# P.O.O(JAVA)

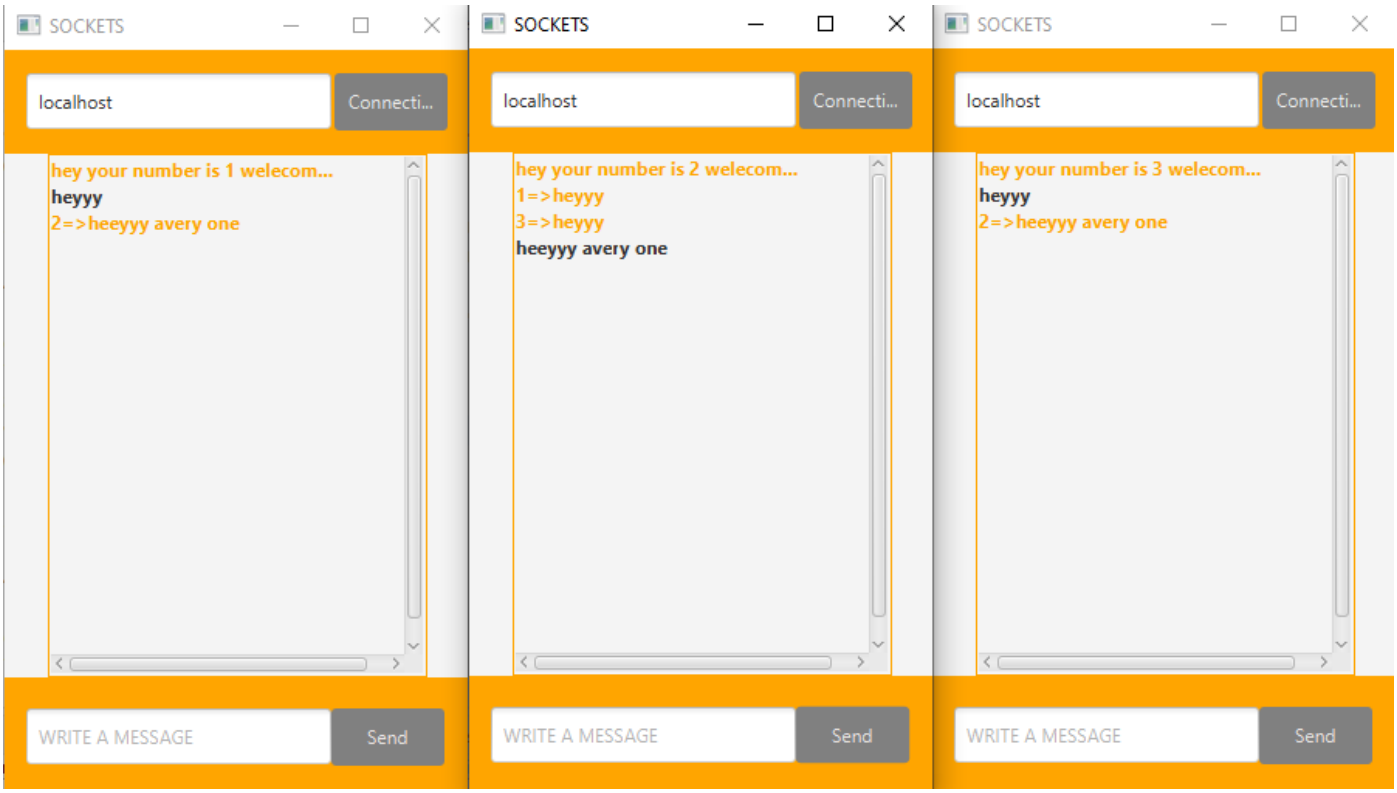
# SOCKETS

We are going to create a project about Client/Server

That can make clients connecting with each others through a server ,with the ability to change messages.

The client can send message to a specific client or as a broadcast to all client that are connecting to the server

Here is an example of 3 clients connected to the server one of them send a message to an other specific client.



