**Instituto Politécnico Nacional**

Escuela Superior de Cómputo

Reporte Práctica Chat Multicast

**Integrantes del Equipo:**

Barrera Pérez Carlos Tonatihu

Castillo Reyes Juan Daniel

**Grupo:** 2CM11

**Unidad de Aprendizaje:** Aplicaciones para Comunicaciones en Red

**Profesor:** Axel Ernesto Moreno Cervantes

Introducción

Muchas empresas hacen uso de Internet para ofrecer sus productos y servicios, ya que a través de este medio pueden estar en contacto directo con los clientes potenciales. Algunas de las principales ventajas de Internet como medio para realizar negocios son la cobertura a nivel global, así como la disponibilidad del servicio los 365 días, las 24 hrs. del día.

Sin embargo, estas ventajas se convierten también en un reto para las empresas, pues se debe proporcionar a los clientes una vía de comunicación directa para resolver las dudas de los clientes potenciales, atender sus peticiones, así como proporcionarles información adicional a la expuesta en sus portales. Existen diversos recursos que pueden ser utilizados, como los foros o el correo electrónico, pero ninguno de ellos permite la comunicación en tiempo real.

El chat es un excelente recurso para este tipo de propósito, ya que además de ser una vía de comunicación en tiempo real, permite comunicar a dos o más usuarios entre sí. Típicamente los chats se han implementado haciendo uso de sockets de flujo, pero existen otras alternativas que también merece la pena probar, tales como los sockets de datagrama o los de multidifusión

Desarrollo

Esta practica consistió en el desarrollo de un chat grupal utilizando sockets de datagrama mediante multicast, dicho chat tiene la posibilidad de enviar imágenes y emojis a todos los usuarios que se encuentren conectados en algún momento, además tiene la posibilidad de enviar mensajes privados a cada usuario conectado.

Fue necesario crear dos proyectos de Java separados, el primero para los clientes que se conectaran y el segundo un simple servidor que permitirá llevar un registro de todos los usuarios conectados.

# Cliente

La parte del cliente se divide en dos tipos de archivos diferentes los cuales son, archivos destinados a la interfaz de usuario y archivos de la lógica de conexión entre sockets.

## Interfaz

La interfaz consta de dos clases, una para el inicio de sesión de usuario u otro para la vista general del chat que se tendrá a lo largo de la ejecución del programa.

