다빈치 코드 게임

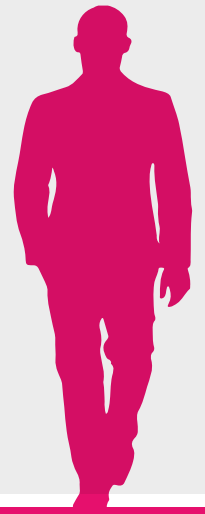
[TEAM 7] 정희승 김선우 박동현 정현재 주윤지

CONTENTS

목표 & 게임 설명

프로젝트 목표

- UML을 이용한 올바른 설계
- MVC구조를 이해하고 이에 맞춘 설계&구현
- 소켓 프로그래밍을 이용한 서버 & 채팅 연동
- 다양한 기능의 GUI 응용
- Database의 이해와 활용
- 이외 기능... TTS 활용



게임 설명

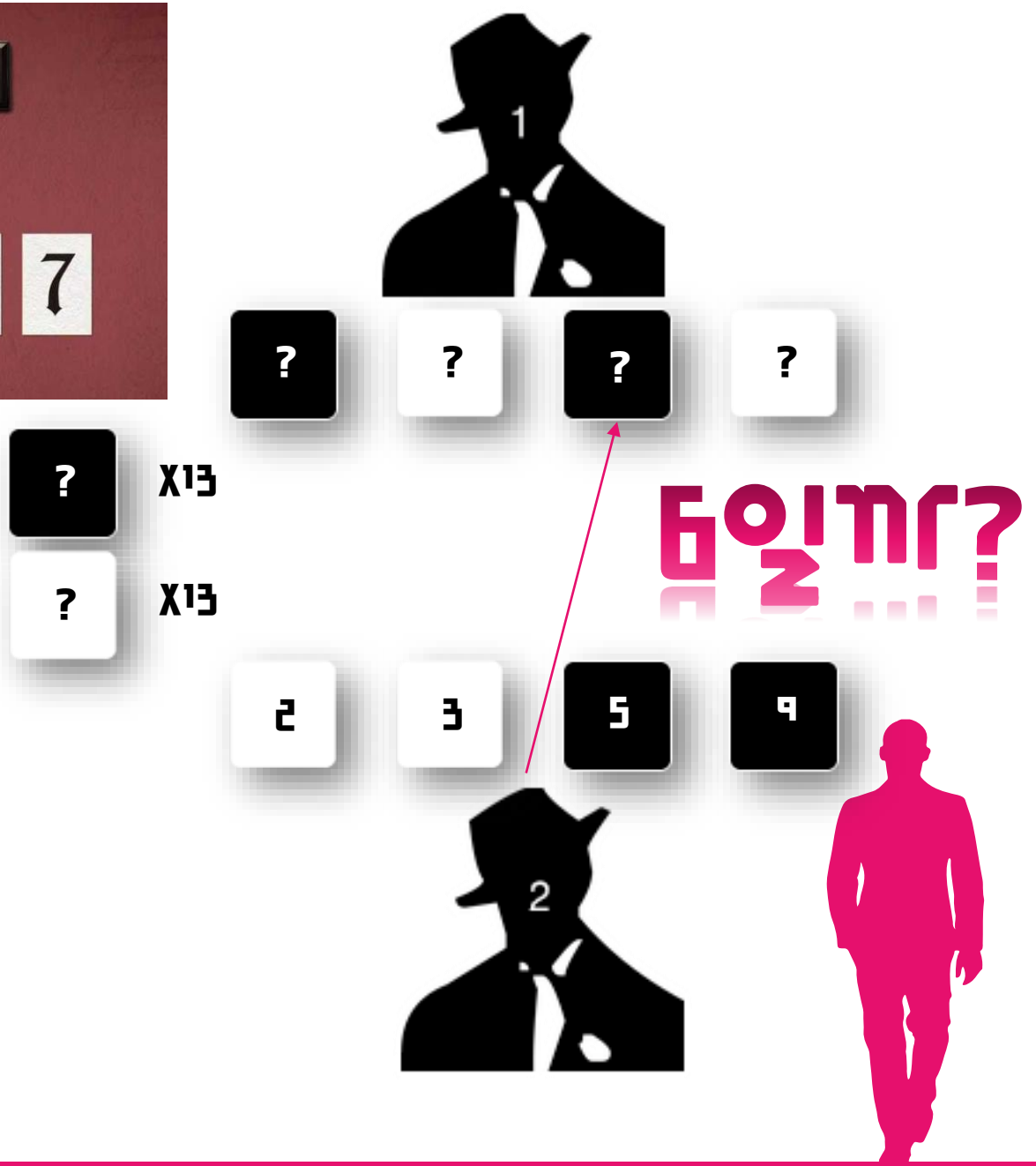
1. 2~4인 진행

2. 검은/하얀 불족 각각 13개씩 존재
   (숫자 12개, 랜덤으로 놓아지는 조커 1개)

3. 2~3인 진행은 4개, 4인 진행은 3개씩 받고 시작 때 턴 불족을 1개씩 받음

4. 각자 받은 불족의 숫자를 타인에게 비공개
   (불족을 왼쪽부터 작은 숫자 순으로 정렬)

