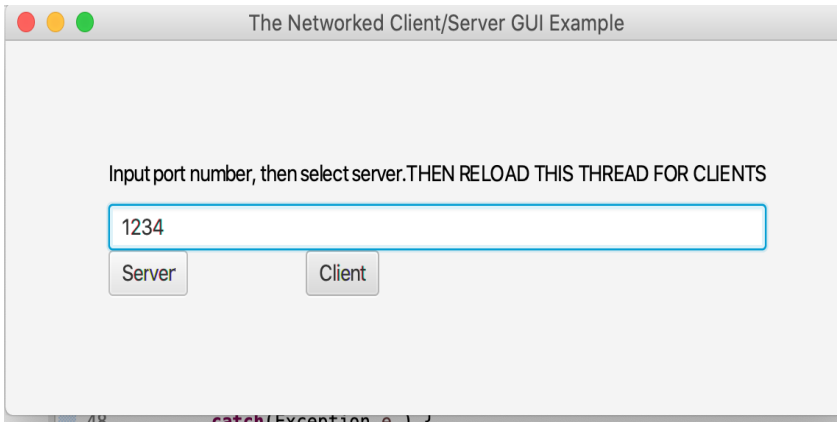
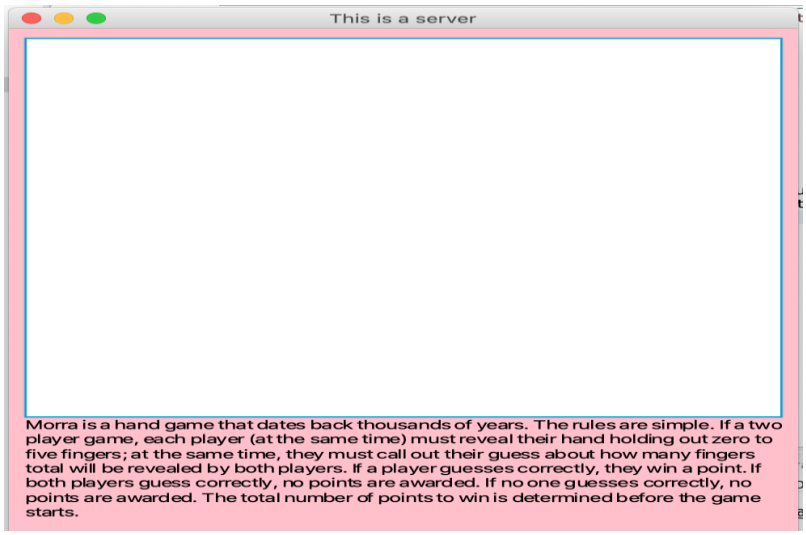
Khang Mach
Game of Mora (Project 3)

A. How the game work ?

The game of Mora will be divided into 2 section: Give number and guess total number from both player. Give number section will ask each user to give their a number from range 0-5. Once both give their own number, each will continue to guess sum number added by 2 guesses. Whoever guess correct will earn score. When one reach a certain amount will win. During the process, Server interface (GUI_ser.java) will coordinate who already to s/t, what section(give number/guess total), identify guess, give score, and winner announcement.

