴려 큰 수에 따라 순서를 정할 수 있다.

설명

- 카운트다운이 모두 수행되면 주사위를 굴린다.
- 주사위 숫자에 따라 흑 / 백이 결정된다. (흑이 선, 백이 후.)
- 플레이어들의 오목돌 색이 사용자 상태창에 표시된다.

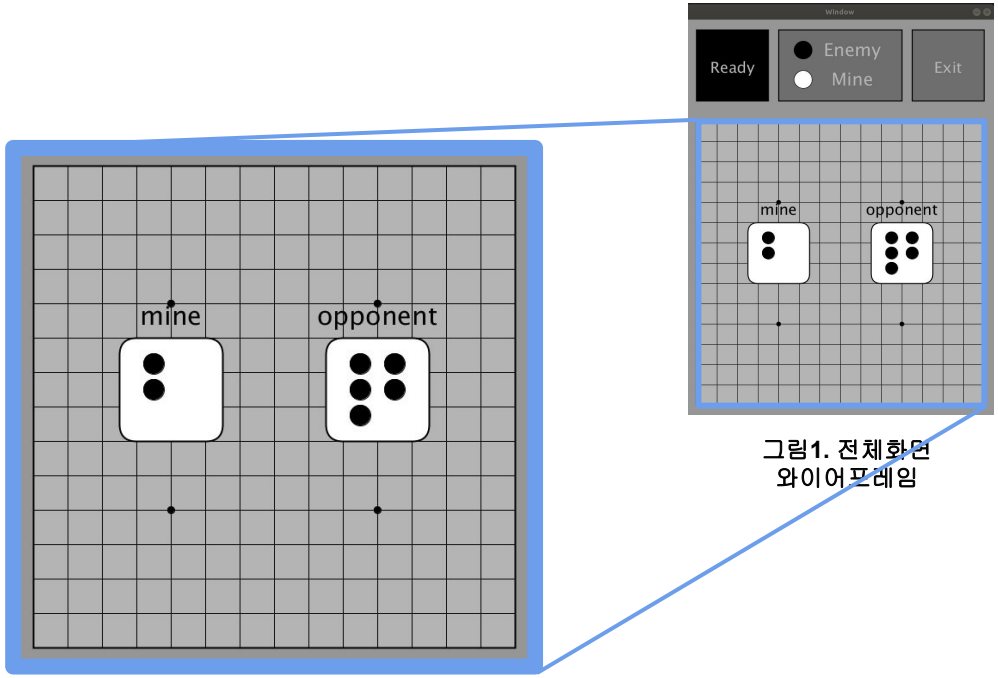
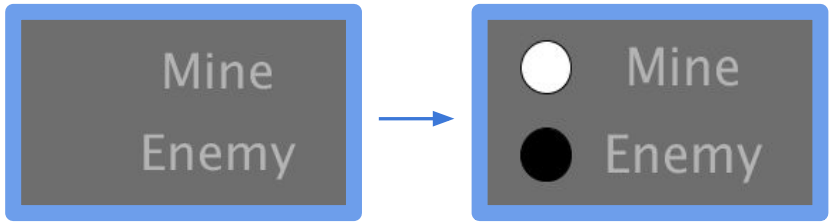