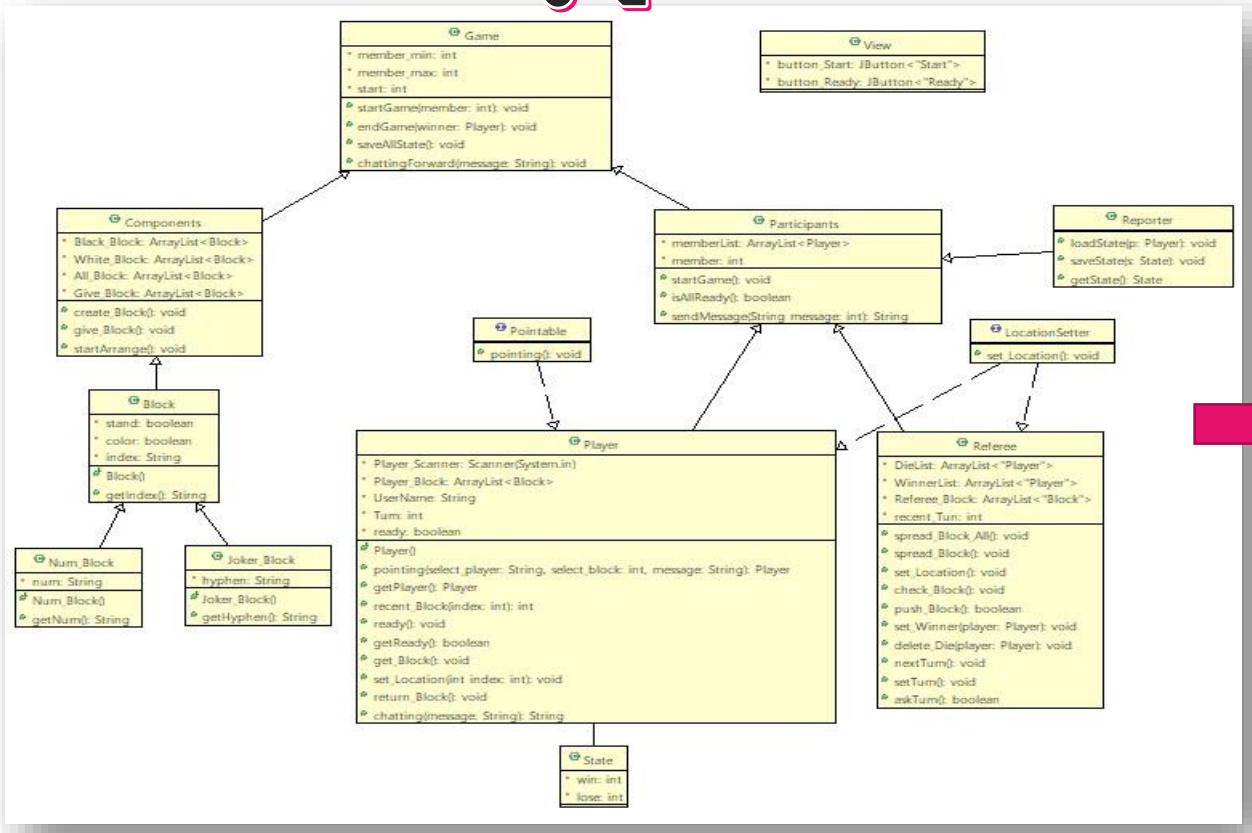
5. 상대의 불족을 선택하고 임의의 숫자를 유추
   맞추면 숫자가 공개되고 틀리면 턴 넘어감



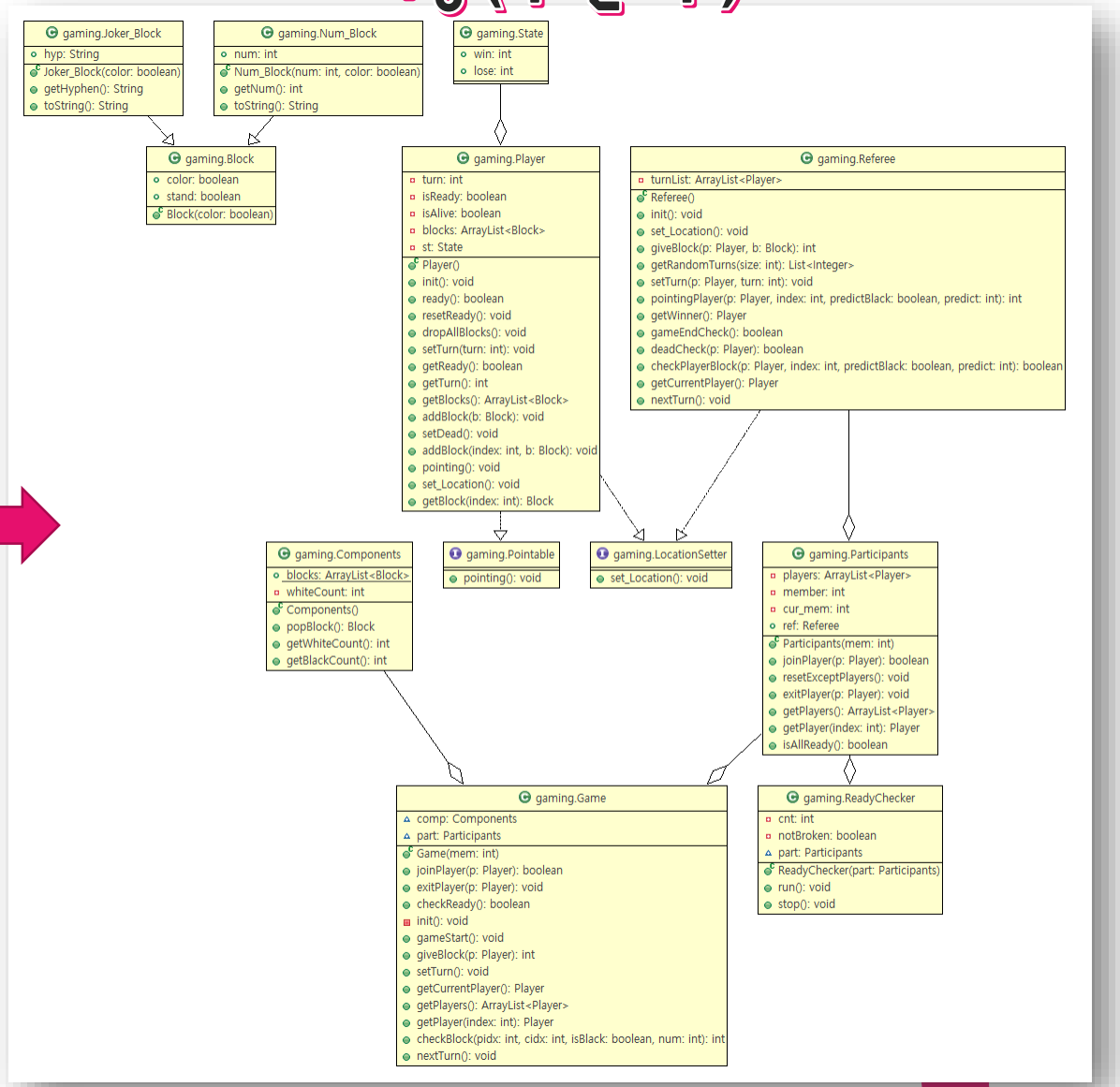
설계 및 기능 분석

설계 : 클래스 다이어그램(게임)

