UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA FACULTAD DEINGENIERÍA EN SISTEMAS DE INFORMACIÓN

SISTEMAS OPERATIVOS II SECCION A
ING. SAUL OROZCO

# Manual Técnico



| GRUPO PROYECTO | Carnet |
|---|---|
| Pedro Daniel Vásquez Chamalé | 7690-12-9149 |
| Mario Fernando Castañeda Pérez | 7690-20-145 |
| Lester Navil Pérez Marroquin | 7690-20-14011 |
| Manuel Alejandro Sazo Linares | 7690-20-13585 |
| | |

# CODIGO FUENTE

1) SERVER

La clase server

```
//import com.sun.awt.AWTUtilities;
import java.awt.Color;
//import java.awt.Shape;
import java.awt.geom.RoundRectangle2D;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;       librerias
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 *
 * @author PÉDRO VASQUEZ
 */
public class Server extends javax.swing.JFrame implements Runnable{
ArrayList<Controles> lista2=new ArrayList<Controles>();
    /**
     * Crea un nuevo servidor
     */
    public Server() {
        initComponents();
        this.setLocationRelativeTo(null);
     //   Shape forma = new
RoundRectangle2D.Double(0,0,this.getBounds().width,this.getBounds().heigh
t,27,27);

     //   AWTUtilities.setWindowShape(this, forma);
        cerrar.setBackground(new Color(0,0,0,64));
        cerrar.setBorder(null);
        scroll.setOpaque(false);
        scroll.getViewport().setOpaque(false);
        scroll.setBorder(null);
        scroll.setViewportBorder(null);
        historial.setBorder(null);

        historial.setBackground(new Color(0,0,0,64));

    //    cerrar.setBackground(new Color(0,0,0,64));
        cerrar.setOpaque(false);
```

```
        cerrar.setContentAreaFilled(false);
        cerrar.setBorderPainted(false);
       //creando el hilo
       Thread mihilo = new Thread(this);
       mihilo.start();     //ejecutando el hilo
    }

    /**
     * Este método se llama desde dentro del constructor para
inicializar el formulario.
     * El contenido de este método siempre * es generado por el
Editor de formularios
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jFormattedTextField1 = new javax.swing.JFormattedTextField();
        cerrar = new javax.swing.JButton();
        scroll = new javax.swing.JScrollPane();
        historial = new javax.swing.JTextArea();
        jLabel1 = new javax.swing.JLabel();

        jFormattedTextField1.setText("jFormattedTextField1");


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setCursor(new java.awt.Cursor(java.awt.Cursor.MOVE_CURSOR));
        setMinimumSize(new java.awt.Dimension(1090, 500));
        setUndecorated(true);
        addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
            public void mouseDragged(java.awt.event.MouseEvent evt) {
                formMouseDragged(evt);
            }
        });
        addMouseListener(new java.awt.event.MouseAdapter() {
            public void mousePressed(java.awt.event.MouseEvent evt) {
                formMousePressed(evt);
            }
        });
        getContentPane().setLayout(null);

        cerrar.setFont(new java.awt.Font("Eras Bold ITC", 2, 18)); //
NOI18N
        cerrar.setForeground(new java.awt.Color(150, 191, 218));
        cerrar.setText("X");
        cerrar.setBorder(null);
        cerrar.setBorderPainted(false);
        cerrar.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        cerrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cerrarActionPerformed(evt);
```

```
            }
        });
        getContentPane().add(cerrar);
        cerrar.setBounds(1030, 10, 30, 22);

        historial.setColumns(20);
        historial.setForeground(new java.awt.Color(150, 191, 218));
        historial.setRows(5);
        historial.setCaretColor(new java.awt.Color(150, 191, 218));
        historial.setCursor(new
java.awt.Cursor(java.awt.Cursor.TEXT_CURSOR));
        historial.setSelectedTextColor(new java.awt.Color(150, 191,
218));
        scroll.setViewportView(historial);

        getContentPane().add(scroll);
        scroll.setBounds(0, 120, 1090, 380);

        jLabel1.setForeground(new java.awt.Color(255, 255, 255));
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/ima/fondo2.jpg"))); //
NOI18N
        jLabel1.setVerticalAlignment(javax.swing.SwingConstants.TOP);
        getContentPane().add(jLabel1);
        jLabel1.setBounds(0, 0, 1090, 500);

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void cerrarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_cerrarActionPerformed
        System.exit(0);

    }//GEN-LAST:event_cerrarActionPerformed
     private int xx;
        private int yy;
    private void formMousePressed(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_formMousePressed
        // TODO add your handling code here:


         xx=evt.getX();
         yy=evt.getY();
    }//GEN-LAST:event_formMousePressed

    private void formMouseDragged(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_formMouseDragged
         int x=evt.getXOnScreen();
         int y=evt.getYOnScreen();
         setLocation(x-xx,y-yy);
// TODO add your handling code here:
    }//GEN-LAST:event_formMouseDragged

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
```

```
        /* esto establece el aspecto y la sensación de
Nimbus */
        //

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Server.class.getName()).log(java.util.
logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Server.class.getName()).log(java.util.
logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Server.class.getName()).log(java.util.
logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Server.class.getName()).log(java.util.
logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

       /* Crear y mostrar el formulario */

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Server().setVisible(true);
            }
        });
    }

    // esta es la declaracion de variables
   - //GEN-BEGIN:variables
    private javax.swing.JButton cerrar;
    private javax.swing.JTextArea historial;
    private javax.swing.JFormattedTextField jFormattedTextField1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JScrollPane scroll;
    // Fin de la declaración de variables //GEN-END:variables

    @Override
    public void run() {
        //System.out.println("jejejej");
        try {
            //puerto que debe abrir
```

```
            ServerSocket servidor = new ServerSocket(1024); // que
esté a la escucha y abra ese puerto
            String nick,mensaje2,ip;
            ArrayList <String> listaIp4 = new ArrayList<String>();
//arrayList que alamcenará las ip conectadas

            Paquete1 paquete_recibido;
              Controles aux = new Controles();

            while(true){    //bucle infinito que acepta las
conexiones
            Socket misocket = servidor.accept(); //que acepte las
peticiones de conexion




            /*...........fin de deteccion online */
              //flujo de datos de entrada
            ObjectInputStream entrada =new
ObjectInputStream(misocket.getInputStream());

            paquete_recibido=(Paquete1) entrada.readObject();

            nick=paquete_recibido.getNick2();
            mensaje2=paquete_recibido.getMensaje();
            ip=paquete_recibido.getIp();
            int puertoCliente=9999;
            if(!mensaje2.equals(" Online")){
                String aux2=null;
                for(int i=0;i<lista2.size();i++){

if(lista2.get(i).getNombrecontrol().equalsIgnoreCase(paquete_recibido.get
Destinatario())){
                 aux2=lista2.get(i).getIpcontrol();
                  puertoCliente=lista2.get(i).getPuerto();
                  System.out.println("se quiere conectar con:
"+puertoCliente);
                }
                }
            historial.append("\n"+ nick+": "+mensaje2+"  para "+aux2);
            //ahora creamos un flujo de datos de entrada para
que sea capaz de recoger los datos
          //socket viaja la informacion de entrada

            /*DataInputStream entrada = new
DataInputStream(misocket.getInputStream());
            String mensaje = entrada.readUTF();
            historial.append("\n"+mensaje);*/
            System.out.println(puertoCliente);
            Socket enviaDestinatario = new Socket(ip,puertoCliente);
            ObjectOutputStream reenvio = new
ObjectOutputStream(enviaDestinatario.getOutputStream());
```

```
            reenvio.writeObject(paquete_recibido);


            reenvio.close();
            enviaDestinatario.close();
            misocket.close();
            }else{
                 /*++++++++++detección de usuarios onlines +++++
              almacenamos dentro de la variable localizacion la
direccion del cliente que se conectó
            */
      Controles hola = new Controles();

                    InetAddress localizacion =
misocket.getInetAddress();
                    int puerto = paquete_recibido.getPuerto();
                    String ipremota = localizacion.getHostAddress();
//almacenamos la direccion en string
                    aux.setIpcontrol(ipremota);
                    aux.setNombrecontrol(paquete_recibido.getNick2());
                    hola.setIpcontrol(ipremota);
                    hola.setNombrecontrol(nick);
                    hola.setPuerto(puerto);
                    lista2.add(hola);
                 //   lista2.add(aux);

                    historial.append("\n---------------------------------
------------------- -Conectado-----------------------------------------
-----------");
                    historial.append("\n"+ nick+" : ip "+ipremota +"
conectado desde el puerto "+puerto);

                    historial.append("\n\n");
                  //  listaIp4.add(ipremota);
                    //paquete_recibido.setDirIps(listaIp4);
                    paquete_recibido.setControl(lista2);

                   for(int i=0;i<lista2.size();i++){
                  // System.out.println("array: "+z);
                    Socket enviaDestinatario = new
Socket(lista2.get(i).getIpcontrol(),lista2.get(i).getPuerto());
                    ObjectOutputStream reenvio = new
ObjectOutputStream(enviaDestinatario.getOutputStream());