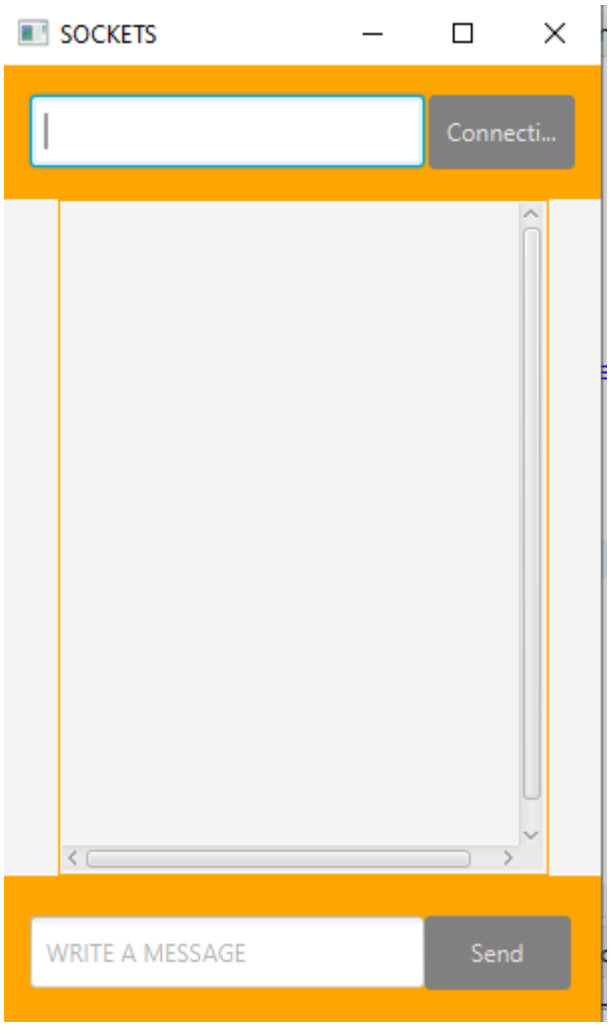
To make this app u should follow the instructions bellow:

```
//SERVER SIDE

public class ServerChat extends Thread {
            private int nbClient;
            private Conversation cnv;
            private ArrayList<Conversation> Clients=new
ArrayList<Conversation>();
            public static void main(String[] args) {
                  new ServerChat().start();
            }
            @Override
            public void run() {
                  try {
                        //show a message in the console
                        System.out.println("the server is runing");

            //create an instance of server sockets with port (1234)
            ServerSocket serverSocket=……

//create an infinite loop so the server can always listen
                        while(true) {
                        //create socket object
                              Socket sk=…….
                              ++nbClient;

//here we create inner class like a Thread
                              cnv=new Conversation(sk,nbClient);
                              Clients.add(cnv);
                              cnv.start();
                        }
                  } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                  }

            }
```

```
      class Conversation extends Thread{
                  protected Socket socket;
                  protected int nbClient;
                   public Conversation(Socket sk,int nb) {
                        this.socket=sk;
                        this.nbClient=nb;
                   }
                  public void sendMsg(String s,Socket sk,int nb){
                       try {
                            for(Conversation client:Clients) {
                            if (client.socket!=sk) {
                                 if(client.nbClient==nb || nb==-1) {
                                      OutputStream
os=client.socket.getOutputStream();
                                 PrintWriter pw=new
PrintWriter(os,true);
                                 pw.println(s);
                                 }

                            }
                       }
                       }catch (IOException e) {
                            // TODO Auto-generated catch block
                            e.printStackTrace();
                       }


                  }
                  @Override
                  public void run() {

                       try {
      // create the proper object to read a message from client
                            InputStream is=…
                          InputStreamReader isr=…
                          BufferedReader br=…

         // create the proper object to write a message to client

                          OutputStream os=…
                          PrintWriter pw =…

         System.out.println("the client "+nbClient+" is connected ");
                     pw.println("hey youu "+nbClient+" welecom... ");
```

```
                    while(true) {
       //wait until the client send a message then we read it
                           String req=br….
      //here where we knew which client should receive the msg
      //split: The string split() method breaks a given string
around matches of the given regular expression
                           if(req.contains("=>")) {
                                 String[] msgParams=req.split("=>");
                           String msg=msgParams[1];
                          int nbCli=Integer.parseInt(msgParams[0]);
                           sendMsg(msg,socket,nbCli);
                           }else {
                                 sendMsg(req,socket,-1);
                           }



                          }

                    } catch (IOException e) {
                           // TODO Auto-generated catch block
                           e.printStackTrace();
                    }



              }
        }

}
```

```
//Client Side
 //create graphical interface two button (connection) and (send)  with
scene builder
```




```
//configure the controller of this graphic

public class InterfaceClientController {
	@FXML
	private TextField host;
	@FXML
	private TextField msg;
	ObservableList<Label> ListModel=FXCollections.
                                              observableArrayList();
	@FXML
	private VBox msgs;
	PrintWriter pw;
```

```
@FXML
      public void ConncteToServer(ActionEvent e) {
            String hostName=host.getText();
            int port=6666;

            try {
                  / create the proper object to read a message from Server
                              InputStream is=...
                              InputStreamReader isr=...
                               BufferedReader br=....
                  // create the proper object to write a message to client
                              OutputStream os=...
                  /we already created this variable (pw) previously
                  pw=new PrintWriter(os,true);
                  new Thread(()->{
                        while(true){
                                    try {
                                    String rep=br.readLine();
                                    Platform.runLater(()->{
                                    Label label=new Label(rep);
label.setStyle("-fx-font-weight: bold; -fx-text-fill:orange;  ");
                                    ListModel.add(label);
                              msgs.getChildren().setAll(ListModel);
                                                      });
                                    } catch (IOException exp) {
                                    // TODO Auto-generated catch block
                                    exp.printStackTrace();
                                    }}}).start();

            } catch (IOException exp) { exp.printStackTrace();}}

                  @FXML
      public void sendMsg(ActionEvent e) {
            String textMsg=msg.getText();
            Label label;
            if(textMsg.contains("=>")) {
            String[] msgParams=textMsg.split("=>");
      String msgt=msgParams[1];
            label=new Label(msgt);
            }else {
                  label=new Label(textMsg);
            }
            label.setStyle("-fx-font-weight: bold;");
            label.setTextAlignment(TextAlignment.RIGHT);
            ListModel.add(label);
            msgs.getChildren().setAll(ListModel);
            msg.setText("");
            pw.println(textMsg);
      }}
```