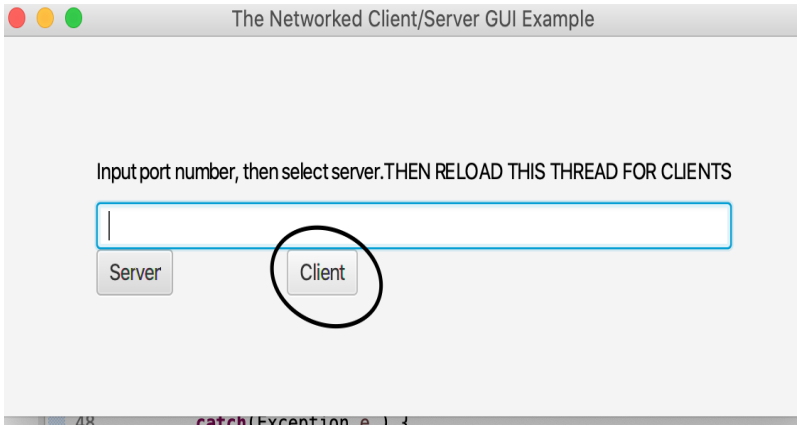
B. Game illustration

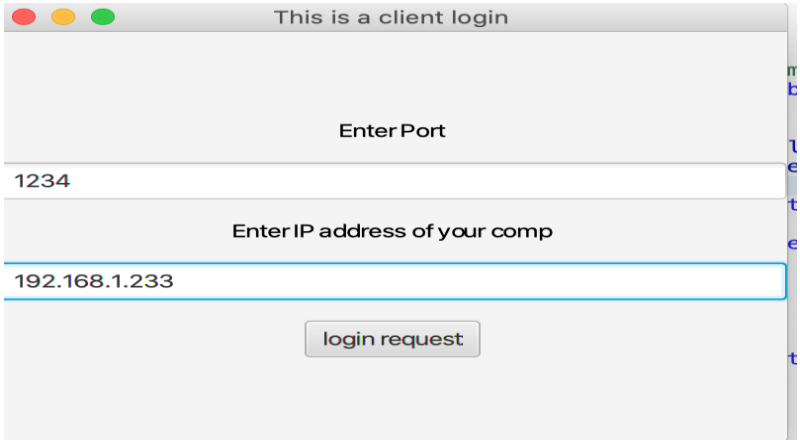




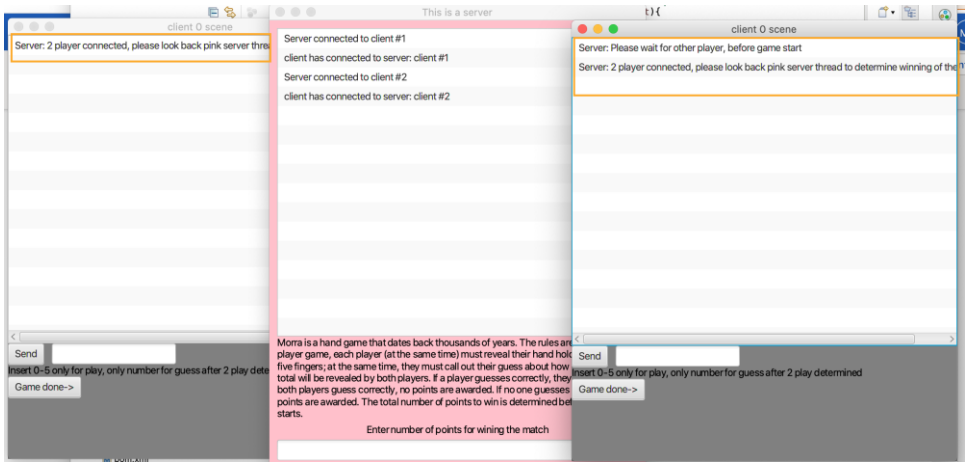
Create port for Server , any number, then click Server. Server thread (GUI_ser.java) created.

Khang Mach
Game of Mora (Project 3)



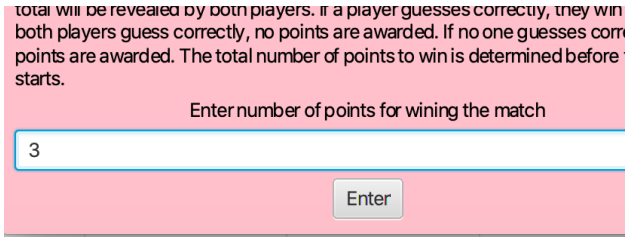


Next, reload and click Client. Then enter port of Server and your comp ip to create Client interface.