그림1. 전체화면 와이어프레임



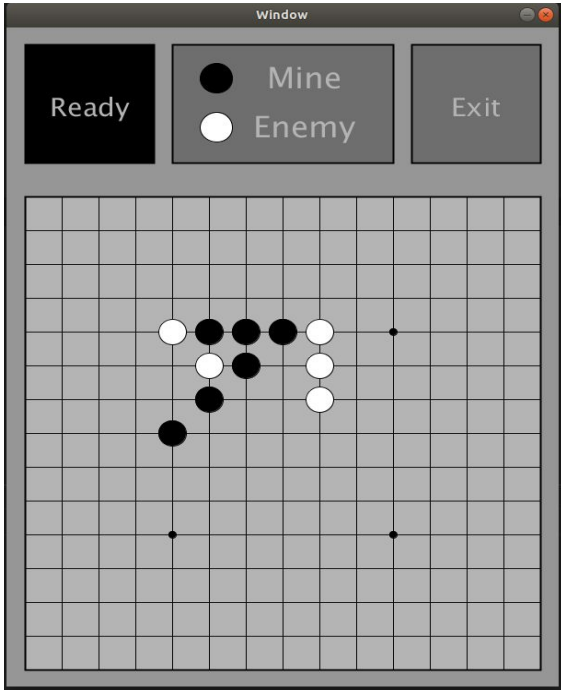
상세 화면 설명서 5

사용자

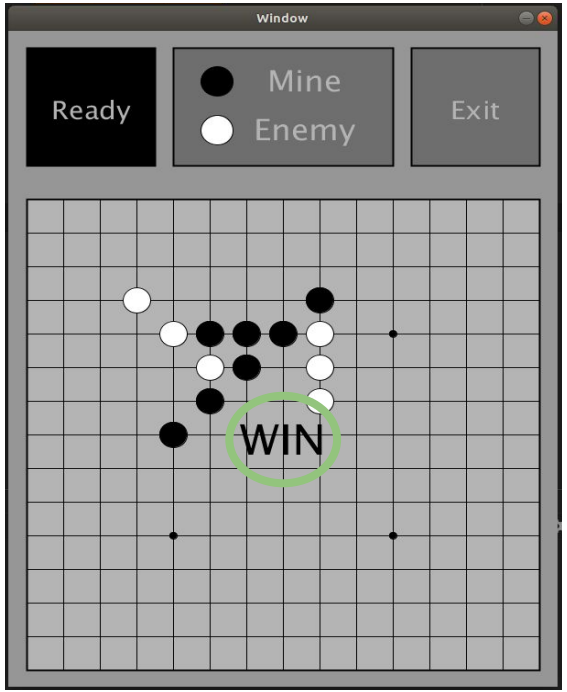
- 오목을 둘 수 있다.

설명

- 사용자는 원하는 위치에 오목돌을 둘 수 있다.
- 오목을 완성하면 승리 메세지(WIN)를 , 패배하면 패배메시지(LOSE)를 띄워준다.
- 게임이 끝나면 다시 Ready 버튼을 누를 수 있다.



게임 중



게임 종료

상세 화면 설명서 6

사용자

- 게임을 나갈 수 있다.

설명

- 게임이 종료되면 게임 종료 상태에서 3초 뒤에 게임 대기 상태로 돌아온다.
- Exit 버튼은 게임 대기 상태에서만 누를 수 있기 때문에 게임 시작 전이거나 게임이 끝나야만 게임에서 나갈 수 있다.
- 상대방이 게임을 나가면 사용자 상태창에 상대방의 문구(Enemy)가 사라진다.