Chat.java

|  |
| --- |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package interfaz;  import logica.Enviar;  import logica.Mensaje;  import logica.MulticastUtilidades;  import logica.Recibir;  import javax.imageio.ImageIO;  import javax.swing.\*;  import javax.swing.text.BadLocationException;  import javax.swing.text.Element;  import javax.swing.text.StyleConstants;  import javax.swing.text.html.HTML;  import javax.swing.text.html.HTMLDocument;  import javax.swing.text.html.HTMLEditorKit;  import javax.swing.text.html.StyleSheet;  import java.awt.\*;  import java.io.File;  import java.io.IOException;  import java.net.InetAddress;  import java.net.MulticastSocket;  import java.util.ArrayList;  import java.util.Objects;  import java.util.logging.Level;  import java.util.logging.Logger;  /\*\*  \* @author tona  \*/  public class Chat extends javax.swing.JFrame implements MulticastUtilidades {  private static Element body = null;  private static HTMLDocument doc;  private File imagen;  private Enviar enviar;  private static String nickname;  private static DefaultListModel modelo;  private static String dest;  /\*\*  \* Creates new form Chat  \*/  public Chat() {  initComponents();  modelo = new DefaultListModel();  listUsuarios.setModel(modelo);  obtenerBody();  }  public Chat(String nickname) {  this();  Chat.nickname = nickname;  crearSocket();  enviar.enviarAnuncio(Chat.nickname);  try {  ponerIconos();  } catch (IOException e) {  e.printStackTrace();  }  }  private void ponerIconos() throws IOException {  int dimension = btnCaquita.getHeight() - 10;  Image iconoBoton = ImageIO.read(Mensaje.SADNESS);  iconoBoton = iconoBoton.getScaledInstance(dimension, dimension, Image.SCALE\_SMOOTH);  btnTriste.setIcon(new ImageIcon(iconoBoton));  btnTriste.setText("");  iconoBoton = ImageIO.read(Mensaje.ANGER);  iconoBoton = iconoBoton.getScaledInstance(dimension, dimension, Image.SCALE\_SMOOTH);  btnEnojado.setIcon(new ImageIcon(iconoBoton));  btnEnojado.setText("");  iconoBoton = ImageIO.read(Mensaje.HEART);  iconoBoton = iconoBoton.getScaledInstance(dimension, dimension, Image.SCALE\_SMOOTH);  btnCorazon.setIcon(new ImageIcon(iconoBoton));  btnCorazon.setText("");  iconoBoton = ImageIO.read(Mensaje.POOP);  iconoBoton = iconoBoton.getScaledInstance(dimension, dimension, Image.SCALE\_SMOOTH);  btnCaquita.setIcon(new ImageIcon(iconoBoton));  btnCaquita.setText("");  iconoBoton = ImageIO.read(Mensaje.SMILE);  iconoBoton = iconoBoton.getScaledInstance(dimension, dimension, Image.SCALE\_SMOOTH);  btnFeliz.setIcon(new ImageIcon(iconoBoton));  btnFeliz.setText("");  }  /\*\*  \* This method is called from within the constructor to initialize the form. WARNING: Do NOT  \* modify this code. The content of this method is always regenerated by the Form Editor.  \*/  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jLabel1 = new javax.swing.JLabel();  jScrollPane2 = new javax.swing.JScrollPane();  panelMensajes = new javax.swing.JEditorPane();  btnCargar = new javax.swing.JButton();  btnCaquita = new javax.swing.JButton();  btnFeliz = new javax.swing.JButton();  btnEnojado = new javax.swing.JButton();  btnCorazon = new javax.swing.JButton();  btnTriste = new javax.swing.JButton();  btnEnviar = new javax.swing.JButton();  jScrollPane3 = new javax.swing.JScrollPane();  txtAreaMensaje = new javax.swing.JTextArea();  jLabel2 = new javax.swing.JLabel();  jLabel3 = new javax.swing.JLabel();  btnVolver = new javax.swing.JButton();  jScrollPane1 = new javax.swing.JScrollPane();  listUsuarios = new javax.swing.JList<>();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  setTitle("Super Chat Grupal");  setResizable(false);  jLabel1.setText("Super chat grupal");  panelMensajes.setEditable(false);  panelMensajes.setContentType("text/html"); // NOI18N  panelMensajes.setEditorKit(new HTMLEditorKit());  panelMensajes.setText("<!DOCTYPE html>\n<html>\n <head>\n\n </head>\n <body></body>\n</html>\n");  panelMensajes.setAutoscrolls(false);  jScrollPane2.setViewportView(panelMensajes);  btnCargar.setText("Cargar Imagen");  btnCargar.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnCargarActionPerformed(evt);  }  });  btnCaquita.setText("<:3");  btnCaquita.setToolTipText("");  btnCaquita.setMaximumSize(new java.awt.Dimension(31, 31));  btnCaquita.setMinimumSize(new java.awt.Dimension(31, 31));  btnCaquita.setPreferredSize(new java.awt.Dimension(31, 31));  btnCaquita.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnCaquitaActionPerformed(evt);  }  });  btnFeliz.setText("=)");  btnFeliz.setMaximumSize(new java.awt.Dimension(31, 31));  btnFeliz.setMinimumSize(new java.awt.Dimension(31, 31));  btnFeliz.setPreferredSize(new java.awt.Dimension(31, 31));  btnFeliz.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnFelizActionPerformed(evt);  }  });  btnEnojado.setText(">=|");  btnEnojado.setMaximumSize(new java.awt.Dimension(31, 31));  btnEnojado.setMinimumSize(new java.awt.Dimension(31, 31));  btnEnojado.setPreferredSize(new java.awt.Dimension(31, 31));  btnEnojado.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnEnojadoActionPerformed(evt);  }  });  btnCorazon.setText("<3");  btnCorazon.setMaximumSize(new java.awt.Dimension(31, 31));  btnCorazon.setMinimumSize(new java.awt.Dimension(31, 31));  btnCorazon.setPreferredSize(new java.awt.Dimension(31, 31));  btnCorazon.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnCorazonActionPerformed(evt);  }  });  btnTriste.setText("='(");  btnTriste.setMaximumSize(new java.awt.Dimension(31, 31));  btnTriste.setMinimumSize(new java.awt.Dimension(31, 31));  btnTriste.setPreferredSize(new java.awt.Dimension(31, 31));  btnTriste.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnTristeActionPerformed(evt);  }  });  btnEnviar.setText("Enviar");  btnEnviar.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnEnviarActionPerformed(evt);  }  });  txtAreaMensaje.setColumns(10);  txtAreaMensaje.setLineWrap(true);  txtAreaMensaje.setRows(5);  txtAreaMensaje.setWrapStyleWord(true);  jScrollPane3.setViewportView(txtAreaMensaje);  jLabel2.setText("Usuarios conectados");  jLabel3.setText("Escribe un mensaje");  btnVolver.setText("Volver al chat grupal");  btnVolver.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnVolverActionPerformed(evt);  }  });  listUsuarios.addMouseListener(new java.awt.event.MouseAdapter() {  public void mouseClicked(java.awt.event.MouseEvent evt) {  listUsuariosMouseClicked(evt);  }  });  jScrollPane1.setViewportView(listUsuarios);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED\_SIZE, 291, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(btnCargar, javax.swing.GroupLayout.DEFAULT\_SIZE, 157, Short.MAX\_VALUE)  .addComponent(btnEnviar, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))  .addComponent(jScrollPane2)  .addGroup(layout.createSequentialGroup()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jLabel1)  .addComponent(jLabel3)  .addGroup(layout.createSequentialGroup()  .addComponent(btnCaquita, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(btnFeliz, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(btnEnojado, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(btnCorazon, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(btnTriste, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))  .addGap(0, 0, Short.MAX\_VALUE)))  .addGap(18, 18, 18)  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 176, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel2)  .addComponent(btnVolver, javax.swing.GroupLayout.PREFERRED\_SIZE, 176, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap())  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel1)  .addComponent(jLabel2))  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)  .addGroup(layout.createSequentialGroup()  .