                     reenvio.writeObject(paquete_recibido);
                     reenvio.close();
                      enviaDestinatario.close();
                      misocket.close();
                  }
                  }
            }
        } catch (IOException ex) {
            System.out.println(ex.getMessage());
            Logger.getLogger(Server.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(Server.class.getName()).log(Level.SEVERE,
```

```
null, ex);
        }
    }
}

//constructor  de paquete o mensajes

class Paquete1 implements Serializable{  //atributos de clase
private String mensaje;
private String ip;
private String nick2;
private ArrayList<String> dirIps;
private ArrayList<Controles> control;
private int puerto;
private String destinatario;

    public String getDestinatario() {// asignación de valores de los
parámetros
        return destinatario;
    }

    public void setDestinatario(String destinatario) {
        this.destinatario = destinatario;
    }
    public int getPuerto() {
        return puerto;
    }

    public void setPuerto(int puerto) {
        this.puerto = puerto;
    }

    public ArrayList<Controles> getControl() {
        return control;
    }

    public void setControl(ArrayList<Controles> control) {
        this.control = control;
    }


    public ArrayList<String> getDirIps() {
        return dirIps;
    }

    public void setDirIps(ArrayList<String> dirIps) {
        this.dirIps = dirIps;
    }




    public String getMensaje() {
        return mensaje;
    }
```

```
    public void setMensaje(String mensaje) {
        this.mensaje = mensaje;
    }

    public String getIp() {
        return ip;
    }

    public void setIp(String ip) {
        this.ip = ip;
    }

    public String getNick2() {
        return nick2;
    }

    public void setNick2(String nick2) {
        this.nick2 = nick2;
    }

  //constructor  de controles

}
class Controles implements Serializable{
private String ipcontrol;
private String nombrecontrol;
private int puerto;

    public int getPuerto() {
        return puerto;
    }

    public void setPuerto(int puerto) {
        this.puerto = puerto;
    }

    public String getIpcontrol() {
        return ipcontrol;
    }

    public void setIpcontrol(String ipcontrol) {
        this.ipcontrol = ipcontrol;
    }

    public String getNombrecontrol() {
        return nombrecontrol;
    }

    public void setNombrecontrol(String nombrecontrol) {
        this.nombrecontrol = nombrecontrol;
    }

}
```

2) CLIENTE

La clase cliente representa a los usuarios que estaran interactuando en el chat
Se utiliza en programas java para administrar los recursos.

```
import java.awt.Color;
import java.awt.Shape;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.geom.RoundRectangle2D;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.net.InetAddress;                        estas son librerías
import java.net.ServerSocket;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;



/**
 *
 * @author Grupo 9 S.O.
 * Manuel Sazo
 * Pedro Vasquez
 * Lester Perez
 * Mario Castañeda
 */

public class Cliente extends javax.swing.JFrame implements Runnable{
    ArrayList<Controles> listaClientes=new ArrayList<Controles>();
     private static int puertoCliente;
    private static String NombreCliente;

    public String getNombre() {
        return NombreCliente;
    }

    public void setNombre(String nombre) {
        this.NombreCliente = nombre;
    }

    public int getPuerto() {
        return puertoCliente;
    }

    public void setPuerto(int puerto) {
```

```
        this.puertoCliente = puerto;
    }
    //Creando clientes

    public Cliente() {
        try {
            String NombreAux = JOptionPane.showInputDialog("Nombre: ");
            Socket misocket = new Socket("192.168.1.13",1024);
            //Se debe de colocar la ip del server proporcionada por la
maquina que tiene instalado el servidor.
            ServerSocket auxiliar = new ServerSocket(0);
            int puertoDisponible=auxiliar.getLocalPort();
            auxiliar.close();

            Paquete1 datos = new Paquete1();
            NombreCliente=NombreAux;
            datos.setNick2(NombreCliente);
            datos.setPuerto(puertoDisponible); // los datos del cliente
            puertoCliente=puertoDisponible;
            System.out.println(puertoCliente);
            datos.setMensaje(" Online");

            ObjectOutputStream salida = new//llamado salida para el
socket ObjectOutputStream(misocket.getOutputStream());
            salida.writeObject(datos);//salida para cada dato en el flujo
de salida
            misocket.close();//cierre del socket al estar offline la
conexión del usuario.