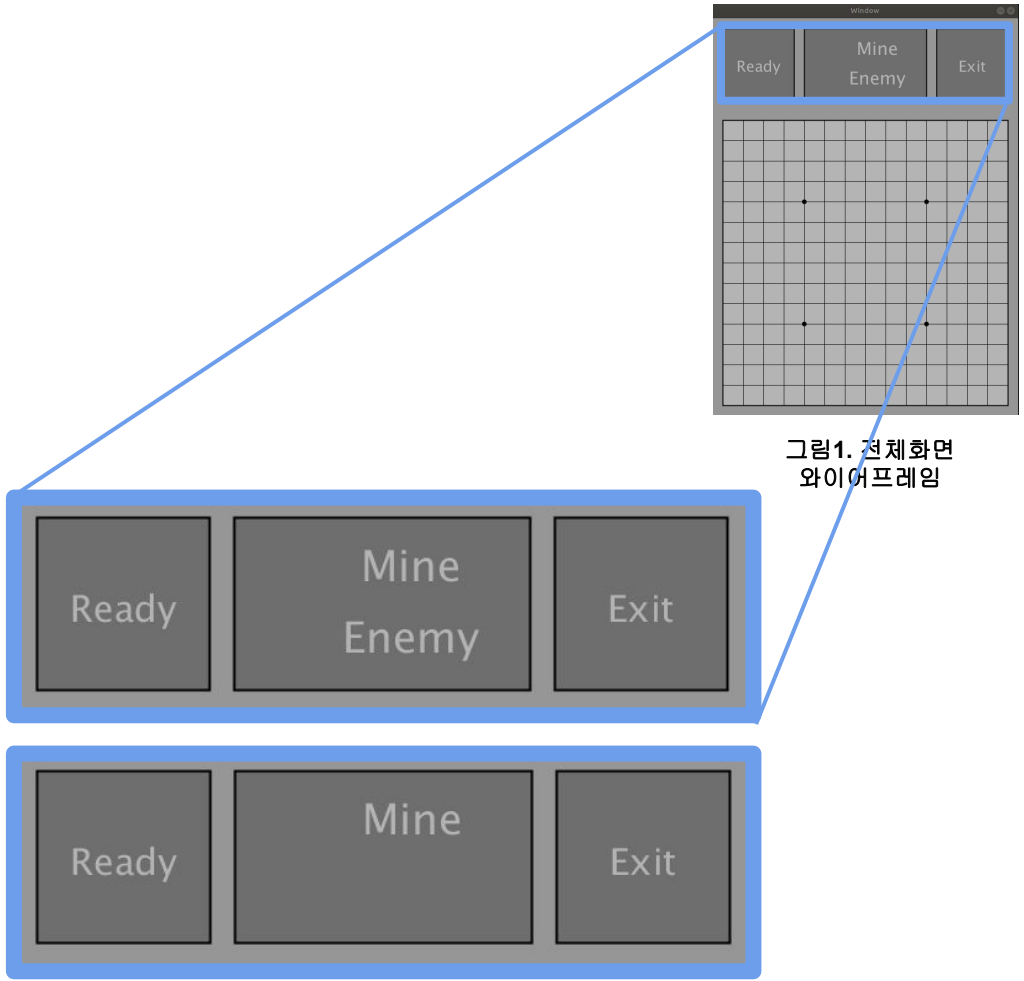
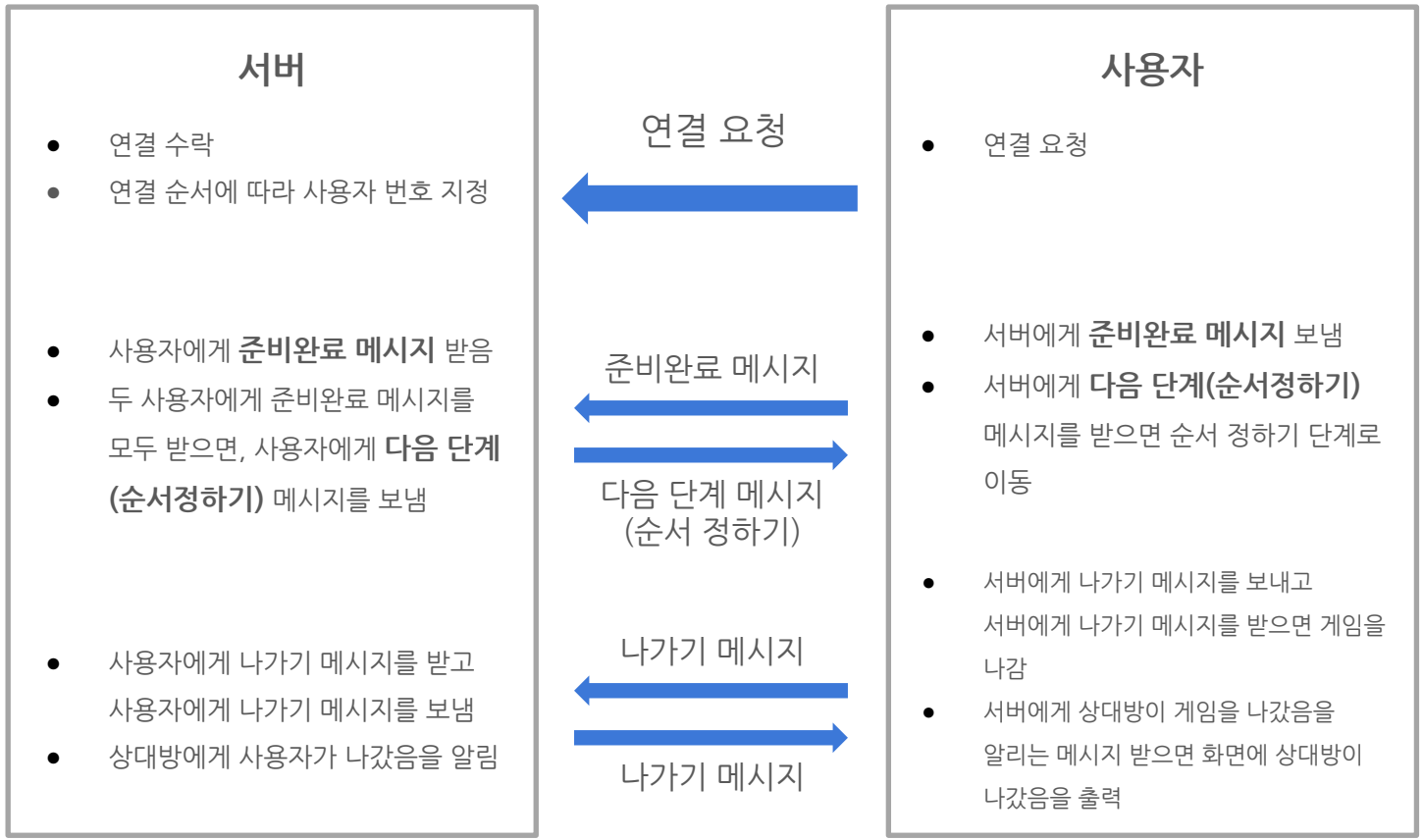