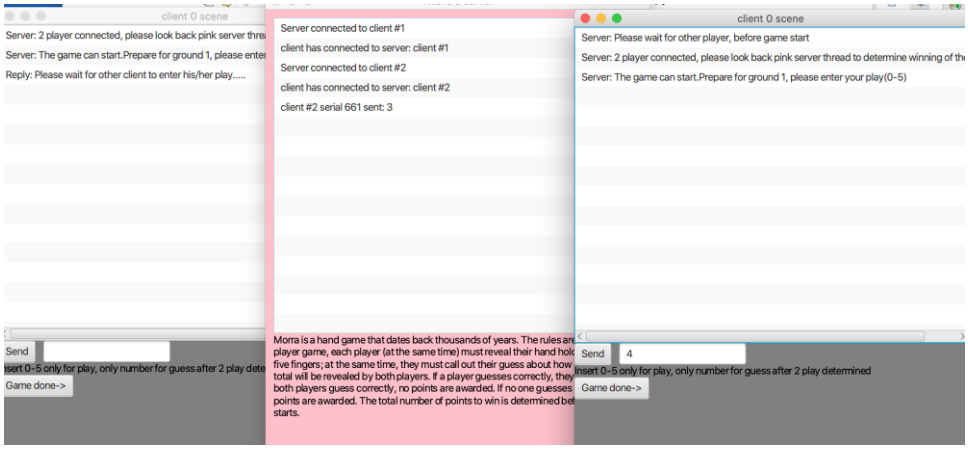
You have like this, after 2 client enter and prepare to play. Notice that Client #1 (on right), has first paragraph to wait Client #2. The game only start, after Client #2 enter, then paragraph “2 player connected...” appear on both Client interface to notify.

Khang Mach
Game of Mora (Project 3)

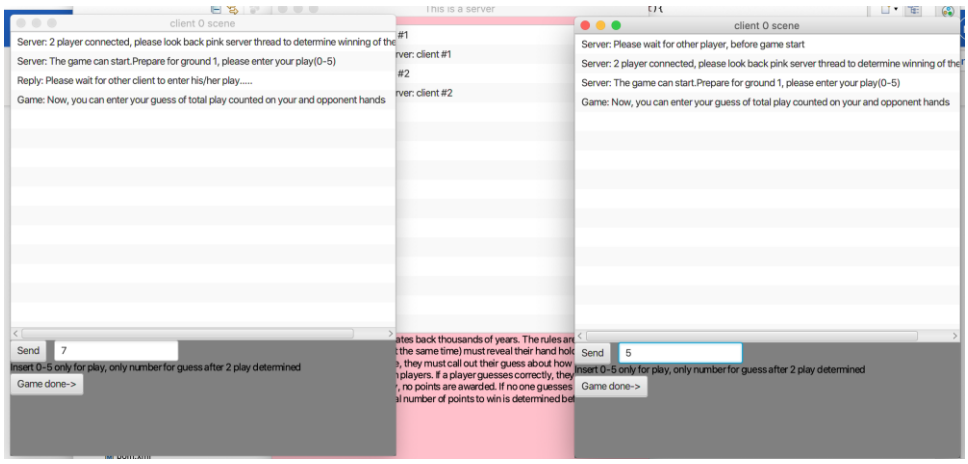


Server thread, control number of guess to win. 3 correct guess to win in this case

---

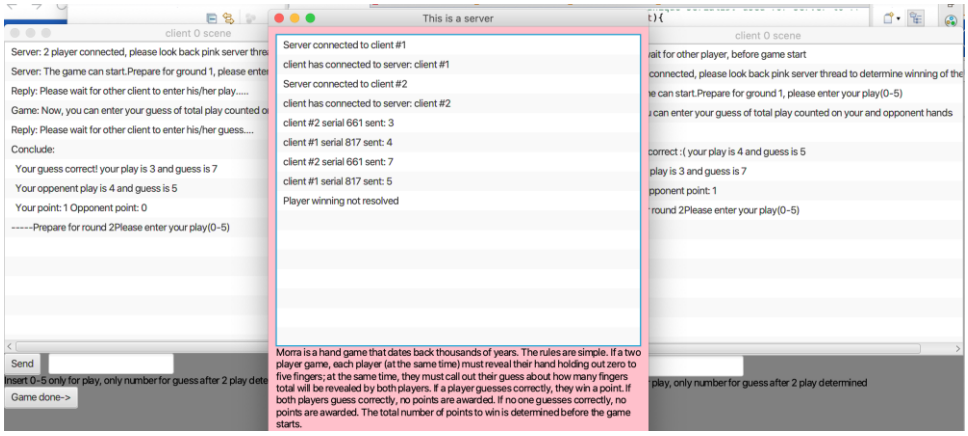


Round 1 start, Client #2 (on left) enter 3 for the play. Then Client #1 prepare enter 4. There is no issue of who enter play first.