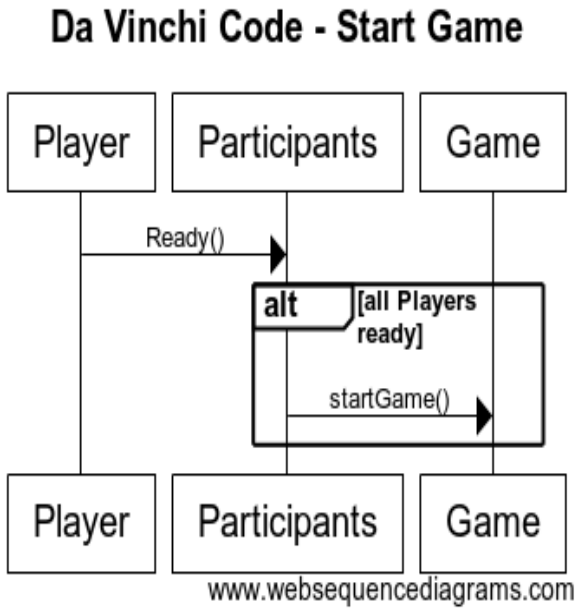
중간



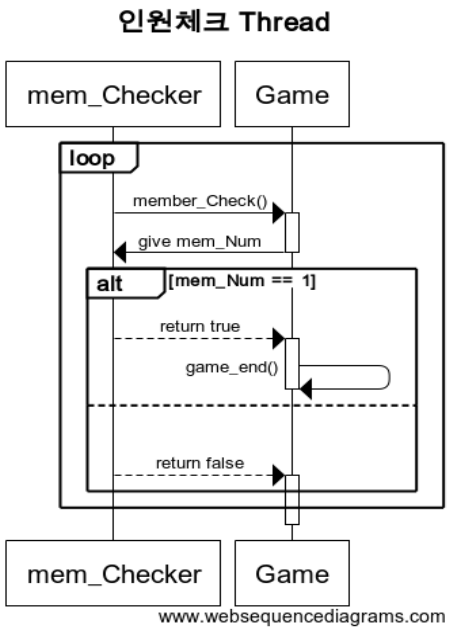
최종(구현 후)



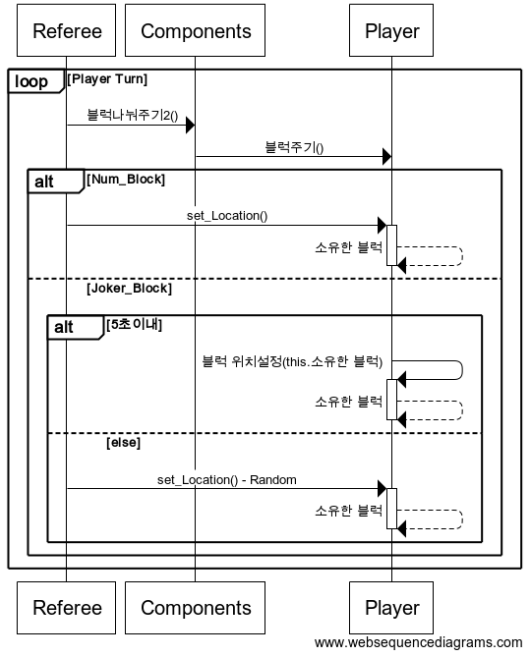
설계 : 시퀀스 다이어그램(게임)