그림1. 전체화면 와이어프레임

게임 대기


서버
연결 수락
연결 순서에 따라 사용자 번호 지정
사용자에게 준비완료 메시지 받음
두 사용자에게 준비완료 메시지를 모두 받으면, 사용자에게 다음 단계 (순서정하기) 메시지를 보냄
사용자에게 나가기 메시지를 받고 사용자에게 나가기 메시지를 보냄
상대방에게 사용자가 나갔음을 알림
연결 요청
준비완료 메시지
다음 단계 메시지 (순서 정하기)
나가기 메시지
나가기 메시지
사용자
연결 요청
서버에게 준비완료 메시지 보냄
서버에게 다음 단계(순서정하기) 메시지를 받으면 순서 정하기 단계로 이동
서버에게 나가기 메시지를 보내고 서버에게 나가기 메시지를 받으면 게임을 나감
서버에게 상대방이 게임을 나갔음을 알리는 메시지 받으면 화면에 상대방이 나갔음을 출력

순서 정하기

서버

- **카운트 다운 번호**를 3부터 시작해서 1초마다 사용자에게 보냄 (3 => 2 => 1)
- 두 사용자의 **주사위의 값**을 각각 구하고(주사위의 값이 서로 같으면 다시 구함), 각각의 사용자에게 사용자와 상대방의 주사위 값 모두 보냄
- 두 사용자에게 **다음 단계(게임 중)** 메시지를 보냄

카운트 다운 →

주사위 값 (사용자, 상대방 각각) →

다음 단계 메시지 (게임 중) →

사용자

- 카운트 다운 번호를 서버에게 받는대로 오목판 중앙에 출력
- 서버에게 자신의 주사위 값과 상대방의 주사위값을 받고 값에 따라 자신의 **순서 결정**하고 사용자 상태창에 **바둑돌 색 표시** (선 - 흑돌 , 후 - 백돌)
- 서버에게 **다음 단계(게임 중)** 메시지를 받으면 게임 중 단계로 이동