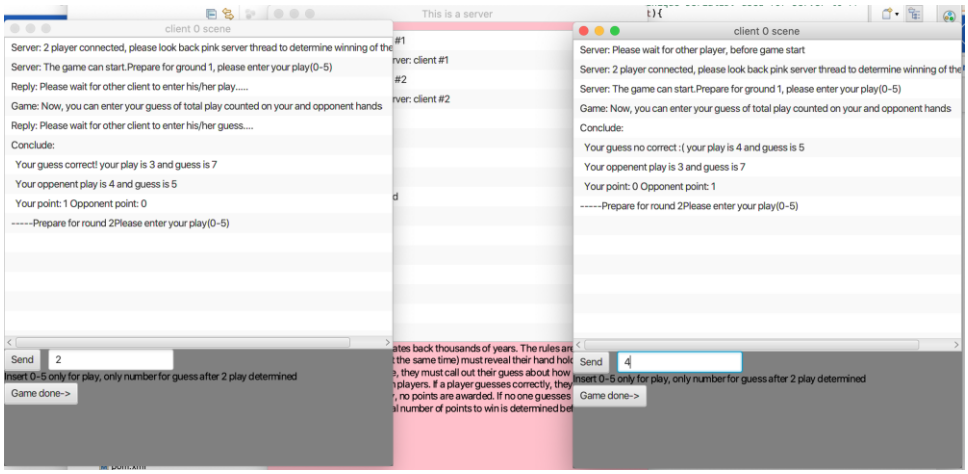


Khang Mach
Game of Mora (Project 3)

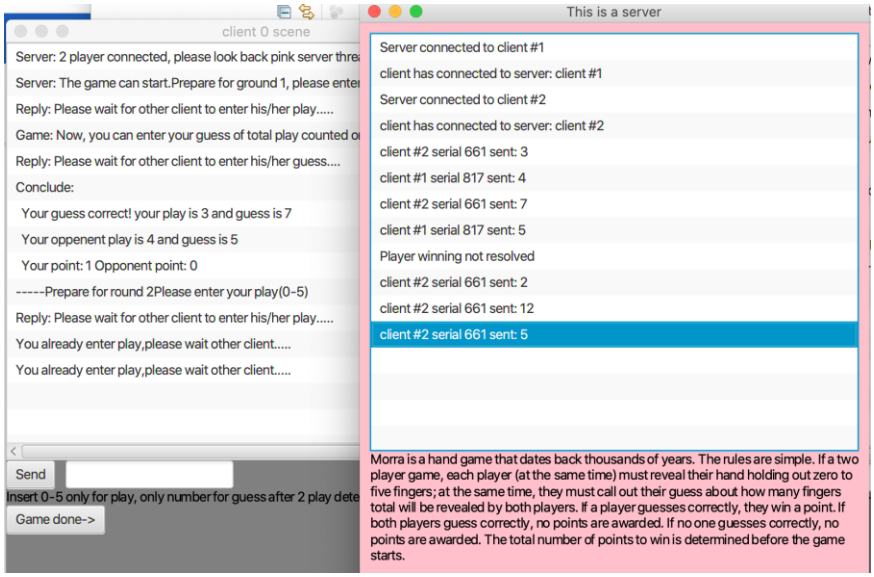
Round 1 continue, after both Client enter their play. Client prepare to enter guess of total play. There is no issue of who enter guess first.



Round 1 continue, Server thread keep track play and guess of both Clients, and determine who guess correct to give score. Here, Client #2 guess 7, which is correct.

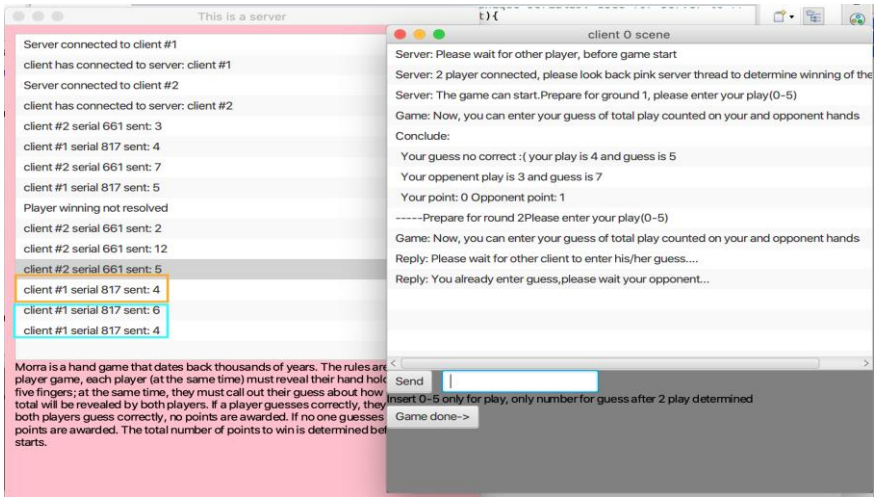


Round 2, both Client prepare enter their play.

