UI/UX 전문가 과정비트캠프

2021년 01월 20일 16회차

장 해 솔

이메일 : wkdgothf@gmail.com

github : https://github.com/legossol

목표

1. Server 가위바위보 문제 분석
   1-1예외처리
   1-2 가위바위보 문제 순서 복습

2. 1번 과정에서의 따른 질문

* Server 가위바위보 문제 분석

1-1예외처리.

예외처리1) Throws를 알아보자.

throws는 프로그램이 일을 처리하는 과정에서 문제가 생겼을 경우 처리를 중단하고 다른 처리를 하는 것을 말한다. 다르게 말하면 예외처리를 상위계층으로 넘겨 상위계층에서 처리하는 방법 즉, **내가 하기 싫으니 다른 메소드 니가 처리하라고 던져버리는 것.**

<throws 처리 코드>

```
public void testParentFunction(){
   try{
        testFunction();
   }catch(Exception e){
       //예외처리
   }
}

public void testFunction() throws Exception{
     int r = 4/0;
}
```

만약 여러개의 예외를 처리하고 싶다면 다음과 같이 ,를 사용해 주면 된다.

```
public void testFunction() throws NullPointerException, ArithmeticException{
    //to do
}
```

예를 들어 내가 4를 싫어해서 4를 예외 처리하고 싶을때 아래와 같은 매소드를 이용해 4를 뺄 수 있는 것이다.

```
public void me4(int num) throws Exception{
    if(num==4){
        throw new Exception();
    }
}
```

그리고 만약 server를 배우면서 예외처리된 것을 알려주고 싶을 경우에는 아래와 같은 방법으로 작성하면 된다.

```
public class ShowException extends NullPointerException{
    private static final long serialVersionUID = 1L;  //<== 요건 무시하세요.

    public ShowException(){
        super("NullPoint하하ㅏㅎㅎ하하하");
    }
}
```

```
public void me6(int num){
    if(num==4){
        throw new ShowException();
    }
}
```

예외처리 2) try catch를 알아보자

예외처리1에서 잠깐 등장하였다.try catch 는 try내에서 실행되는 코드에서 예외가 발생할 경우, catch가 exception을 잡아 예외처리를 가능하게 해주는 것

<try, catch, (finally)>

```
try{
   //To do
}catch(예외1){
   //예외처리1
}catch(예외2){
   //예외처리2
```

```
}finally{
   //예외처리 이후 처리할 문장.
}
```

이후에 사용되는 finally구문은 예외처리와 상관없이 무조건 수행되는 문장으로 생략이 가능하다.
단, 주의해야 할 점은 try문을 돌다가 catch1에 걸릴경우 catch2는 처리하지 않는다는 것을 명심하자.

*아래표는 예측할 수 있는 예외 사항들을 말함.

예외	설명
NullPointerException	널 객체를 참조하는 경우
NegativeArraySizeException	배열의 크기가 음수인 경우
ArithmeticException	어떤 수를 0으로 나눌 경우
ClassCastException	적철치 못한 클래스로 형변환하는 경우
ArrayIndexOutOfBoundsException	배열을 참조하는 인덱스가 잘못된 경우
NoClassDefFoundError	클래스를 찾지 못하는 경우

1-1가위바위보 문제 순서 복습.

1)클라이언트 입장

```java
// 게임에 접속해서 플레이하는 고객
public class MainClient {
    public static void main(String[] args) throws IOException, InterruptedException {
        // 집에서 하는 경우 자신의 집 컴퓨터 아이피 주소
        // 학원인 경우는 자신의 자리 ip 주소 혹은 동일하게 해도 무방
        String ip = "192.168.0.9"; // 접속하려는 IP.
        int port = 7777;

        ClientSocketManager csm = new ClientSocketManager(ip, port); // ①
        System.out.println("접속 요청 완료!");
        System.out.println("가위는 1, 바위는 2, 보는 3");

        csm.send(csm.getClntSock()); // ③ ②
        System.out.println("전송 완료!");
```

RUN하는 순간 아래 Println 실행

```java
public class ClientSocketManager extends SocketManager {
    private Socket clntSock;

    public ClientSocketManager(String hostIp, int portNum)
            throws IOException {

        super();
        clntSock = new Socket(hostIp, portNum);
    }

    public Socket getClntSock() {
        return clntSock;
    }
}
```

상속

Run을 돌리는 순간 클라이언트의 Ip와 port번호를 상속자인 ClientSocketManager로 입력되고 ClntSock에 대입된다.