서버 클라이언트 통신

게임 중

서버

- 사용자의 턴임을 알리는 메시지를 사용자에게 보냄
- 사용자에게 사용자가 놓은 돌의 위치 값을 받고 기록하고 상대방에게 사용자의 돌 위치 값을 보냄
- 돌의 위치값을 받을 때마다 서버에서 오목이 완성됐는지 검사

- 상대방의 턴임을 알리는 메시지를 사용자에게 보냄
- 사용자에게 상대방이 놓은 돌의 위치 값을 보냄

- 서버가 오목 검사를 하여, 오목이 완성됐으면 승리한 사용자에게 승리 메시지를 보냄. 만약 상대방이 승리했다면 사용자에게 패배 메시지를 보냄
- 두 사용자에게 **다음 단계(게임 종료)** 메시지를 보냄

사용자 턴임을 알림

놓은 돌의 위치값

상대방 턴임을 알림

상대방이 놓은 돌의 위치값

승리 및 패배 메시지

다음 단계 메시지 (게임 종료)

사용자

- 서버에게 자신의 턴임을 알리는 메시지를 받음
- 돌을 두고 위치 값을 기록
- 서버에게 돌의 위치 값을 보냄

- 서버에게 상대방의 턴임을 알리는 메세지를 받음
- 서버에게 상대방이 놓은 돌의 위치 값을 받고 기록

- 서버에게 승리 및 패배 메시지를 받으면 자신의 승패를 기록
- 서버에게 **다음 단계(게임 종료)** 메시지를 받으면 게임 종료 단계로 이동

게임 종료

서버

- 서버에서의 사용자들의 모든 정보를 초기화.
- 3초가 지나면 모든 사용자에게 **다음 단계(게임 대기)** 메시지를 보냄

다음 단계 메시지
(게임 대기)

사용자

- 게임 종료 상태가 되면 놓은 오목돌의 정보를 초기화
- 자신의 승패 기록에 따라 화면에 텍스트 출력(승 -> WIN , 패 -> LOSE 출력)
- **다음 단계(게임 대기)** 메시지를 받으면 게임 대기 상태로 이동 후 초기화 해야 될 정보들 모두 초기화

- 사용자에게 나가기 메시지를 받고 사용자에게 나가기 메시지를 보냄
- 상대방에게 사용자가 나갔음을 알림

나가기 메시지

나가기 메시지

- 서버에게 나가기 메시지를 보내고 서버에게 나가기 메시지를 받으면 게임을 나감
- 서버에게 상대방이 게임을 나갔음을 알리는 메시지 받으면 화면에 상대방이 나갔음을 출력

Gson 사용

```
private void sendReady() {
   ReadyData ready = new ReadyData(ReadyData.READY);
   Gson gson = new Gson();

   String data = gson.toJson(ready);
   String type = ConstantProtocol.READY;

   sendToServer(data, type);
}

private void sendToServer(String data, String type) {

       try {
           OutputStream os = socket.getOutputStream();
           DataOutputStream dos = new DataOutputStream(os);

           Gson gson = new Gson();

           Protocol protocol = new Protocol(data, type);
           String json = gson.toJson(protocol);

           int len = json.length();

           dos.writeInt(len);
           os.write(json.getBytes());

       } catch (IOException e) {
           e.printStackTrace();
       }
   }
}
```

=> 위의 코드는 클라이언트에서 **Ready 데이터**를 보내는 코드입니다.

=> 이와 같은 방식으로 주고 받을 데이터들을 **Gson을 이용해 Json 형식으로 데이터를 전송하였습니다.** 보내야 할 데이터 타입이 다양하여 각각의 데이터들을 클래스화 하였고 이 정보들을 ConstantProtocol에 static final String 필드들로 구별하였습니다.