**Simple GUI Based Chatting Application With Datagrams In Java**

**HYDER NABI**

**ENROLLMENT NUMBER: 20045110005**

**SEMISTER: MCA – III**

**BATCH: 2020**

**SUBJECT: JAVA PROGRAMMING**

**Description of the project: - This project is a Simple GUI Based Chatting Application written in java using Java Datagrams. Datagrams are bundles of information passed between machines. Datagram Transmission is Unreliable Transmission. Java implements Datagrams on top of UDP (User Datagram Protocol) by using two classes:**

1. **DatagramPacket: - This Object is the data container.**
2. **DatagramSocket: - This Object is the mechanism used to send or receive the DatagramPackets. It defines two standard methods:**
   * **Send(DatagramPacket packet): used to send DatagramPacket.**
   * **Receive(DatagramPacket packet): used to receive DatagramPacket.**

**These classes are under the package of *java.net.\****

**In order to send data and receive data concurrently, two threads have been used:**

1. **Send(): Thread used to send a Datagram Packet to the intended destination.**
2. **Receive(): Thread used to receive a Datagram Packet.**

import java.net.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.IOException;

//::::::::DESIGNED AND DEVELOPED BY HYDER NABI:::::::::://

//::::::::::::::::::USER INTERFACE FOR CHAT PROGRAM:::::::::::::::://

class UserInterface extends JFrame{

    JPanel panelUp,panelDown;//PANELS TO HOLD TEXT

    JTextField input;//INPUT FIELD WHERE TEXTS ARE WRITTEN

    JButton send;//BUTTON USED TO SEND TEST

    JScrollPane scrollpane; //SCROLL PANE USED TO SCROLL TEXTS WHEN OVERFLOW OCCURS IN THE PANEL

    public UserInterface(String name)

    {

        this.setSize(500, 700);//SIZE OF PANEL

        this.setTitle(name);//TITLE OF PANEL

        this.setLayout(new BorderLayout());//LAYOUT OF PANEL

        panelUp = new JPanel();//PANEL UP : USED TO HOLD TEXTS

        panelDown = new JPanel();//PANEL DOWN : USED TO HOLD TEXTFIELD AND SEND BUTTON

        input = new JTextField(20);//INPUT FILED

        send = new JButton("SEND");//SEND BUTTON

        scrollpane = new JScrollPane(panelUp);//SCROLL PANE FOR SCROLLING EFFECTS FOR TEXTS

        panelUp.setLayout(new GridLayout(0,1));//LAYOUT OF PANELUP

        scrollpane.setPreferredSize(new Dimension(500, 500));//SIZE OF SCROLL PANE

        panelDown.add(input);//ADD TEXTFIELD AND SEND BUTTON TO PANEL DOWN

        panelDown.add(send);

        //::::::::::ADD ALL COMPONENTS TO THE TOP LEVEL FRAME::::::::::::::://

        this.add(scrollpane,BorderLayout.CENTER);

        this.add(panelDown,BorderLayout.PAGE\_END);

        this.setVisible(true);

    }

    //INSETS USED TO LEAVE SPACES AROUND THE BORDER

    public Insets getInsets()

    {

        return new Insets(40,40,40,40);

    }

}

//:::::::::::::::::::::::::END OF USER INTERFACE CLASS::::::::::::::::::::://

//::::::::::::::::::::::::: THE HOST CLASS::::::::::::::::::::://  
class HOST implements Runnable

{

    public  int buffer\_size = 1024; //MAX BUFFER SIZE

    public  int RecieverPort;   //PORT ADDRESS OF RECIEVER HOST

    public  int SenderPort;  //PORT OF SENDER(THIS) HOST

    public  byte[] buffer1;  //BUFFER FOR SENDING THREAD

    public  byte[] buffer2;  //BUFFER FOR RECIEVIGN THREAD

    public  DatagramSocket socket;//DATAGRAM SOCKET

    public  DatagramPacket packet1,packet2;//DATAGRAM PACKETS FOR SENDING AND RECIEVING THREAD

    public  Thread thread;  //THREAD OBJECT

    public  UserInterface frame;//FRAME OBJECT

    public  InetAddress ipAddress; //IP ADDRESS OF THE DESTINATION HOST

    public HOST(String name,DatagramSocket socket,int r,int s,byte[] ip,UserInterface ui) throws Exception