        } catch (IOException ex) {
            historial.append(ex.getMessage());

        }

        initComponents();
        mensaje.setBackground(new Color(0,0,0,64));
        this.setLocationRelativeTo(null);



        scroll.setOpaque(false);
        scroll.getViewport().setOpaque(false);
        scroll.setBorder(null);
        scroll.setViewportBorder(null);
        historial.setBorder(null);
        historial.setBackground(new Color(0,0,0,64));


        cerrar1.setOpaque(false);
        cerrar1.setContentAreaFilled(false);
        cerrar1.setBorderPainted(false);
        enviar.setOpaque(false);
        enviar.setContentAreaFilled(false);
        enviar.setBorderPainted(false);
         nick.setText(NombreCliente);
       // ip destino.addItem("127.0.0.1");
```

```
        Thread mihilo = new Thread(this);
        mihilo.start();

    }

//----------------------------------envia señal de clientes conectados----
--------------------

    /
     ** Este método se llama desde dentro del constructor para
inicializar el formulario..
     *El contenido de este método es siempre
      regenerado por el Editor de formularios     */

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jLabel2 = new javax.swing.JLabel();  //es una etiqueta que se
utiliza  para mostrar información en la interfaz
        mensaje = new javax.swing.JTextField();//es un campo
        num = new javax.swing.JLabel();
        nick = new javax.swing.JLabel();
        scroll = new javax.swing.JScrollPane();
        historial = new javax.swing.JTextArea();
        ipdestino = new javax.swing.JComboBox<>();
        cerrar1 = new javax.swing.JButton(); //es un botón para cerrar la
interfaz.
        jLabel3 = new javax.swing.JLabel();//otra etiqueta
        enviar = new javax.swing.JButton();// un botón para enviar un
mensaje
        jLabel4 = new javax.swing.JLabel();// otra etiqueta de texto
        jLabel1 = new javax.swing.JLabel();

        jLabel2.setText("jLabel2");


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("servidor");
        setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        setMinimumSize(new java.awt.Dimension(760, 430));
        setUndecorated(true);
        setSize(new java.awt.Dimension(760, 400));
        addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
            public void mouseDragged(java.awt.event.MouseEvent evt) {

                formMouseDragged(evt);
            }
        });
        addMouseListener(new java.awt.event.MouseAdapter() {
            public void mousePressed(java.awt.event.MouseEvent evt) {
                formMousePressed(evt);
            }
```

```
        });
        getContentPane().setLayout(null);

        mensaje.setForeground(new java.awt.Color(150, 191, 218));
        mensaje.setBorder(null);
        mensaje.setCaretColor(new java.awt.Color(150, 191, 218));
        mensaje.setSelectedTextColor(new java.awt.Color(150, 191, 218));
        mensaje.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mensajeActionPerformed(evt);
            }
        });
        getContentPane().add(mensaje);
        mensaje.setBounds(30, 290, 560, 110);

        num.setFont(new java.awt.Font("Haettenschweiler", 1, 24)); //
NOI18N
        num.setForeground(new java.awt.Color(255, 255, 255));
        getContentPane().add(num);
        num.setBounds(350, 30, 50, 30);

        nick.setBackground(new java.awt.Color(255, 255, 255));
        nick.setFont(new java.awt.Font("Sitka Text", 1, 14)); // NOI18N
        nick.setForeground(new java.awt.Color(255, 255, 255));
        nick.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        nick.setVerticalAlignment(javax.swing.SwingConstants.TOP);
        nick.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
"", javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Segoe UI Black", 1, 14), new java.awt.Color(255, 255,
255))); // NOI18N
        getContentPane().add(nick);
        nick.setBounds(30, 30, 110, 30);

        historial.setColumns(20);
        historial.setForeground(new java.awt.Color(150, 191, 218));
        historial.setRows(5);
        historial.setCaretColor(new java.awt.Color(150, 191, 218));
        historial.setFocusable(false);
        historial.setSelectedTextColor(new java.awt.Color(150, 191,
218));
        scroll.setViewportView(historial);

        getContentPane().add(scroll);
        scroll.setBounds(30, 90, 710, 180);

        ipdestino.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ipdestinoActionPerformed(evt);
            }
        });
        getContentPane().add(ipdestino);
        ipdestino.setBounds(490, 50, 121, 22);

        cerrar1.setBackground(new java.awt.Color(0, 0, 0));
        cerrar1.setFont(new java.awt.Font("Sitka Text", 3, 18)); //
NOI18N
```

```
        cerrar1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/ima/Close_Icon_Dark_icon-
icons.com_69143.png"))); // NOI18N
        cerrar1.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        cerrar1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cerrar1ActionPerformed(evt);
            }
        });
        getContentPane().add(cerrar1);
        cerrar1.setBounds(640, 10, 110, 50);

        jLabel3.setBackground(new java.awt.Color(255, 255, 255));
        jLabel3.setFont(new java.awt.Font("Sitka Text", 1, 14)); //
NOI18N
        jLabel3.setForeground(new java.awt.Color(255, 255, 255));

jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel3.setText("Usuarios conectados");
        getContentPane().add(jLabel3);
        jLabel3.setBounds(160, 30, 170, 30);

        enviar.setForeground(new java.awt.Color(255, 255, 255));
        enviar.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/ima/email_send_22054.png")
)); // NOI18N
        enviar.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        enviar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                enviarActionPerformed(evt);
            }
        });
        getContentPane().add(enviar);
        enviar.setBounds(600, 290, 150, 110);

        jLabel4.setFont(new java.awt.Font("Sitka Text", 1, 14)); //
NOI18N
        jLabel4.setForeground(new java.awt.Color(255, 255, 255));
        jLabel4.setText("Enviar a:");
        getContentPane().add(jLabel4);
        jLabel4.setBounds(510, 20, 90, 20);


jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/ima/fondo.jpeg"))); //
NOI18N
        getContentPane().add(jLabel1);
        jLabel1.setBounds(0, 0, 760, 420);

        pack();
    }// </editor-fold>//GEN-END:initComponents
        private int xx;
        private int yy;
    private void formMousePressed(java.awt.event.MouseEvent evt) {//GEN-
```

```
FIRST:event_formMousePressed
         xx=evt.getX();
         yy=evt.getY();

        // TODO add your handling code here:
    }//GEN-LAST:event_formMousePressed

    private void formMouseDragged(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_formMouseDragged
         int x=evt.getXOnScreen();
         int y=evt.getYOnScreen();
         setLocation(x-xx,y-yy);
        // TODO add your handling code here:
    }//GEN-LAST:event_formMouseDragged

// es la accion de   generar como un evento para la ip  de destino
    private void ipdestinoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_ipdestinoActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_ipdestinoActionPerformed

// esto es la acción de cerrar esta ejecución
    private void cerrar1ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_cerrar1ActionPerformed
        System.exit(0);
        // TODO add your handling code here:
    }//GEN-LAST:event_cerrar1ActionPerformed

// es la ejecución de enviar
    private void enviarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_enviarActionPerformed

           //Muestra ip del servidor y el puerto que van a
estar abiertos
        try {
            Socket miSocket =new Socket("192.168.1.13",1024);
            Paquete1 datos = new Paquete1();
            datos.setNick2(nick.getText());

datos.setDestinatario(ipdestino.getSelectedItem().toString());
            String aux;
            aux=ipdestino.getSelectedItem().toString();
            for(int i=0;i<listaClientes.size();i++){

if(aux.equalsIgnoreCase(listaClientes.get(i).getNombrecontrol())){
                    datos.setIp(listaClientes.get(i).getIpcontrol());
                }

            }

            historial.append("\n-"+mensaje.getText());
            datos.setMensaje(mensaje.getText());

            ObjectOutputStream salida = new
ObjectOutputStream(miSocket.getOutputStream());
            salida.writeObject(datos);
            mensaje.setText(null);
```

```
            miSocket.close();

        } catch (IOException ex) {
            System.out.println(ex.getMessage());

        }

        // TODO add your handling code here:
    }//GEN-LAST:event_enviarActionPerformed


    private void mensajeActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_mensajeActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_mensajeActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Cliente.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Cliente.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Cliente.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Cliente.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Cliente().setVisible(true);



            }
```

```
        });
    }

    // Variables declarada //GEN-BEGIN:variables
    private javax.swing.JButton cerrar1;
    private javax.swing.JButton enviar;
    private javax.swing.JTextArea historial;
    private javax.swing.JComboBox<String> ipdestino;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JTextField mensaje;
    private javax.swing.JLabel nick;
    private javax.swing.JLabel num;
    private javax.swing.JScrollPane scroll;
    //fin de las variables declaradas  //GEN-END:variables

    @Override
    public void run(){
        try {
            System.out.println(puertoCliente);
            ServerSocket servidor_cliente = new
ServerSocket(puertoCliente);// aquí el server está llamando a el cliente


            ArrayList<String> ipsMenu = new ArrayList<String>();
            Socket cliente;
            Paquete1 paquete_recibido;
            while(true){
            cliente = servidor_cliente.accept();
            ObjectInputStream entrada = new
ObjectInputStream(cliente.getInputStream());
            paquete_recibido= (Paquete1) entrada.readObject();//se crea
un object llamado entrada

            if(!paquete_recibido.getMensaje().equalsIgnoreCase("
Online")){//verifica el paquete  recibido

            historial.append("\n\t\t"+paquete_recibido.getNick2()+ ": "+
paquete_recibido.getMensaje());
            } else{// Historial de los paquetes recibidos.