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 347, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  .addComponent(jLabel3)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))  .addGroup(layout.createSequentialGroup()  .addGap(384, 384, 384)  .addComponent(btnCargar)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 13, Short.MAX\_VALUE)  .addComponent(btnEnviar))  .addComponent(jScrollPane1))  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(btnVolver)  .addComponent(btnTriste, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(btnCorazon, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(btnEnojado, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(btnFeliz, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(btnCaquita, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  );  pack();  }// </editor-fold>//GEN-END:initComponents  //GEN-FIRST:event\_btnCargarActionPerformed  private void btnCargarActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  JFileChooser jfc = new JFileChooser();  jfc.setDialogTitle("Selecciona una imagen a enviar");  if (jfc.showDialog(this, "Seleccionar") == JFileChooser.APPROVE\_OPTION) {  imagen = jfc.getSelectedFile();  JOptionPane.showMessageDialog(this, "Imagen lista para enviar");  btnCargar.setText("Imagen en espera");  }  }//GEN-LAST:event\_btnCargarActionPerformed  //GEN-FIRST:event\_btnVolverActionPerformed  private void btnVolverActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  listUsuarios.clearSelection();  dest = null;  }//GEN-LAST:event\_btnVolverActionPerformed  // <editor-fold defaultstate="collapsed" desc="Emoji">//GEN-BEGIN:emojibuttons  //GEN-FIRST:event\_btnCorazonActionPerformed  private void btnCorazonActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  agregarEmoji(obtenerMensaje() + " <3 ");  }//GEN-LAST:event\_btnCorazonActionPerformed  //GEN-FIRST:event\_btnCaquitaActionPerformed  private void btnCaquitaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  agregarEmoji(obtenerMensaje() + " :poop: ");  }//GEN-LAST:event\_btnCaquitaActionPerformed  //GEN-FIRST:event\_btnFelizActionPerformed  private void btnFelizActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  agregarEmoji(obtenerMensaje() + " =) ");  }//GEN-LAST:event\_btnFelizActionPerformed  //GEN-FIRST:event\_btnEnojadoActionPerformed  private void btnEnojadoActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  agregarEmoji(obtenerMensaje() + " >=| ");  }//GEN-LAST:event\_btnEnojadoActionPerformed  //GEN-FIRST:event\_btnTristeActionPerformed  private void btnTristeActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  agregarEmoji(obtenerMensaje() + " ='( ");  }//GEN-LAST:event\_btnTristeActionPerformed  // </editor-fold>//GEN-END:emojiButtons  //GEN-FIRST:event\_btnEnviarActionPerformed  private void btnEnviarActionPerformed(java.awt.event.ActionEvent evt) {  try {  // TODO add your handling code here:  Mensaje msj = new Mensaje();  msj.setUsuario(nickname);  msj.setMensaje(obtenerMensaje());  if (dest != null)  msj.setDestinatario(dest);  if (imagen != null) {  msj.setTipoMensaje(Mensaje.IMAGEN);  btnCargar.setText("Cargar Imagen");  enviar.enviarImagen(imagen, msj);  imagen = null;  } else {  enviar.enviarMensaje(msj);  }  txtAreaMensaje.setText("");  } catch (IOException ex) {  Logger.getLogger(Chat.class.getName()).log(Level.SEVERE, null, ex);  }  }//GEN-LAST:event\_btnEnviarActionPerformed  //GEN-FIRST:event\_listUsuariosMouseClicked  private void listUsuariosMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  JList jlist = (JList) evt.getSource();  if (evt.getClickCount() == 1) {  int index = jlist.locationToIndex(evt.getPoint());  System.out.println("Destinatario: "+ modelo.get(index));  dest = (String) modelo.get(index);  }  }//GEN-LAST:event\_listUsuariosMouseClicked  // Variables declaration - do not modify//GEN-BEGIN:variables  private javax.swing.JButton btnCaquita;  private javax.swing.JButton btnCargar;  private javax.swing.JButton btnCorazon;  private javax.swing.JButton btnEnojado;  private javax.swing.JButton btnEnviar;  private javax.swing.JButton btnFeliz;  private javax.swing.JButton btnTriste;  private javax.swing.JButton btnVolver;  private javax.swing.JLabel jLabel1;  private javax.swing.JLabel jLabel2;  private javax.swing.JLabel jLabel3;  private javax.swing.JScrollPane jScrollPane1;  private javax.swing.JScrollPane jScrollPane2;  private javax.swing.JScrollPane jScrollPane3;  private javax.swing.JList<String> listUsuarios;  private javax.swing.JEditorPane panelMensajes;  private javax.swing.JTextArea txtAreaMensaje;  // End of variables declaration//GEN-END:variables  private String obtenerMensaje() {  return txtAreaMensaje.getText();  }  private void obtenerBody() {  StyleSheet styleSheet = new StyleSheet();  HTMLEditorKit kit = (HTMLEditorKit) panelMensajes.getEditorKit();  styleSheet.addRule("div {max-width:295px; word-wrap:break-word; overflow: hidden; width:295px;}");  kit.setStyleSheet(styleSheet);  panelMensajes.setEditorKit(kit);  doc = (HTMLDocument) kit.createDefaultDocument();  panelMensajes.setDocument(doc);  Element[] roots = doc.getRootElements();  for (int i = 0; i < roots[0].getElementCount(); i++) {  Element element = roots[0].getElement(i);  System.out.println(element.getAttributes().getAttribute(StyleConstants.NameAttribute));  if (element.getAttributes().getAttribute(StyleConstants.NameAttribute) == HTML.Tag.BODY) {  body = element;  break;  }  }  }  private void crearSocket() {  try {  InetAddress grupo = InetAddress.getByName(DIRECCION);  MulticastSocket multicastSocket = new MulticastSocket(PUERTO);  multicastSocket.joinGroup(grupo);  multicastSocket.setReuseAddress(true);  Recibir recibir = new Recibir(multicastSocket, nickname);  enviar = new Enviar(multicastSocket, grupo, PUERTO);  new Thread(recibir).start();  } catch (IOException e) {  e.printStackTrace();  }  }  public static void agregarMensaje(Mensaje msj) throws IOException, BadLocationException {  if (Objects.equals(msj.getDestinatario(), dest) || msj.getDestinatario() == null || Objects.equals(msj.getDestinatario(), nickname) )  doc.insertBeforeEnd(body, msj.construirMensaje());  }  private void agregarEmoji(String msj) {  txtAreaMensaje.setText(msj);  }  public static void agregarUsuarioLista(String usuario) {  modelo.addElement(usuario);  }  public static void cargarListaConectados(ArrayList<String> conectados) {  modelo.clear();  for (String usuario : conectados)  agregarUsuarioLista(usuario);  }  } |