    {

        //::::::::::::::::::::INITIALIZATIONS::::::::::::::::::://

        buffer1 = new byte[buffer\_size];

        buffer2 = new byte[buffer\_size];

        this.socket = socket;

        this.RecieverPort = r;//GET RECIEVER PORT

        this.SenderPort = s;//GET SENDER PORT

        this.frame = ui;

        this.ipAddress = InetAddress.getByAddress(ip);//GET IP ADDRESS

        //CREATING THREAD AND CALLING RUNNABLE

        this.thread = new Thread(this,name);

        thread.start();

    }

    @Override

    public void run() {

        try {

            if(thread.getName() == "SEND")//IF THREAD-NAME IS : "SEND" THEN CALL Send() METHOD

                Send();

            else if(thread.getName() == "RECIEVE")//IF THREAD-NAME IS : "RECIEVE" THEN CALL Recieve() METHOD

                Recieve();

            }catch(Exception e){}

    }

        //:::::::::::::METHOD USED TO SEND TEXT OT RECIEVER HOST:::::::::::::://

        public void Send() throws Exception

        {

            //CREATE NEW DATAGRAM PACKET

            packet1 = new DatagramPacket(buffer1, buffer\_size,ipAddress,RecieverPort);

 //ON BUTTON CLICK, SEND THE MESSAGE

            //ACTION LISTENER ON SENDING BUTTON

            frame.send.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    String msg = frame.input.getText();//GET THE STRING WRITTEN IN TEXTFIELD

                    frame.input.setText("");//RESET THE TEXTFIELD FIELD

                    if(!msg.isEmpty())//CHECK IF INPUT FIELD IS EMPTY

                    {

                        for(int i=0;i<msg.length();i++) {

                            buffer1[i] = (byte)msg.charAt(i);//STORE THE CHARACTERS OF THE STRING IN BUFFER

                        }

                        //:::::::PREPAIR THE LABLE WITH THE ABOVE STRING AND DISPLAY IT ON THE UI:::::::::://

                        JLabel TextSent = new JLabel(msg);

                        TextSent.setHorizontalAlignment(SwingConstants.RIGHT);

                        TextSent.setFont(new Font("Courier",Font.BOLD,15));

                        frame.panelUp.add(TextSent);

                        frame.revalidate();//REFRESH THE FRAME

                        packet1.setLength(msg.length());//SET THE LENGTH OF THE PACKET

                        try {

                            socket.send(packet1);   //SEND THE DATAGRAM PACKET TO THE INTENDED DESTINATION

                        } catch (IOException e1) {}

                    }

                }

            });

        }

        //::::::::::::::::METHOD USED TO RECIEVE AND DISPLAY TEXTS:::::::::::::::://

        public void Recieve() throws Exception

        {

            while(true) {

                //:::CREATE NEW DATAGRAM:::://

                packet2 = new DatagramPacket(buffer2, buffer2.length);

                socket.receive(packet2);//RECIEVE THE DATAGRAM PACKET

                int length = packet2.getLength();//GET TEH LENGTH OF THE DATAGRAM PACKET

                //DISPLAY THE MESSAGE ON THE SCREEN[UI]

                JLabel TextRec = new JLabel(new String(packet2.getData(),0,length));

                TextRec.setHorizontalAlignment(SwingConstants.LEFT);

                TextRec.setFont(new Font("Courier",Font.BOLD,15));

                frame.panelUp.add(TextRec);

                frame.revalidate();//REFRESH THE FRAME

            }

        }

}

public class Server {

    public static void main(String args[]) throws Exception {

        //::::GET THE SENDERS AND RECIEVERS PORT ADDRESS:::::::::::::://

        int RecieverPort = Integer.parseInt(JOptionPane.showInputDialog("ENTER RECIEVER'S PORT ADDRESS."));

        int SenderPort = Integer.parseInt(JOptionPane.showInputDialog("ENTER YOUR PORT ADDRESS."));

        //::::GET THE IP ADDRESS OF THE RECIEVER:::::::://

        String IPStringFormat[] = JOptionPane.showInputDialog("ENTER RECIEVER'S IP ADDRESS.").split("\\.");

        //:::NAME OF THE SENDER::::://

        String Name = JOptionPane.showInputDialog("ENTER YOUR NAME");

        //::::::CONVERT THE IP ADDRESS INTO BYTE FORMAT

        byte[] IPbyte = new byte[4];

        for(int i=0;i<IPStringFormat.length;i++)

        {

            IPbyte[i] =  (byte)Integer.parseInt(IPStringFormat[i]);