접속요청완료와 가위바위보를 입력하라는 말이 출력되고 getClntSock을 통해 clntSock값이 매소드를 통과한다.

다음으로 

3번인 send메소드로 이동한다.

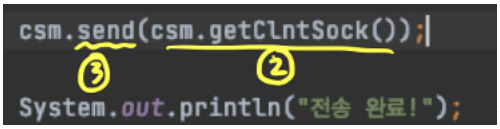
```
public void send(Socket sock) throws IOException {
    System.out.print("숫자를 입력하세요: ");
    String str = scan.nextLine();

    out[ZERO] = sock.getOutputStream();
    out[ZERO].write(str.getBytes());
}
```

Send메소드를 통해 클라이언트에게 숫자를 입력 받는다.

```
out[ZERO] = sock.getOutputStream();
out[ZERO].write(str.getBytes());
```

그리고 클라이언트로 부터 입력 받은 데이터를 송신하기위해 sock.getOutputStream을 하고 기가바이트 형식으로 서버로 보낸다(write)



이 과정을 마친 뒤에 “전송완료!”라는 출력을 받게 되며 후에 아래의 매소드를 통해 새로운 출력을 전달 받는다.

```
if(ssm.canWeGetWinner(ssm.getMaxClnt())) {
    System.out.println("승패가 결정되었습니다.");
} else {
    System.out.println("무승부: 게임을 다시 시작합니다.");
}

ssm.send(ssm.getClntSockArr(), ssm.getMaxClnt());

System.out.println("모든 사용자에게 입력 결과 전달 완료!");
```

1)서버의 입장.

서버 장이 서버를 열면 일단 Tcp 통신 번호를 정하고 참가할 수 있는 참가자 수를 정하여 입력한다.

```
private static final int MAX = 3;

public static void main(String[] args)
        throws IOException, InterruptedException {
① ServerSocketManager ssm =
            new ServerSocketManager( portNum: 7777, MAX);
```

첫번째로 ServerSocketManager에 정보가 담기고

```
public class ServerSocketManager extends SocketManager {
    private ServerSocket servSock;
    private Socket[] clntSockArr;

    private int clntCnt;
    private int maxClnt;

    public ServerSocketManager(int portNum, int max)
            throws IOException {

        super(max);
              3명까지

        System.out.printf("%d 명이 접속해야 게임을 시작할 수 있습니다.\n", max);

        servSock = new ServerSocket(portNum);
            TCP통신을 위한 Port 번호

        clntCnt = 0;
        maxClnt = max;

        clntSockArr = new Socket[max];: (다중) 참여하는 클라이언트를 구별하기위한 배열
    }
```

두번째로 참가자들을 기다리는 매소드가 활용된다.

```
② ssm.waitForClientRequest();
```

이미 처음에ServerSocketManager에 정보가 담기며 clntSockArr이라는 클라이언트를 구별하는 배열을 생성해 놓았고 아래 매소드는 그것을 사용하여 클라이언트가 총 3명이 될때까지 대기한다.

```
public void waitForClientRequest() throws IOException {
    System.out.println("사용자 접속을 대기합니다.");

    for(int i = 0; i < maxClnt; i++) {
        clntSockArr[clntCnt++] = servSock.accept();
    }   =[0],[1],[2] 총 3명의 클라이언트가 참가 할 때까지 기다림.
}
```