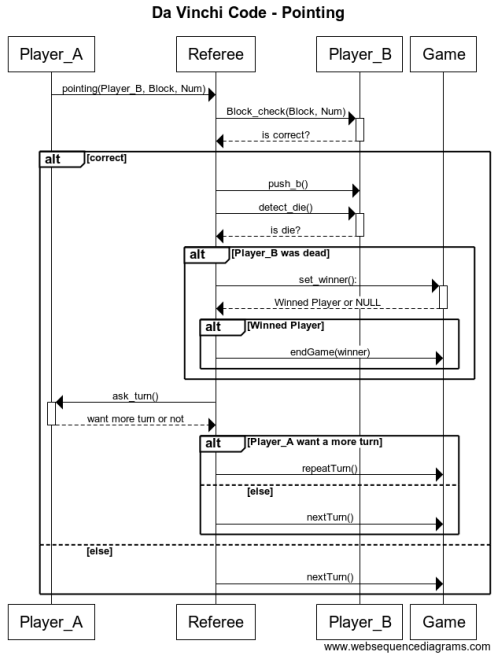
게임 시작



인원 체크

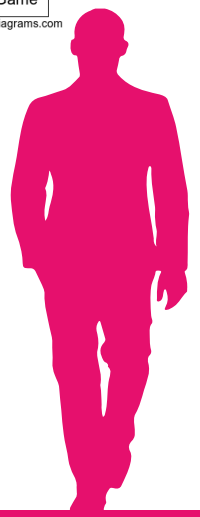


카드 분배



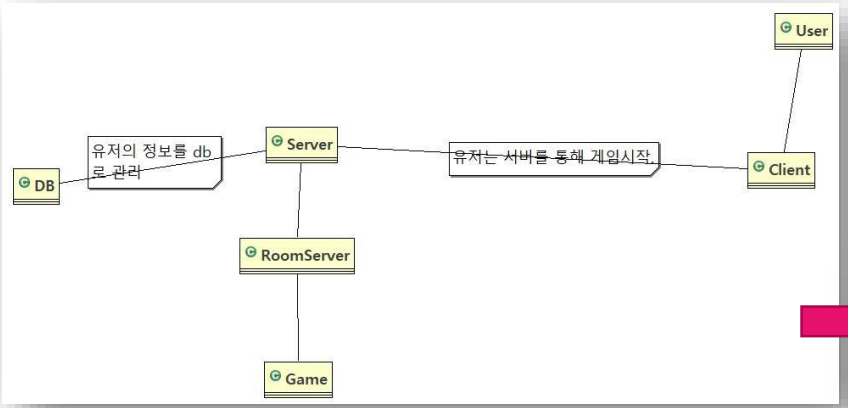
게임 과정

중간 점검때 최대한 분할 시켰던 시퀀스 다이어그램을 그대로 이용

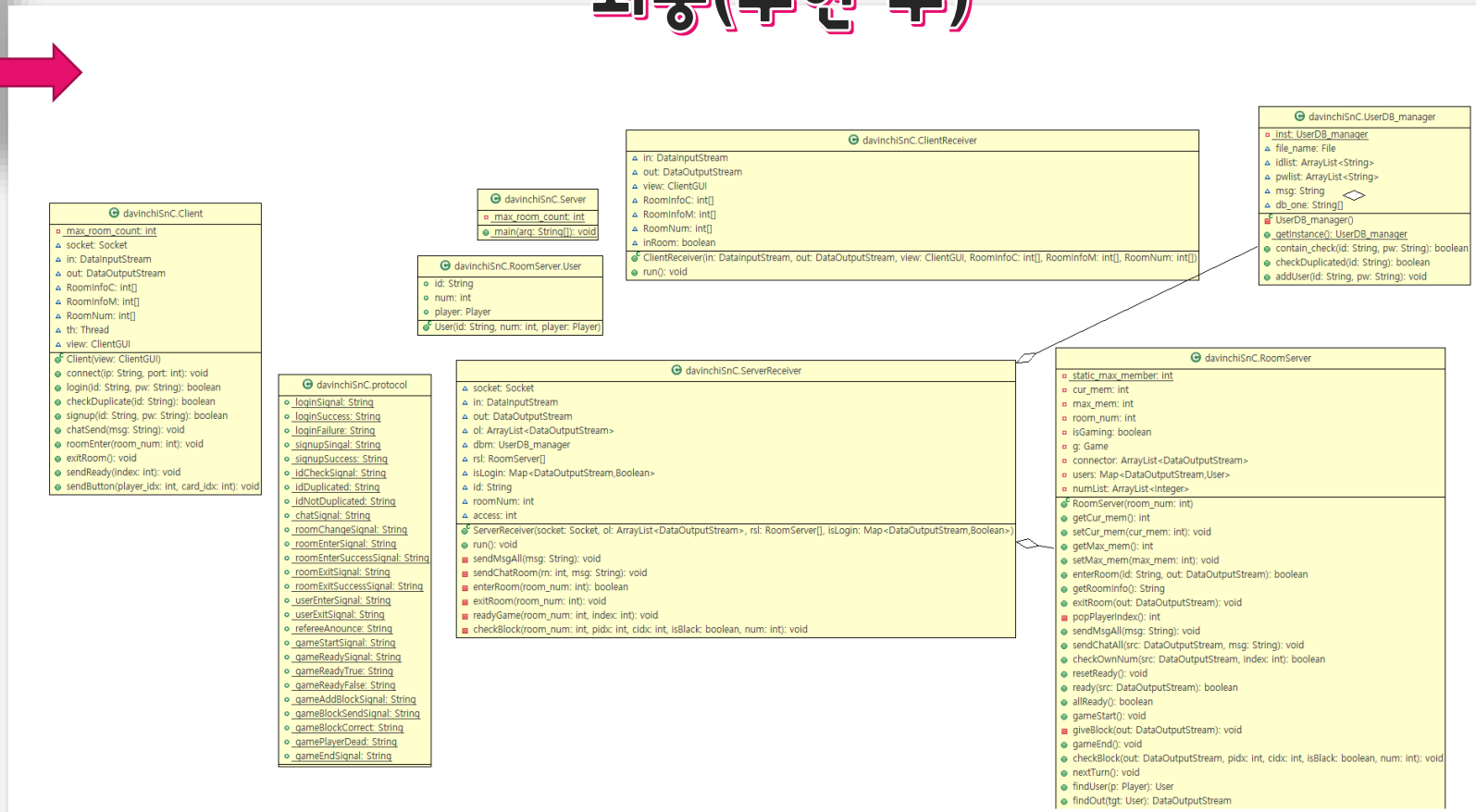


설계 : 클래스 다이어그램(서버)

중간



최종(구현 후)



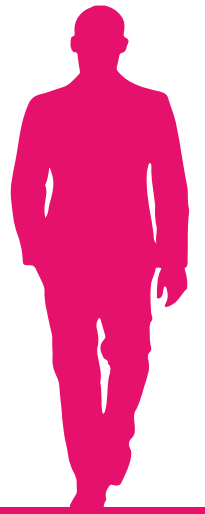
구현할 기능에 대한 고찰

GUI 구성

서버와의 연결

DB

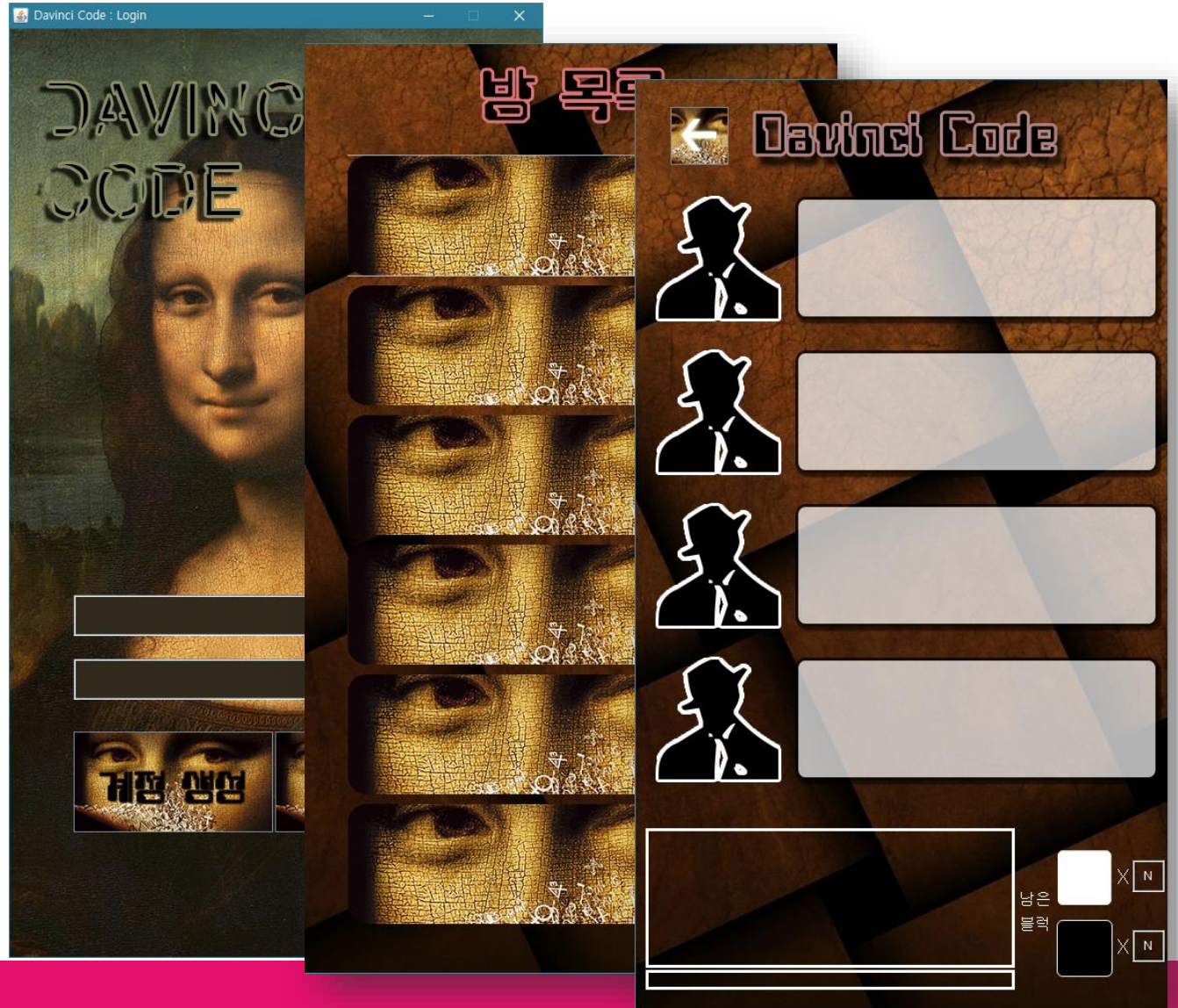
TTS 이용



구현 결과

GUI 구성: 메인

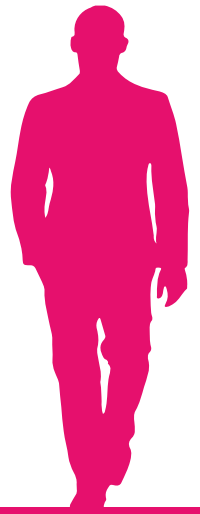
나머지 Frame들을 Panel화 -> 하나의 창에서 패널만 변화 (setContentPane() 활용)




```
mainGUI_final(){

    setTitle("DAVINCI CODE");
    RoomListGUI_p rlP = new RoomListGUI_p(); //방 목록 패널 클래스
    LoginGUI_p lgP = new LoginGUI_p(); //로그인 패널 클래스
    GameGUI_p gP = new GameGUI_p(); //게임방 패널 클래스
```