Login.java

|  |
| --- |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package interfaz;  /\*\*  \* @author tona  \*/  public class Login extends javax.swing.JFrame {  /\*\*  \* Creates new form Login  \*/  private Login() {  initComponents();  }  /\*\*  \* This method is called from within the constructor to initialize the form. WARNING: Do NOT  \* modify this code. The content of this method is always regenerated by the Form Editor.  \*/  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jLabel1 = new javax.swing.JLabel();  labelNickname = new javax.swing.JLabel();  txtNickname = new javax.swing.JTextField();  btnConectarse = new javax.swing.JButton();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  setTitle("Super Chat");  jLabel1.setText("Inicio de Sesión");  labelNickname.setText("Nickname:");  btnConectarse.setText("Conectarse");  btnConectarse.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnConectarseActionPerformed(evt);  }  });  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addComponent(jLabel1)  .addGap(0, 0, Short.MAX\_VALUE))  .addGroup(layout.createSequentialGroup()  .addComponent(labelNickname)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(txtNickname))  .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()  .addGap(0, 174, Short.MAX\_VALUE)  .addComponent(btnConectarse)))  .addContainerGap())  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addComponent(jLabel1)  .addGap(18, 18, 18)  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(labelNickname)  .addComponent(txtNickname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(18, 18, 18)  .addComponent(btnConectarse)  .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  );  pack();  }// </editor-fold>//GEN-END:initComponents  //GEN-FIRST:event\_btnConectarseActionPerformed  private void btnConectarseActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String nickname = txtNickname.getText();  java.awt.EventQueue.invokeLater(() -> new Chat(nickname).setVisible(true));  this.dispose();  }//GEN-LAST:event\_btnConectarseActionPerformed  /\*\*  \* @param args the command line arguments  \*/  public static void main(String args[]) {  /\* Set the Nimbus look and feel \*/  //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">  /\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html  \*/  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(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  //</editor-fold>  /\* Create and display the form \*/  java.awt.EventQueue.invokeLater(() -> new Login().setVisible(true));  }  // Variables declaration - do not modify//GEN-BEGIN:variables  private javax.swing.JButton btnConectarse;  private javax.swing.JLabel jLabel1;  private javax.swing.JLabel labelNickname;  private javax.swing.JTextField txtNickname;  // End of variables declaration//GEN-END:variables  } |

## Lógica

La lógica fue dividida en una clase que envía y en otra que recibe, además lo que se envía a través de datagramas es una clase de Java que contiene el contenido del mensaje, por ultimo se tiene una interfaz con algunas constantes que se utilizan tanto en el cliente como en el servidor.

MulticastUtilidades.java

|  |
| --- |
| package logica;  public interface MulticastUtilidades {  String DIRECCION = "235.1.1.1";  int PUERTO = 4445;  int TAM\_BUFFER = 8000;  } |