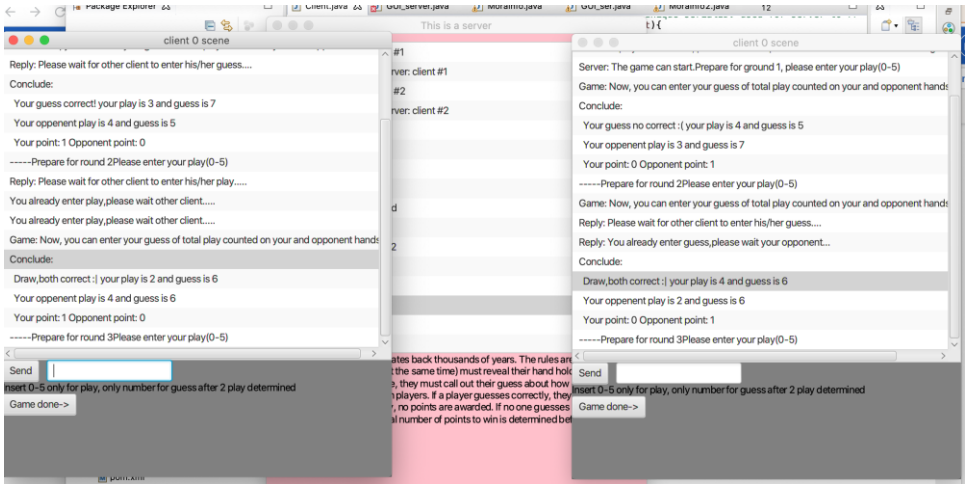


Khang Mach
Game of Mora (Project 3)

Although there no issue who enter guess/play first. But the game need player finish one section of the round first, before move to next section. Here, Client #2 is a bit hasty to enter its guess, but Server stop its action and remind Client #1 not done its play. So, Client #2 still has 2 as its guess.

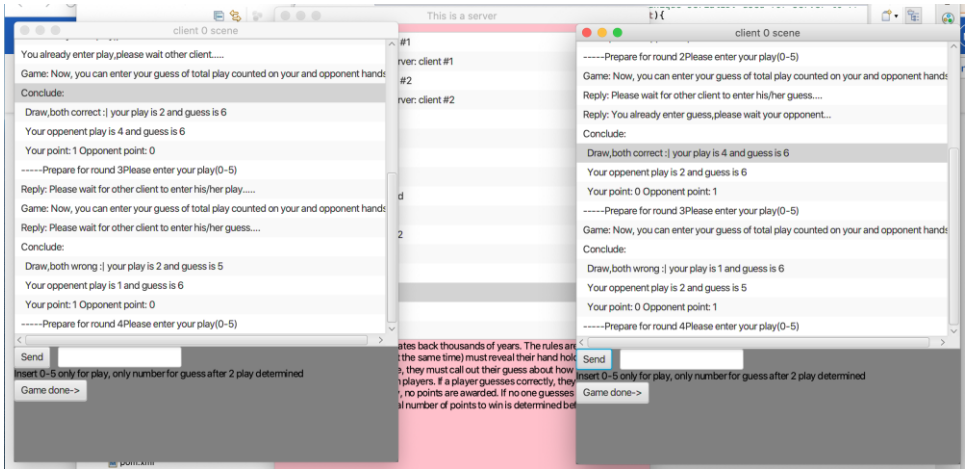


Round 2, on guess section. After Client #1 enter 6 as guess, it might want to change to 4. However, the Server stop it as Client #2 still not done its guess yet.