메인프레임이 Panel들을 has하게 함

-> actionListener를 메인프레임에서 쉽게 추가 가능



GUI 구성: 로그인 창

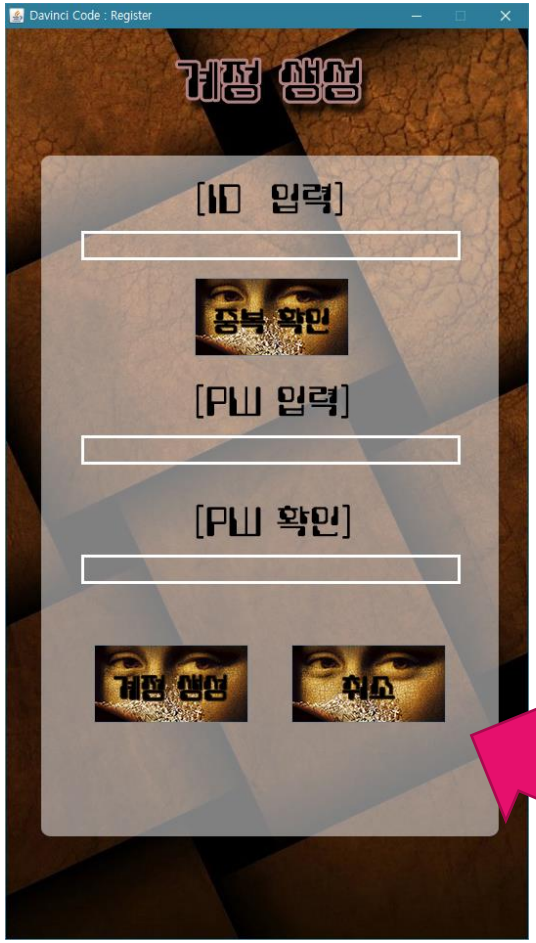
계정 생성 -> 창 띄우기

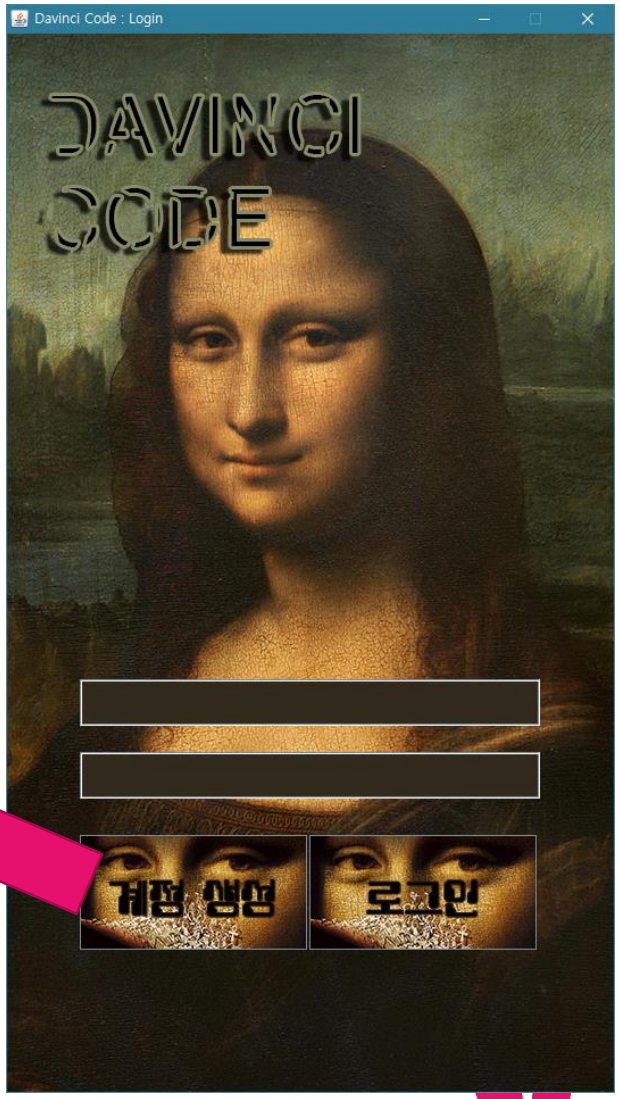
```
accountB.addActionListener(new ActionListener() { // 계정 생성 창 띄우기
    public void actionPerformed(ActionEvent e) { new SignUpGUI().setVisible(true); }
});
```

로그인 -> KeyListener & actionListener
버튼을 눌러도 되고, 엔터를 눌러도 된다

```
//(추가)엔터누르면 넘어감
lgP.IDField.addKeyListener((KeyAdapter) keyPressed(e) → {
        if(e.getKeyCode() == 10){ // 엔터 눌렀을 때

                setContentPane(rIP);

        }
});

//(추가)엔터누르면 넘어감
lgP.pwField.addKeyListener((KeyAdapter) keyPressed(e) → {
        if(e.getKeyCode() == 10){ // 엔터 눌렀을 때

                setContentPane(rIP);

        }
});

lgP.loginB.addActionListener(new ActionListener() { // 로그인 -> 방 목록 패널
    public void actionPerformed(ActionEvent e) {
        id = lgP.IDField.getText();
        //pw = lgP.pwField.getPassword();
        setContentPane(rIP);
    }
});
```