Mensaje.java

|  |
| --- |
| package logica;  import java.io.Serializable;  import java.net.URL;  import java.util.ArrayList;  public class Mensaje implements Serializable {  public static final int ICONO = 0;  public static final int IMAGEN = 1;  public static final int ANUNCIO = 2;  public static final int LISTA\_CONECTADOS = 3;  private String destinatario;  private String usuario;  private String clave;  private String mensaje;  private String imagen;  private int tipoMensaje = 0;  private ArrayList<String> conectados;  private byte[] datos;  private int enviados;  private long imgTam;  // Esto se deberia de cambiar por la direccion del localhost  public static final URL ANGER = ClassLoader.getSystemResource("resources/anger.png");  public static final URL HEART = ClassLoader.getSystemResource("resources/heart.png");  public static final URL POOP = ClassLoader.getSystemResource("resources/poop.png");  public static final URL SADNESS = ClassLoader.getSystemResource("resources/sadness.png");  public static final URL SMILE = ClassLoader.getSystemResource("resources/smile.png");  public String getUsuario() {  return usuario;  }  public void setUsuario(String usuario) {  this.usuario = usuario;  }  public String getMensaje() {  return mensaje;  }  public void setMensaje(String mensaje) {  this.mensaje = mensaje;  }  public String getImagen() {  return imagen;  }  public void setImagen(String imagen) {  this.imagen = imagen;  }  public int getTipoMensaje() {  return tipoMensaje;  }  public void setTipoMensaje(int tipoMensaje) {  this.tipoMensaje = tipoMensaje;  }  public String getDestinatario() {  return destinatario;  }  public void setDestinatario(String destinatario) {  this.destinatario = destinatario;  }  public ArrayList<String> getConectados() {  return conectados;  }  public void setConectados(ArrayList<String> conectados) {  this.conectados = conectados;  }  public byte[] getDatos() {  return datos;  }  public void setDatos(byte[] datos) {  this.datos = datos;  }  public int getEnviados() {  return enviados;  }  public void setEnviados(int enviados) {  this.enviados = enviados;  }  public long getImgTam() {  return imgTam;  }  public void setImgTam(long imgTam) {  this.imgTam = imgTam;  }  public String getClave() {  return clave;  }  public void setClave(String clave) {  this.clave = clave;  }  @Override  public String toString() {  return "Mensaje{" +  "destinatario='" + destinatario + '\'' +  ", usuario='" + usuario + '\'' +  ", clave='" + clave + '\'' +  "\nmensaje='" + mensaje + '\'' +  ", imagen='" + imagen + '\'' +  ", tipoMensaje=" + tipoMensaje +  ", conectados=" + conectados +  ", enviados=" + enviados +  ", imgTam=" + imgTam +  '}';  }  public String construirMensaje() {  String msj;  if (this.tipoMensaje == ANUNCIO) {  return this.mensaje;  } else {  if (destinatario == null) msj = "<div><b>" + this.usuario + ":</b><span>";  else msj = "<div><b>" + this.usuario + "</b> para <b>" + this.destinatario + ":</b><span>";  String temporal = mensaje;  temporal = temporal.replace(">=|", obtenerEtiquetaImagen(ANGER.toString(), ICONO));  temporal = temporal.replace("<3", obtenerEtiquetaImagen(HEART.toString(), ICONO));  temporal = temporal.replace(":poop:", obtenerEtiquetaImagen(POOP.toString(), ICONO));  temporal = temporal.replace("='(", obtenerEtiquetaImagen(SADNESS.toString(), ICONO));  temporal = temporal.replace("=)", obtenerEtiquetaImagen(SMILE.toString(), ICONO));  if (this.tipoMensaje == IMAGEN)  msj += temporal + "</span><p>" + obtenerEtiquetaImagen(this.imagen, IMAGEN) + "</p></div>";  else  msj += temporal + "</span></div>";  }  return msj;  }  private CharSequence obtenerEtiquetaImagen(String img, int tipo) {  if (tipo == ICONO)  return "<img src=\"" + img + "\" width=\"20\" height=\"20\"/>";  else  return "<img src=\"file:" + img + "\" width=\"250\" height=\"250\"/>";  }  } |

Recibir.java

|  |
| --- |
| package logica;  import interfaz.Chat;  import java.io.\*;  import javax.swing.text.BadLocationException;  import java.net.DatagramPacket;  import java.net.MulticastSocket;  import java.security.NoSuchAlgorithmException;  /\*\*  \* @author tona created on 14/09/2017 for Practica4.  \*/  public class Recibir implements Runnable, MulticastUtilidades {  private MulticastSocket m;  private String carpeta;  public Recibir(MulticastSocket multicastSocket, String usuario) {  this.m = multicastSocket;  this.carpeta = usuario;  }  @Override  public void run() {  DatagramPacket p = new DatagramPacket(new byte[TAM\_BUFFER], TAM\_BUFFER);  while (true) {  try {  System.out.println("Recibiendo informacion...");  m.receive(p);  Mensaje mensaje = recuperarMensaje(p);  if (mensaje.getTipoMensaje() == Mensaje.ANUNCIO) {  Chat.agregarMensaje(mensaje);  Chat.agregarUsuarioLista(mensaje.getUsuario());  } else if (mensaje.getTipoMensaje() == Mensaje.LISTA\_CONECTADOS) {  System.out.println("Recibiendo lista de conectados");  Chat.cargarListaConectados(mensaje.getConectados());  } else if (mensaje.getTipoMensaje() == Mensaje.IMAGEN) {  String archivo = this.carpeta + "\_" + mensaje.getImagen();  FileOutputStream fos = new FileOutputStream(archivo, true);  fos.write(mensaje.getDatos(), 0, mensaje.getEnviados());  fos.close();  File f = new File(archivo);  //System.out.println(f.length() + " " + mensaje.getImgTam());  String clave = Enviar.obtenerClaveHash(f);  if (clave.equals(mensaje.getClave())) {  System.out.println("Mostar imagen");  mensaje.setImagen(f.getAbsolutePath());  Chat.agregarMensaje(mensaje);  }  }else  Chat.agregarMensaje(mensaje);  } catch (IOException | ClassNotFoundException | BadLocationException | NoSuchAlgorithmException e) {  e.printStackTrace();  }  p.setLength(TAM\_BUFFER);  }  }    private static Mensaje recuperarMensaje(DatagramPacket paquete) throws IOException, ClassNotFoundException {  // System.out.println("Datagrama recibido, extrayendo informacion...");  // System.out.printf("Host remoto: %s:%s\n", paquete.getAddress().getHostAddress(), paquete.getPort());  // System.out.println("Datos del paquete:");  ObjectInputStream ois = new ObjectInputStream(new ByteArrayInputStream(paquete.getData()));  Mensaje msj = (Mensaje) ois.readObject();  //System.out.println(msj.toString());  //System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  ois.close();  return msj;  }  } |