            Controles hola = new Controles();
            ipsMenu=paquete_recibido.getDirIps();
            String nombre=paquete_recibido.getNick2();
            listaClientes=paquete_recibido.getControl();


            String conected = String.valueOf(listaClientes.size());
            num.setText(conected);
        //borra el arrayList desactualizado
         ipdestino.removeAllItems();

            for(int i=0;i<listaClientes.size();i++){
            ipdestino.addItem(listaClientes.get(i).getNombrecontrol());
```

```
            }
            }


            }
            } catch (IOException ex) {
            Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE,
null, ex);
        }
        }


}
//debemos serializar nuestra clase para poder convertir las
instancias en bytes para el transporte  por la red

class Paquete1 implements Serializable{
private String mensaje;
private String ip;
private String nick2;
private ArrayList<String> dirIps;
private ArrayList<Controles> control;
private int puerto;
private String destinatario;

    public String getDestinatario() {
        return destinatario;
    }

    public void setDestinatario(String destinatario) {
        this.destinatario = destinatario;
    }

    public int getPuerto() {
        return puerto;
    }

    public void setPuerto(int puerto) {
        this.puerto = puerto;
    }

    public ArrayList<Controles> getControl() {
        return control;
    }

    public void setControl(ArrayList<Controles> control) {
        this.control = control;
    }
    public ArrayList<String> getDirIps() {
        return dirIps;
    }

    public void setDirIps(ArrayList<String> dirIps) {
        this.dirIps = dirIps;
```

```
    }



    public String getMensaje() {
        return mensaje;
    }

    public void setMensaje(String mensaje) {
        this.mensaje = mensaje;
    }

    public String getIp() {
        return ip;
    }

    public void setIp(String ip) {
        this.ip = ip;
    }

    public String getNick2() {
        return nick2;
    }

    public void setNick2(String nick2) {
        this.nick2 = nick2;
    }



}
class EnvioOnline extends WindowAdapter{ //implementa todas los
metodos pertenecientes a una interfaz(listener)
/*metodo que se ejecuta cuando se abre la ventana*/

    String nickx,ipserver;
    public void hola (String s){
      nickx=s;
    }
    public void hola2 (String e){
      ipserver=e;
    }

    public void windowOpened(WindowEvent e){
        try {
            //ip del server

            Socket misocket = new Socket("192.168.1.13",1024);

            Paquete1 datos = new Paquete1();

            datos.setNick2(nickx);
            datos.setMensaje(" Online");

            ObjectOutputStream salida = new
ObjectOutputStream(misocket.getOutputStream());
            salida.writeObject(datos);
```

```
            misocket.close();

        } catch (IOException ex) {

Logger.getLogger(EnvioOnline.class.getName()).log(Level.SEVERE, null,
ex);
        }
}
}
class Controles implements Serializable{
private String ipcontrol;
private String nombrecontrol;
private int puerto;

    public int getPuerto() {
        return puerto;
    }

    public void setPuerto(int puerto) {
        this.puerto = puerto;
    }

    public String getIpcontrol() {
        return ipcontrol;
    }

    public void setIpcontrol(String ipcontrol) {
        this.ipcontrol = ipcontrol;
    }

    public String getNombrecontrol() {
        return nombrecontrol;
    }

    public void setNombrecontrol(String nombrecontrol) {
        this.nombrecontrol = nombrecontrol;
    }

}
```