Round 2, draw since both enter 6 as guess, which is correct ans. So no score added to both Clients

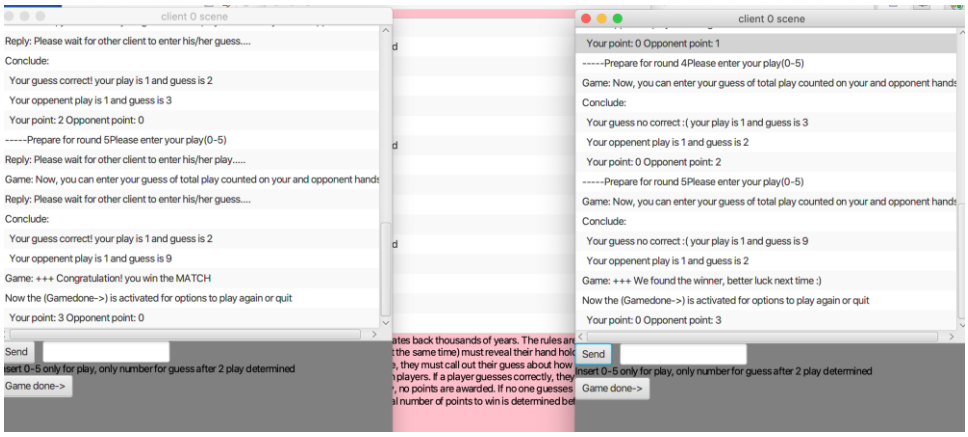
---



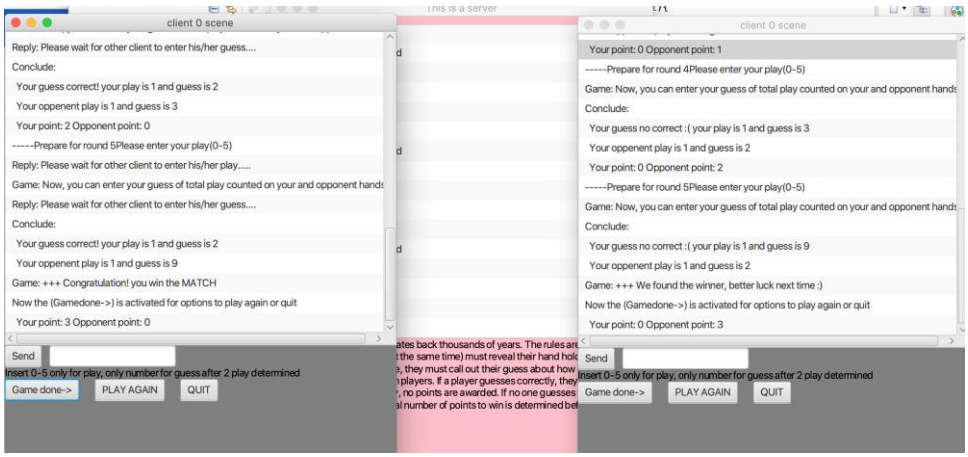
Khang Mach
Game of Mora (Project 3)

Round 3, both Client guess wrong, So no score added to both Clients

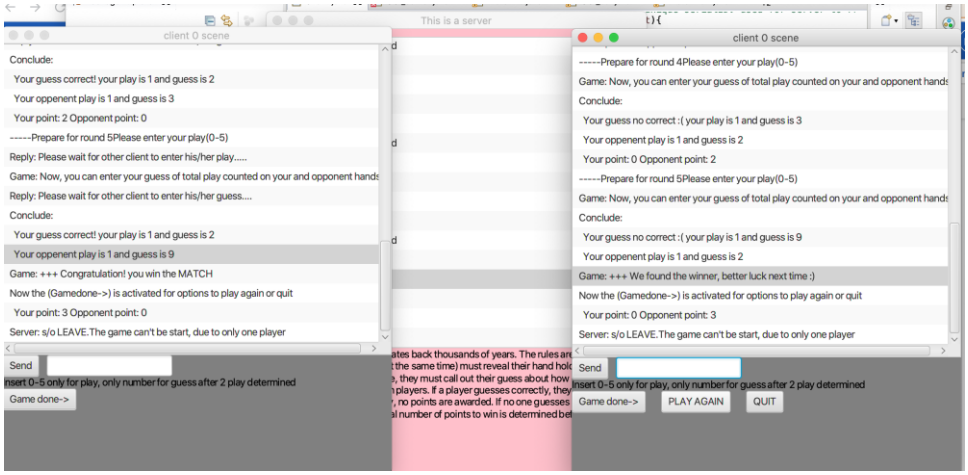
---



After done round 5, Server thread announce winner (Client 3) with 3 point as reach initial winning goal set of Server thread.