클라이언트들이 모두 접속하면 아래 메시지가 서버장에게 출력되며

```
③ System.out.println("모두 접속 완료!");
```

각 사용자들의 ip정보를 출력해주는 메소드를 작동시킨다.

```
④ ssm.checkEachIpAddressInfo();

public void checkEachIpAddressInfo() {
    for(int i = 0; i < maxClnt; i++) {
        System.out.println(
                "[" + clntSockArr[i].getInetAddress() +
                        "] client connected"
        );
```

다음으로 아래의 코드를 수행해야하며

```
③ssm.recv(ssm.getClntSockArr(), ssm.getMaxClnt());
```

getClntSockArr을 통해 clntsockArr이라는 배열을
geMaxClnt를 통해 maxCLnt를 받아서 recv로 간다.

```
private Socket[] clntSockArr;

private int clntCnt;
public Socket[] getClntSockArr() {
    return clntSockArr;
}
public int getMaxClnt() {
    return maxClnt;
}
```

Recv 매소드 안에서 클라이언트들이 입력했던 가위, 바위, 보 에 해당하는 1,2,3 숫자를 기가바이트 형식으로 받아왔다 그것을 in[i]의 배열에 값으로 된다.

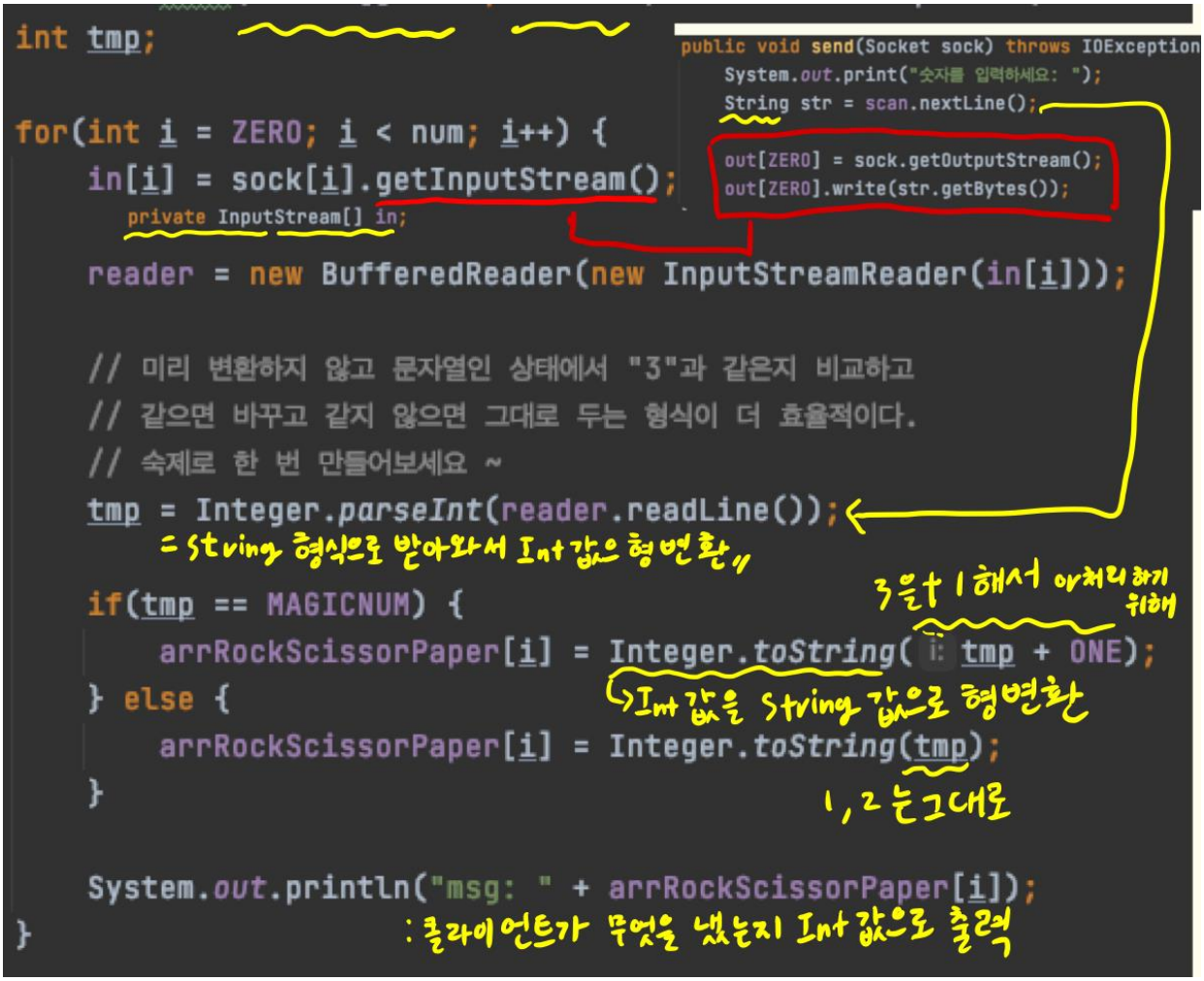
```
public void recv(Socket[] sock, int num) throws IOException {
    int tmp;

    for(int i = ZERO; i < num; i++) {
        in[i] = sock[i].getInputStream();
        private InputStream[] in;
        reader = new BufferedReader(new InputStreamReader(in[i]));
```

```
public void send(Socket sock) throws IOException
    System.out.print("숫자를 입력하세요: ");
    String str = scan.nextLine();

    out[ZERO] = sock.getOutputStream();
    out[ZERO].write(str.getBytes());
```