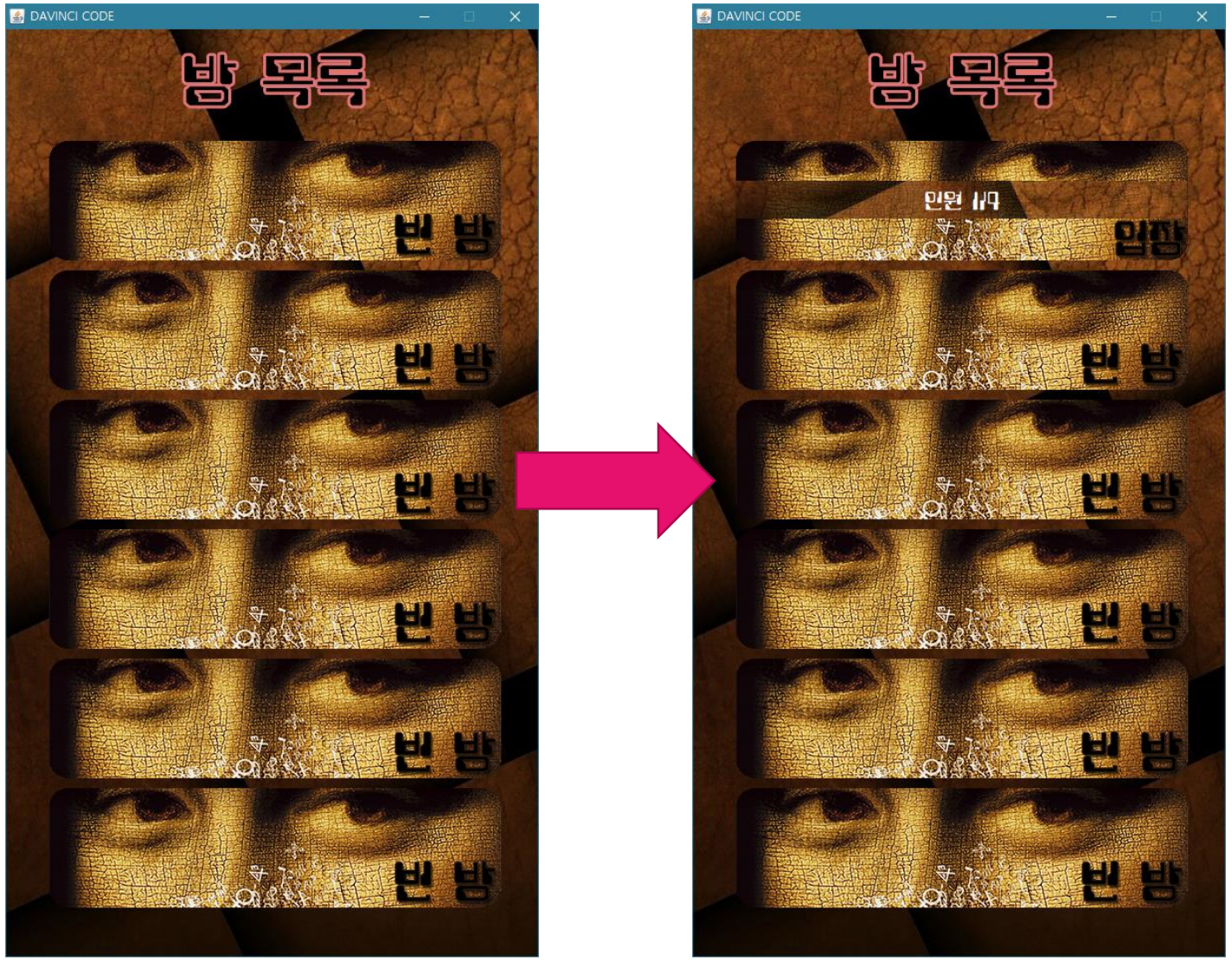
AI 창으로 생성



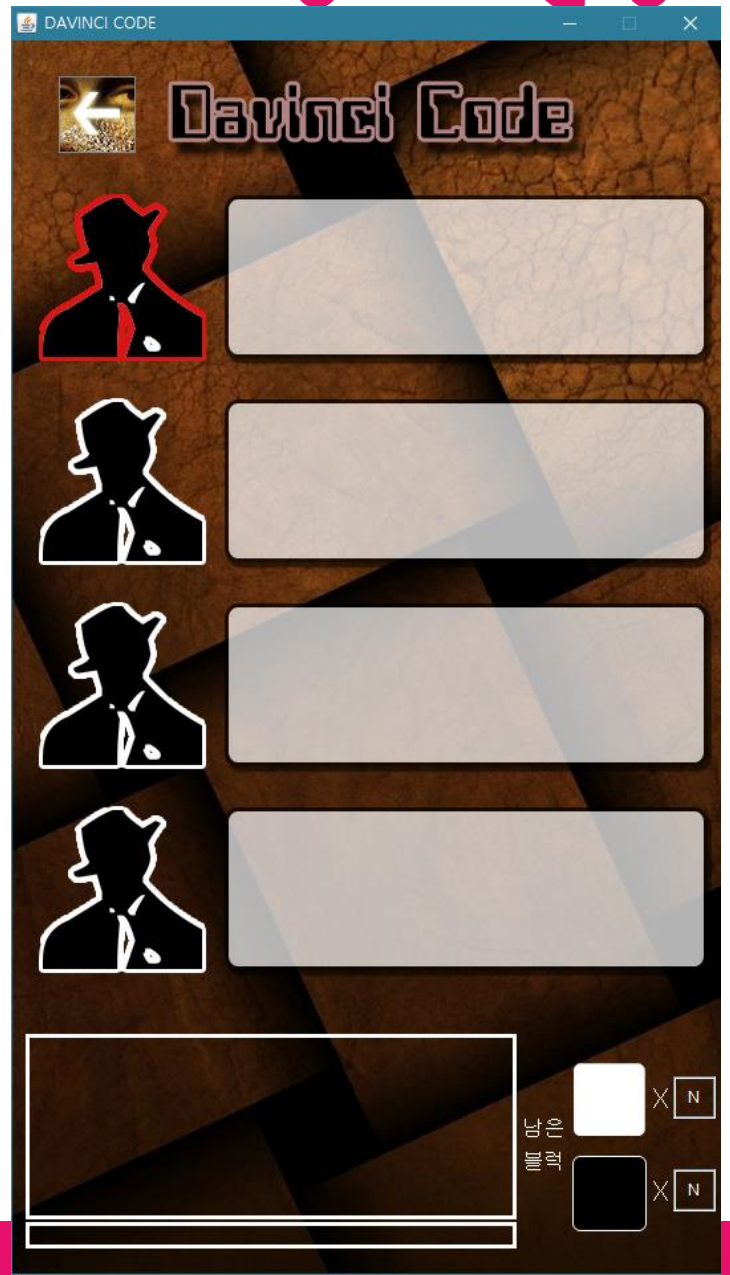


GUI 구성: 방 목록

방 접속 -> 인원 상승 및 이미지 아이콘 변경



GUI 구성: 게임방



접속한 유저

레디한 유저

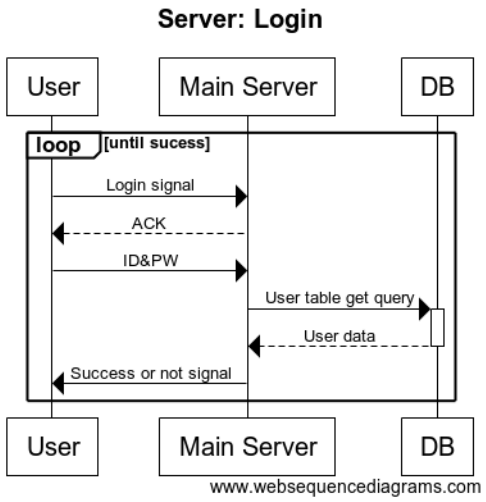
빈 유저

죽은 유저

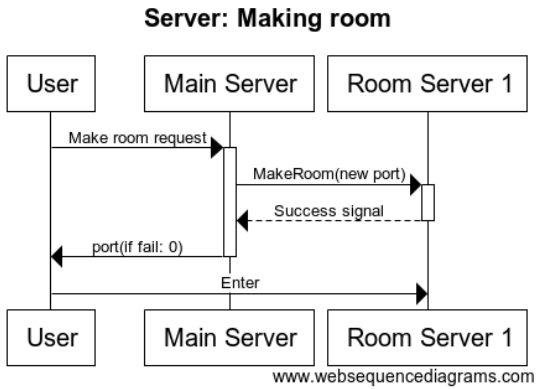
내 블록의 숫자만 보이고
상대 블록은 안 보인다

맞춘 블록은 공개된다

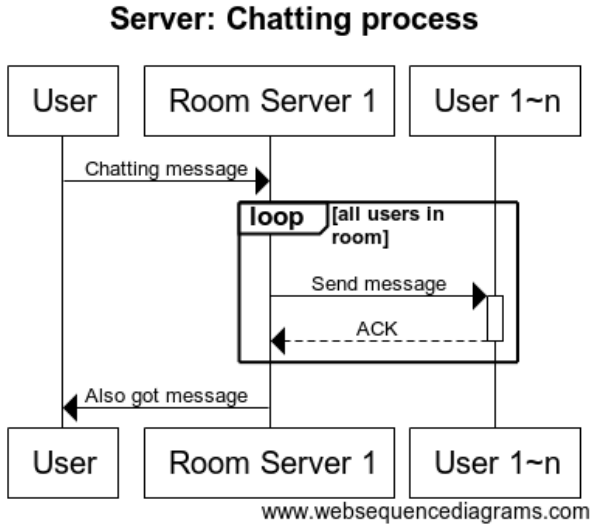
서버



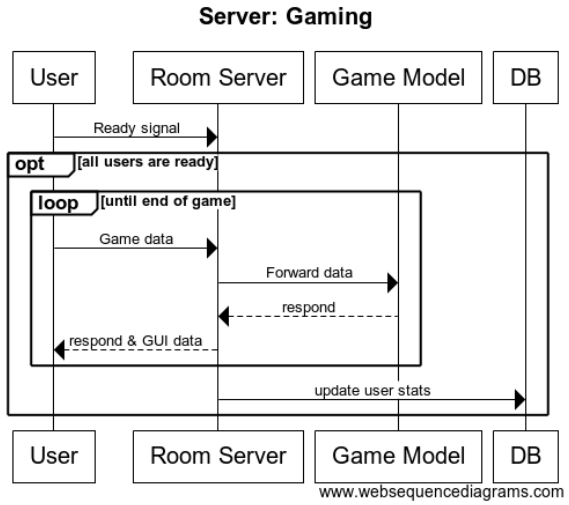
로그인



방 입장

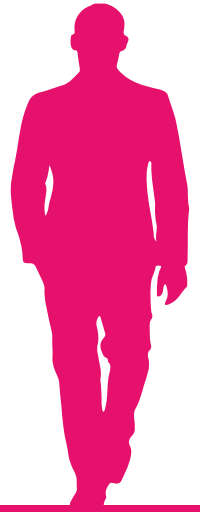


채팅



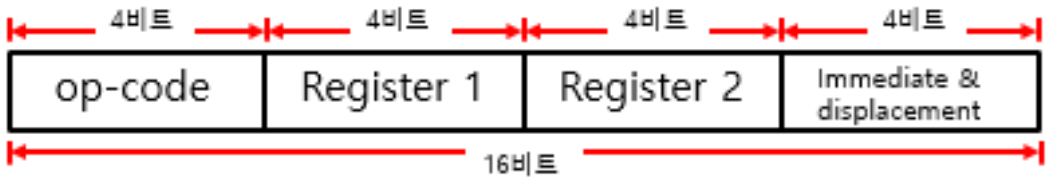
게임 과정

미리 시퀀스 설계한 서버 Flow를 이용



서버

오퍼레이션 코드와 유사한 통신구조



- Op-code
  - ➢ 4비트 → 2^4 = 16가지의 연산 정의.
- Operand
  - ➢ Register no. : 4비트 → 2^4 = 16 개의 Register
  - ➢ Memory address : 4비트 → 주소영역 : 0 ~ $2^4 - 1$
  - ➢ Immediate Value : 4비트 → 표현범위 : -2^3 ~ $+(2^3 - 1)$

    ex) LOAD R1, 8(R2) : R1← M[R2+8]

    ADD R1, R2, #1000b : R1← R2 + (-8)

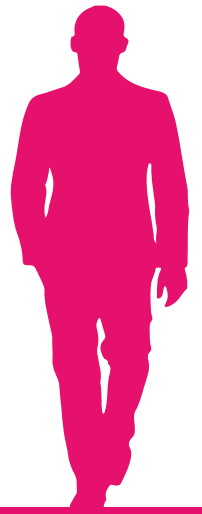
비트대신 String의 단위를 인덱스조 이용

```
public class protocol {
    // Login & Sign Up task
    public static final String loginSignal =          "[LOGIN]";
    public static final String loginSuccess =         "[LGSUC]";
    public static final String loginFailure =         "[LGFAL]";
    public static final String signupSingal =         "[SGNUP]";
    public static final String signupSuccess =        "[SGSUC]";
    public static final String idCheckSignal =        "[IDCHK]";
    public static final String idDuplicated =         "[IDDUP]";
    public static final String idNotDuplicated =      "[IDNDP]";

    // Room Info task ( 3 char + room_num + msg )
    public static final String chatSignal =               "000";
    public static final String roomChangeSignal =         "001";
    public static final String roomEnterSignal =          "002";
    public static final String roomEnterSuccessSignal =   "003";
    public static final String roomExitSignal =           "004";
    public static final String roomExitSuccessSignal =    "005";
    public static final String userEnterSignal        =   "006";
    public static final String userExitSignal         =   "007";

    // Room Info task ( 3 char + msg )
    public static final String refereeAnounce         =   "008";
    public static final String gameStartSignal        =   "009";
    public static final String gameReadySignal        =   "010";
    public static final String gameReadyTrue          =   "011";
    public static final String gameReadyFalse         =   "012";
    public static final String gameAddBlockSignal     =   "013";
    public static final String gameBlockSendSignal    =   "014";
    public static final String gameBlockCorrect       =   "015";
    public static final String gamePlayerDead         =   "016";
    public static final String gameEndSignal          =   "017";
```