        }

        //CREATE THE OBJECT OF THE FRAME

        UserInterface frame = new UserInterface(Name);

        //CREATE THE SOCKET CONNECTION AND BIND IT ON SENDER(THIS) HOST'S PORT ADDRESS

        DatagramSocket socket = new DatagramSocket(SenderPort);

        HOST SEND = new HOST("SEND", socket,RecieverPort,SenderPort,IPbyte,frame);//SENDING THREAD

        HOST RECIEVE = new HOST("RECIEVE",socket,RecieverPort,SenderPort,IPbyte,frame);//RECIEVING THREAD

       //::::::::::::::DISPOSE THE WINDOW::::::::::::::::://

        frame.addWindowListener(new WindowAdapter() {

            @Override

            public void windowClosing(WindowEvent e) {

                socket.close();//CLOSE THE SOCKET

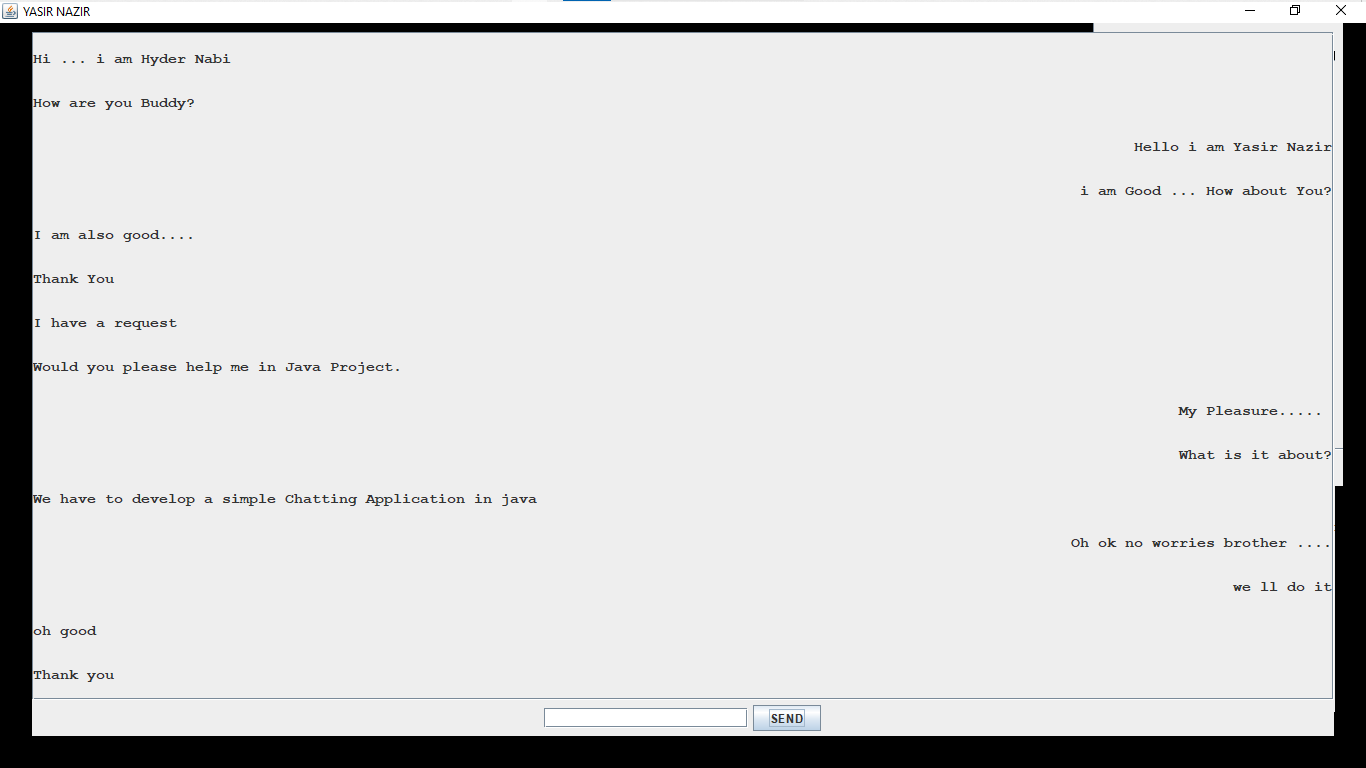
                frame.dispose();//DISPOSE THE FRAME

            }

        });

    }

 }

****

****