그다음 tmp를 통해 string 형식인 str을 인트형으로 변환하고
3보자기가 입력되었으면 임의적으로 + 1을 해서 4로 만들어 or연산을 편하게 한다.
1,2 가위와 바위는 그대로 두어 or연산을 진행한다음
각 플레이어가 어떤 값을 내었는지 알려준다.



다음으로 canweGetWinner 매소드를 통해 무승부를 판별한다.

```
if(ssm.canWeGetWinner(ssm.getMaxClnt())) {
    System.out.println("승패가 결정되었습니다.");
} else {
    System.out.println("무승부: 게임을 다시 시작합니다.");
}
```

```java
public boolean canWeGetWinner(int num) {
    // 3명 이상이 함께 가위바위보를 할 때 승패를 어떻게 판정할 것인가 ?
    // 가위 = 1, 바위 = 2, 보 = 4라면
    // 모든 값을 OR 했을때 7이 나올 것이다.
    // 만약 1, 2, 3이라면 OR 결과는 3이므로
    // 이것이 보인지 무승부인지 판정이 불가!
    int bitOROfAllInputString = ZERO;

    for(int i = ZERO; i < num; i++) {
        bitOROfAllInputString |=
                Integer.parseInt(arrRockScissorPaper[i]);
    }

    if(bitOROfAllInputString == 7) {
        return false;
    } else if(bitOROfAllInputString == 1) {
        return false;
    } else if(bitOROfAllInputString == 2) {
        return false;
    } else if(bitOROfAllInputString == 4) {
        return false;
    }

    return true;
```

1 = 0000 0001
2 = 0000 0010
4 = 0000 0100

OR연산

모두 다른 것을 낸 경우

셋다 1

셋다 2

셋다 3

마지막으로 send 매소드를 통해 각 사용자에게 결과를 출력해주게 된다.

```
ssm.send(ssm.getClntSockArr(), ssm.getMaxClnt());

System.out.println("모든 사용자에게 입력 결과 전달 완료!");
```

```
public void send(Socket[] sock, int num) throws IOException {
    for(int i = ZERO; i < num; i++) {
        out[i] = sock[i].getOutputStream();

        writer = new PrintWriter(out[i], autoFlush: true);

        String str = convertNumber2RSP();

        writer.println(str);
```

숫자로 표현되는 가위바위보를 다시 String 값으로 바꿔주기위해