서버

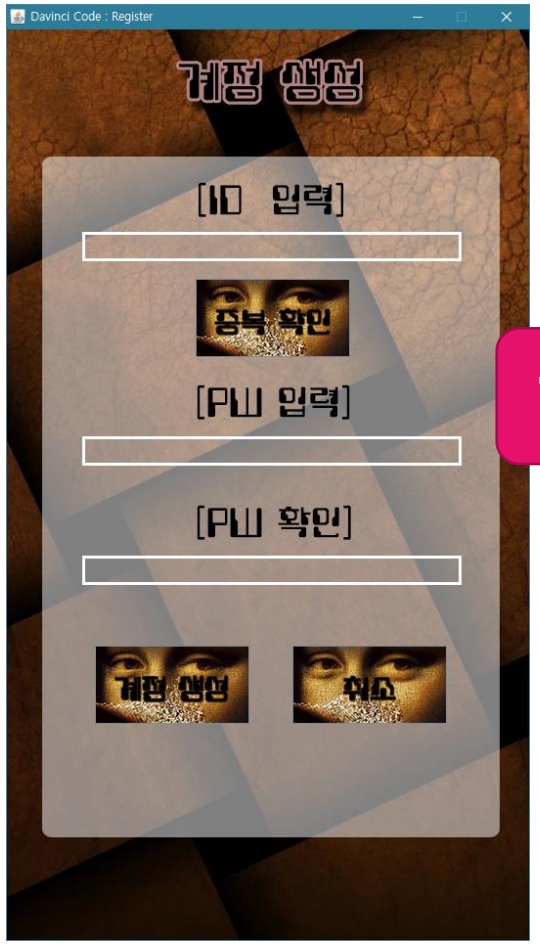
```
if(cmds[0].equals(protocol.loginSignal)) {
    if(dbm.contain_check(cmds[1],cmds[2])) {
        String room_info = "";
        for (RoomServer r: rsl) {
            room_info = room_info.concat(Integer.toString(r.getCur_mem()) + Integer.toString(r.getMax_mem()));
        }
        out.writeUTF( str: protocol.loginSuccess + room_info);
        id = cmds[1];
        break;
    }
    else {
        out.writeUTF(protocol.loginFailure);
    }

}else if(cmds[0].equals(protocol.signupSingal)) {
    if(dbm.checkDuplicated(cmds[1]))
        out.writeUTF(protocol.idDuplicated);
    else {
        dbm.addUser(cmds[1], cmds[2]);
        out.writeUTF(protocol.signupSuccess);
    }
}else if(cmds[0].equals(protocol.idCheckSignal)) {
    if(dbm.checkDuplicated(cmds[1]))
        out.writeUTF(protocol.idDuplicated);
    else
        out.writeUTF(protocol.idNotDuplicated);
}
```

직접 설정해놓은 신호값에 따라서 작동



08

계정 생성 시 ID, PW를 저장하고
게임이 끝날 때마다 유저의 전적을 전송



계정 생성

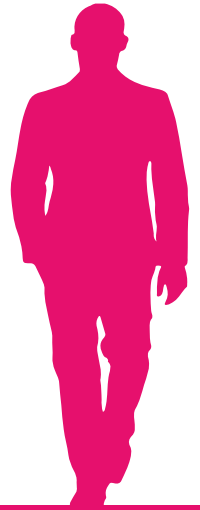
중복 체크

확인 후 INSERT

종료

전적값 UPDATE

데이터 베이스



TTS 이용: 네이버 API

한글 TTS 이용 시도 -> 네이버 API로 선택

NAVER CLOUD PLATFORM

APIURL에게 TEXT를 전송

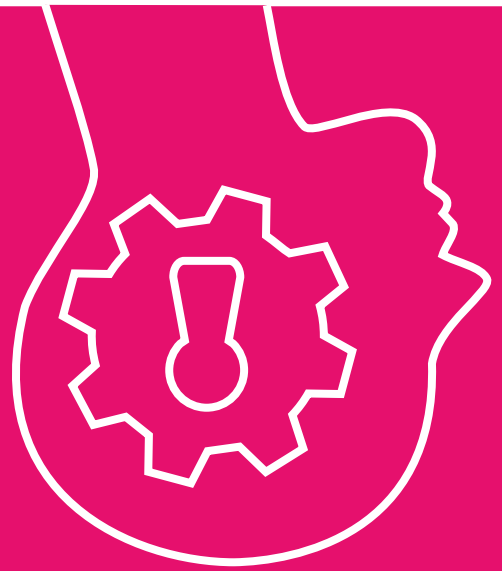
DATASTREAM으로 변환하여 다시 돌려줌

그것을 받아 JAVA ZOOM이 DATASTREAM을 읽어줌

```java
public class Clovatts {
    private String clientId = "q7fry28vkn";//애플리케이션 클라이언트 아이디값";
    private String clientSecret = "GO7dAoV7g2Vzy9kDGXVoi2IsPRgi4vP7wasj2RPG";//애플리케이션 클라이언트 시크릿값";
    public void readdo(String meant) {
        try {
            String text = URLEncoder.encode(meant, enc: "UTF-8"); // 13자
            String apiURL = "https://naveropenapi.apigw.ntruss.com/voice/v1/tts";
            URL url = new URL(apiURL);
            HttpURLConnection con = (HttpURLConnection)url.openConnection();
            con.setRequestMethod("POST");
            con.setRequestProperty("X-NCP-APIGW-API-KEY-ID", clientId);
            con.setRequestProperty("X-NCP-APIGW-API-KEY", clientSecret);
            // post request
            String postParams = "speaker=mijin&speed=0&text=" + text;
            con.setDoOutput(true);
            DataOutputStream wr = new DataOutputStream(con.getOutputStream());
            wr.writeBytes(postParams);
            wr.flush();
            wr.close();
            int responseCode = con.getResponseCode();
            BufferedReader br;
            if(responseCode==200) { // 정상 호
                InputStream is = con.getInputStream();
                Readtext p=new Readtext(is);
                p.play();
                is.close();
            } else {  // 에러 발생
                br = new BufferedReader(new InputStreamReader(con.getErrorStream()));
                String inputLine;
                StringBuffer response = new StringBuffer();
                while ((inputLine = br.readLine()) != null) {
                    response.append(inputLine);
                }
                br.close();
                System.out.println(response.toString());
            }
        } catch (Exception e) {
            System.out.println(e);
        }
    }

}
```

원래는 MP3파일을 생성해주는 코드였으나 소리만 출력하도록 수정!

시연 영상



자기 평가

잘 수행한 것

UML을 기반으로 한 게임코드 구현
(클래스 + 시퀀스 다이어그램)

GUI, DB, Game, Socket들의 병합

MVC 구조를 매우 잘 지킴.

성공적인 API(TTS) 이용

아쉬운 점

다채로운 인터페이스 활용 미흡

필요에 의한 이중 참조

많은 예외처리에도 잡지 못한 예외 상황

분담이후 학습 내용 공유 미흡

감사합니다