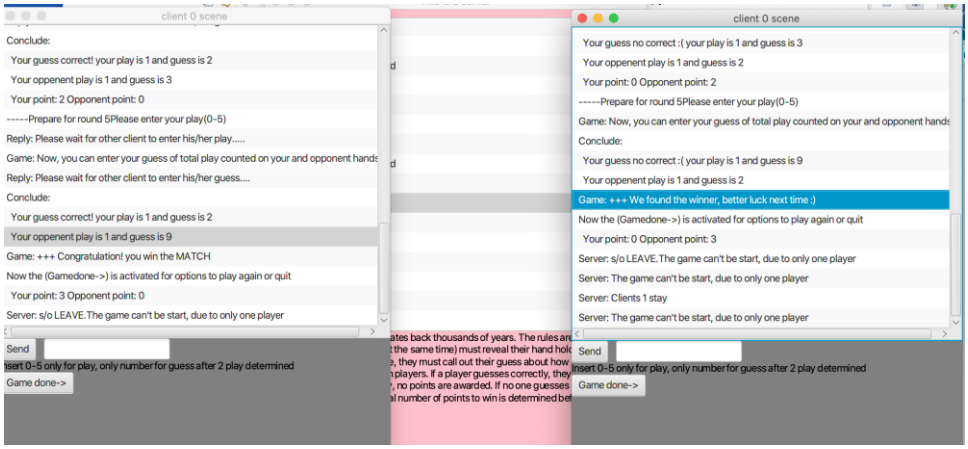


Now click “game done”, Clients has option to continue or quit.

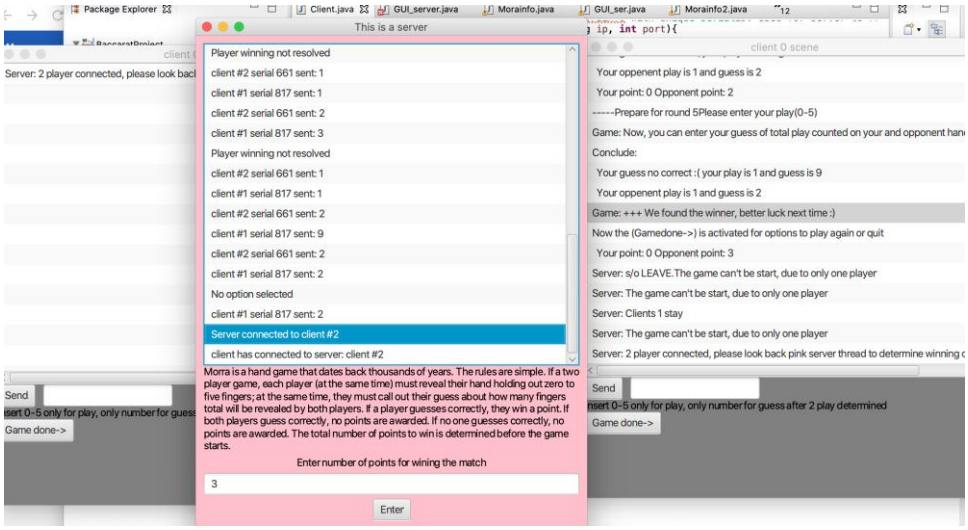


Although both not give there decision at same time. Client #2 decided to quit, while Client#1 not click yet, but we can see Client #1 has notification that other Client quit.

Khang Mach
Game of Mora (Project 3)



Although Client#1 decide to stay, but further send info send to Server will be rejected by notify of "The game can't be start, due to only one player".



Assume Client#1 wait, then new Client reload and enter the room. Both Client instantly got notified that "2 player connected, please look...". Now, we can return to Server thread to enter winning goal to start new game.