Enviar.java

|  |
| --- |
| package logica;  import java.io.\*;  import java.net.DatagramPacket;  import java.net.InetAddress;  import java.net.MulticastSocket;  import java.security.MessageDigest;  import java.security.NoSuchAlgorithmException;  import java.util.Random;  /\*\*  \* @author tona created on 14/09/2017 for Practica4.  \*/  public class Enviar implements MulticastUtilidades {  private MulticastSocket socket;  private InetAddress grupo;  private int puerto;  private Random random;  public Enviar(MulticastSocket multicastSocket, InetAddress grupo, int puerto) {  this.socket = multicastSocket;  this.grupo = grupo;  this.puerto = puerto;  this.random = new Random();  }    public void enviarMensaje(Mensaje msj) throws IOException {  ByteArrayOutputStream baos = new ByteArrayOutputStream(TAM\_BUFFER);  ObjectOutputStream oos = new ObjectOutputStream(new BufferedOutputStream(baos));  oos.flush();  oos.writeObject(msj);  oos.flush();  byte[] datos = baos.toByteArray();  DatagramPacket paquete = new DatagramPacket(datos, datos.length, grupo, puerto);  socket.send(paquete);  oos.close();  baos.close();  }  public void enviarAnuncio(String nickname) {  Mensaje anuncio = new Mensaje();  anuncio.setTipoMensaje(Mensaje.ANUNCIO);  String r = Integer.toHexString(random.nextInt(256));  String g = Integer.toHexString(random.nextInt(256));  String b = Integer.toHexString(random.nextInt(256));  String color = r + g + b;  System.out.println(color);  anuncio.setUsuario(nickname);  nickname = "<div style=\"color:#" + color + "\"><b>" + nickname + "</b> se ha conectado</div>";  anuncio.setMensaje(nickname);  try {  enviarMensaje(anuncio);  } catch (IOException e) {  e.printStackTrace();  }  }  public void enviarImagen(File imagen, Mensaje msj) throws IOException {  long tam = imagen.length();  long leidos = 0;  String clave = "";  try {  clave = obtenerClaveHash(imagen);  } catch (NoSuchAlgorithmException e) {  e.printStackTrace();  }  int n;  DataInputStream dis = new DataInputStream(new FileInputStream(imagen.getAbsolutePath()));  while(leidos < tam) {  Mensaje mensaje = new Mensaje();  mensaje.setTipoMensaje(Mensaje.IMAGEN);  mensaje.setMensaje(msj.getMensaje());  mensaje.setImagen(imagen.getName());  mensaje.setClave(clave);  mensaje.setImgTam(tam);  mensaje.setUsuario(msj.getUsuario());  byte[] b = new byte[7000];  n = dis.read(b);  mensaje.setDatos(b);  mensaje.setEnviados(n);  enviarMensaje(mensaje);  leidos += n;  System.out.println("Bytes " + leidos);  try {  Thread.sleep(5);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  dis.close();  }  static String obtenerClaveHash(File archivo) throws NoSuchAlgorithmException, IOException {  MessageDigest sha1 = MessageDigest.getInstance("SHA1");  FileInputStream fis = new FileInputStream(archivo.getAbsolutePath());  byte[] datos = new byte[1024];  int leidos;  try {  while ((leidos = fis.read(datos)) != -1) {  sha1.update(datos, 0, leidos);  }  } catch (IOException e) {  e.printStackTrace();  }  byte[] hashBytes = sha1.digest();  StringBuilder sb = new StringBuilder();  for (byte hashByte : hashBytes) {  sb.append(Integer.toString((hashByte & 0xff) + 0x100, 16).substring(1));  }  fis.close();  return sb.toString();  }  } |

# Servidor

El servidor solo tiene una parte lógica que se encarga de gestionar a los usuarios que se conectan y de recibir los mensajes de inicio de sesión, es por esto por lo que cuenta con una clase que funge como el mensaje que se envía.

MulticastConstantes.java

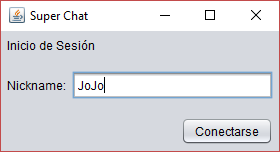
|  |
| --- |
| package logica;  public interface MulticastConstantes {  String DIRECCION = "235.1.1.1";  int PUERTO = 4445;  int TAM\_BUFFER = 6400;  } |