C. Intro to Client and Server thread functionality

GUI_ser.java

```
//login page, where enter port and ip for socketclient to connect client1
b_login = new Button("login request");
this.b_login.setOnAction(e-> {

    primaryStage.setScene(sceneMap.get("client"));
    primaryStage.setTitle("client "+overall_count+" scene");
    clientConnection = new Client(data->{
    Platform.runLater(()->{listItems2.getItems().add(data.toString());
            });
    },ip.getText(),Integer.parseInt(port.getText()));
    clientConnection.start();

    System.out.print("out of b_login1");

    System.out.println("  run_count"+overall_count+ "and "+count_only);

});
```

Client.java

```
public class Client extends Thread {


    Socket socketClient;

    ObjectOutputStream out;
    ObjectInputStream in;
    String ip;
    int port;
    Scanner scan= new Scanner(System.in);
    String info;
    private Consumer<Serializable> callback;
    Morainfo2 player = new Morainfo2(); // Morainfo with unique serialID. used for serv
    Client(Consumer<Serializable> call,String ip, int port){
        this.ip =ip;
        this.port =port;
        callback = call;
    }
```

The project focused on making interaction between Server and Clients thread. Each Clients interface in Client.java connected to Server need valid socket of ip/port, and implement of serialization.

Client.java

```
public void run() {

try {
//client socket + input/output stream established
System.out.println("CLient run1 established");

socketClient= new Socket(ip,port);
out = new ObjectOutputStream(socketClient.getOutputStream());
in = new ObjectInputStream(socketClient.getInputStream());
socketClient.setTcpNoDelay(true);
System.out.println("before player established");

System.out.println("player established"+player );
out.writeObject(player);
out.reset();

}
```

Khang Mach
Game of Mora (Project 3)

For each client interface in Client.java created will actively send and receive to/from particular clientThread.

Gui_ser.java

```
class TheServer extends Thread {
    Consumer<Serializable> callback;

    int port;
    Morainfo2 moraTurn = new Morainfo2(); // hold important information of both player, used t
    ArrayList<ClientThread> clients = new ArrayList<ClientThread>(); // collection of ClientTh
```

Here we can see each clientThread put in an array list once created

Gui_ser.java

```
//run() from The Server thread
public void run() {

    //try to connect single Clientsocket once established in while loop
    try(ServerSocket mysocket = new ServerSocket(Integer.parseInt(port_num.getText()))){
        System.out.println("Server is waiting for a client!");


        while(true) {

            ClientThread c = new ClientThread(mysocket.accept(), overall_count+1);
            int x =overall_count+1;
            callback.accept("client has connected to server: " + "client #" + x);
            clients.add(c); // add to clients array
            c.start();
            overall_count++; //track overall # clients in server


        }
```

Here, clientServer in clientThread is created through Server socket of Server thread (mysocket.accept())

GUI_ser.java

```
//
class ClientThread extends Thread{
    ObjectInputStream in;
    ObjectOutputStream out;
    Socket serverClient;
    int count;
    private long my_ID =0;
    ClientThread(Socket a, int b){
        serverClient = a;
        count =b;

        callback.accept("Server connected to client #"+count);
    }
```

GUI_ser.java

Khang Mach
Game of Mora (Project 3)

```
public void run() {

        try {
            System.out.println("CLientThread established");
            in = new ObjectInputStream(serverClient.getInputStream());
            out = new ObjectOutputStream(serverClient.getOutputStream());
            serverClient.setTcpNoDelay(true);


        } catch (Exception e) {
            // TODO Auto-generated catch block
            System.out.println("Streams not open");
        }


    try {
        // take advantage to synch UID of ClientThread to Client.
        // So not to guest clients array when who leave and what information lost, but
        System.out.println("infoPass established ");
        Morainfo2 infoPass = (Morainfo2)in.readObject();
        System.out.println("infoPass established "+infoPass.my_ID);
        System.out.println("infoPass play established "+infoPass.play[0]);

         my_ID  =infoPass.my_ID;
```

Client.java

```
public void run() {

try {
//client socket + input/output stream established
System.out.println("CLient run1 established");

socketClient= new Socket(ip,port);
out = new ObjectOutputStream(socketClient.getOutputStream());
in = new ObjectInputStream(socketClient.getInputStream());
socketClient.setTcpNoDelay(true);
System.out.println("before player established");

System.out.println("player established"+player );
out.writeObject(player);
out.reset();

}
```

Each clientThread and its I/O stream located GUI.ser.java response to connect I/O stream of specific Clients interface and global/local GUI_ser.java variables to facilitate data management and analysis at Morainfo.java (progress of each player). Later converted to I/O stream when clientThread run() activated. clientThread run() and Client interface run() will exchange information through I/O stream

Khang Mach
Game of Mora (Project 3)