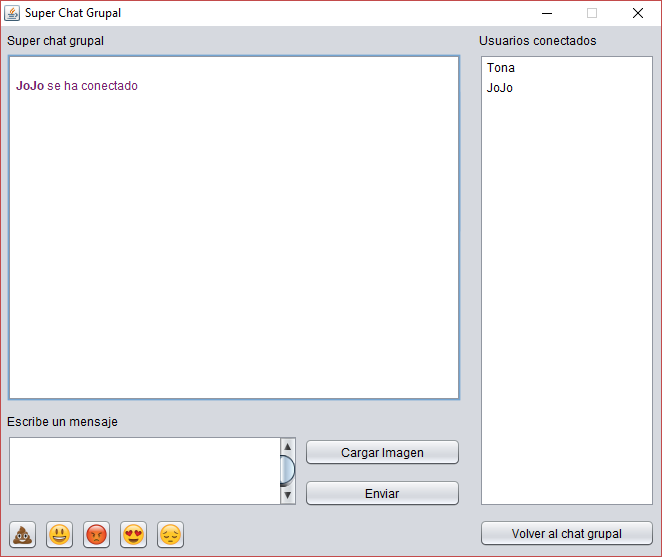
Servidor.java

|  |
| --- |
| package logica;  import java.io.BufferedOutputStream;  import java.io.ByteArrayInputStream;  import java.io.ByteArrayOutputStream;  import java.io.IOException;  import java.io.ObjectInputStream;  import java.io.ObjectOutputStream;  import java.net.DatagramPacket;  import java.net.InetAddress;  import java.net.MulticastSocket;  import java.util.ArrayList;  public class Servidor implements MulticastConstantes {  private static ArrayList<String> conectados = new ArrayList<>();  private static InetAddress grupo;  private static MulticastSocket multicastSocket;  public static void main(String[] args) throws ClassNotFoundException {  try {  grupo = InetAddress.getByName(DIRECCION);  multicastSocket = new MulticastSocket(PUERTO);  multicastSocket.joinGroup(grupo);  multicastSocket.setReuseAddress(true);  DatagramPacket paquete = new DatagramPacket(new byte[TAM\_BUFFER], TAM\_BUFFER);  System.out.println("Servidor iniciado");  while (true) {  System.out.println("Recibiendo informacion en el servidor...");  multicastSocket.receive(paquete);  Mensaje msj = recuperarMensaje(paquete);  System.out.println(msj.toString());  if (msj.getTipoMensaje() == Mensaje.ANUNCIO){  System.out.println("Anuncio recibido...");  conectados.add(msj.getUsuario());  Mensaje respuesta = new Mensaje();  respuesta.setConectados(conectados);  respuesta.setTipoMensaje(Mensaje.LISTA\_CONECTADOS);  enviarMensaje(respuesta);  System.out.println("Lista de conectados enviados");  }  paquete.setLength(TAM\_BUFFER);  }  } catch (IOException e) {  e.printStackTrace();  }  }    private static Mensaje recuperarMensaje(DatagramPacket paquete) throws IOException, ClassNotFoundException {  System.out.println("Datagrama recibido, extrayendo informacion...");  System.out.printf("Host remoto: %s:%s\n", paquete.getAddress().getHostAddress(), paquete.getPort());  System.out.println("Datos del paquete:");  ObjectInputStream ois = new ObjectInputStream(new ByteArrayInputStream(paquete.getData()));  Mensaje msj = (Mensaje) ois.readObject();  System.out.println(msj.toString());  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  ois.close();  return msj;  }  private static void enviarMensaje(Mensaje msj) throws IOException {  ByteArrayOutputStream baos = new ByteArrayOutputStream(TAM\_BUFFER);  ObjectOutputStream oos = new ObjectOutputStream(new BufferedOutputStream(baos));  oos.flush();  oos.writeObject(msj);  oos.flush();  byte[] datos = baos.toByteArray();  DatagramPacket paquete = new DatagramPacket(datos, datos.length, grupo, PUERTO);  multicastSocket.send(paquete);  System.out.println("Enviando");  oos.close();  baos.close();  }  } |

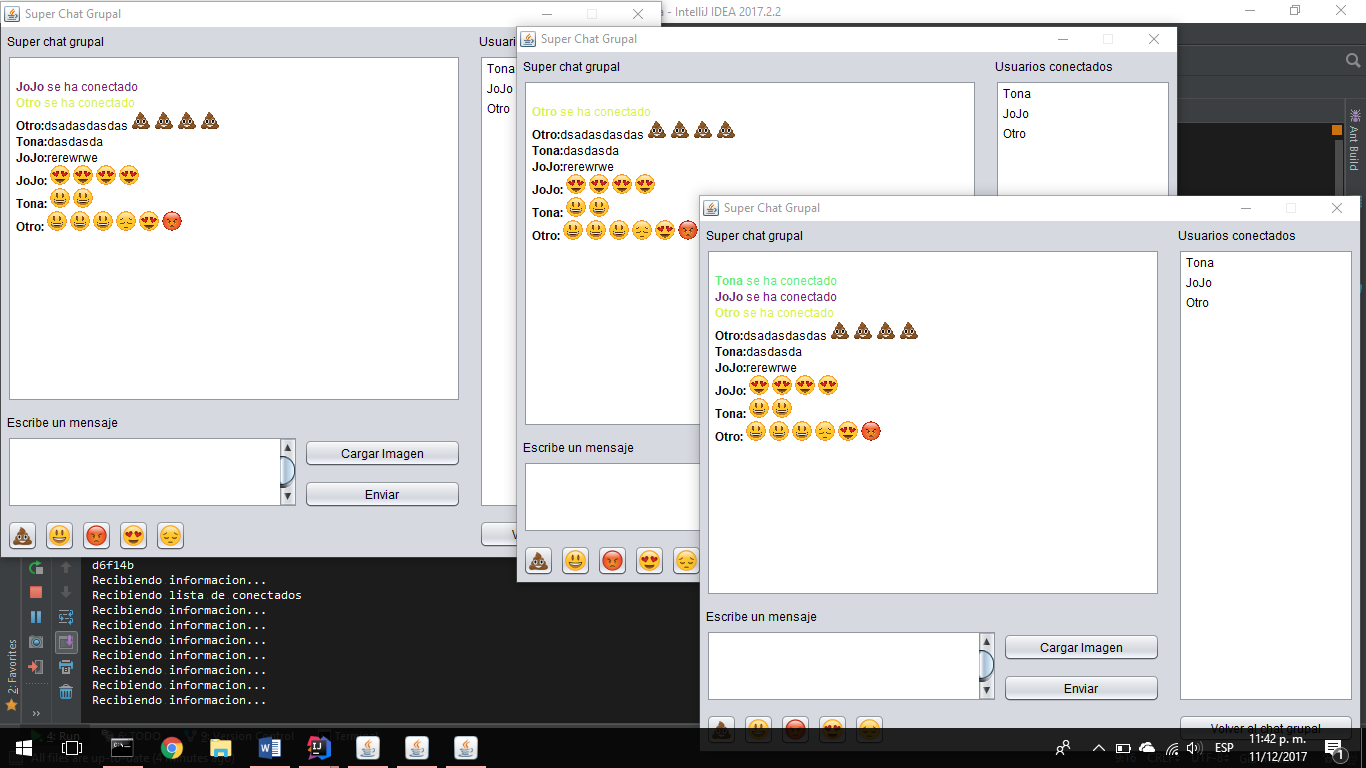
Pruebas

En la siguiente imagen se puede observar el inicio de sesión que se requiere para unirse al chat, solo se debe de ingresar un nombre de usuario y presionar conectarse.

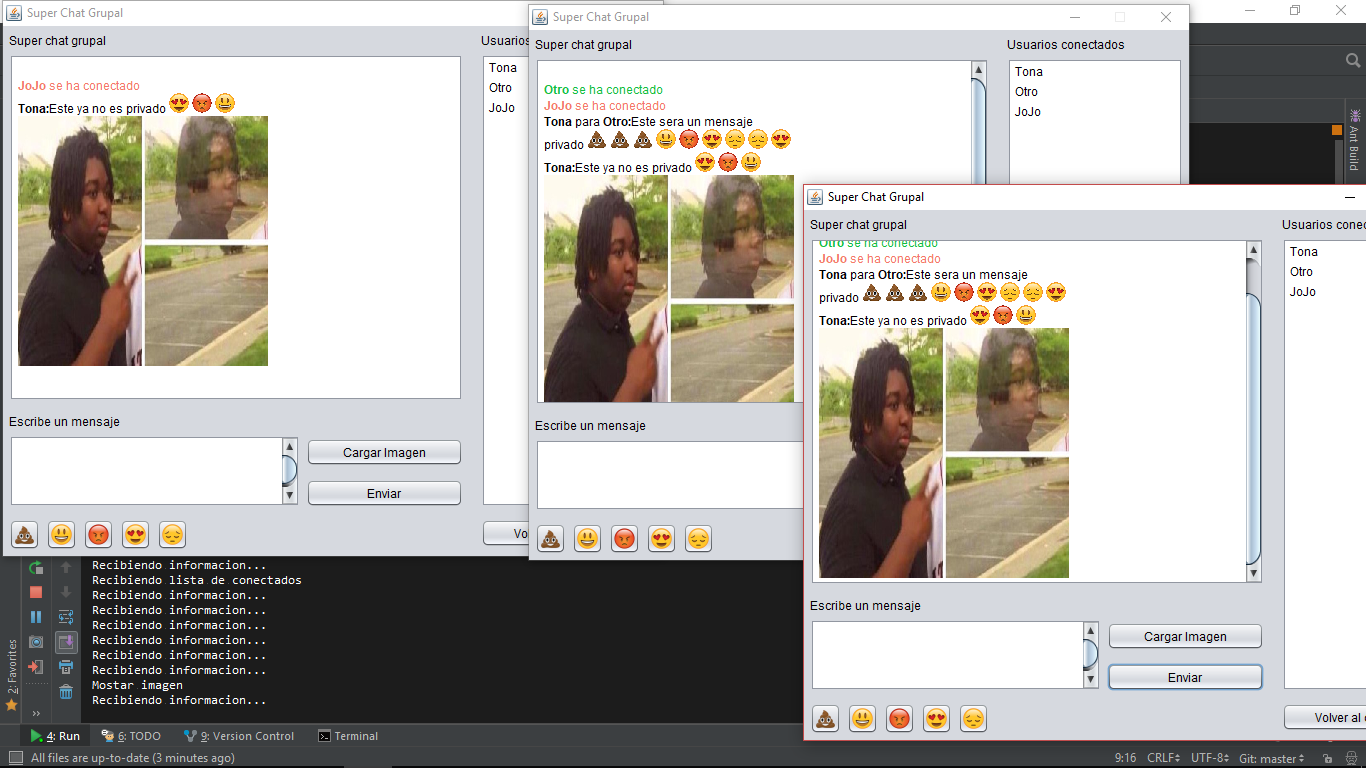




Esta será la siguiente pantalla que vera el usuario, una ventana con un textarea para escribir y agregar emojis además de contar con un panel para desplegar los mensajes recibidos junto a una lista con el nombre de los usuarios conectados, también están los botones de enviar y de cargar imagen, este ultimo permite seleccionar una imagen para después enviarla, el botón volver a chat grupal es usado para salir del modo mensaje privado que se activa al pulsar el nombre de algún usuario en la lista de la derecha.



Esta imagen muestra el envío de mensajes entre los usuarios, como se aprecia en la imagen dichos mensajes son públicos.



En esta ultima imagen se aprecia el envío de imágenes y el de mensajes privados entre usuarios.

Conclusiones

Al elaborar esta práctica fue evidente que los sockets de datagrama son más complejos que los sockets de flujo, pero al mismo tiempo la versatilidad que presentan los hace una buena opción para aplicaciones que deben de tratar con diversos agentes interactuando entre ellos. Sin embargo, es importante cuidar la persistencia de los datos ya que es bastante probable que en un inicio se produzcan errores al tratar de mandar tramas de datos ya que a diferencia de TCP no hay una forma de garantizar el envío de las tramas.

Referencias

* [1] K. Calvert and M. Donahoo, TCP/IP sockets in Java. San Francisco: Morgan Kaufmann Publishers, 2002.