ALUMNO:

DÍAZ MEDINA JESÚS KAIMORTS

ESQUIVEL VALDEZ ALBERTO

VARGAR ROMERO ERICK EFRÁIN

MATERÍA:

REDES COMPUTACIONALES

PROFESOR:

MORENO CERVANTES AXEL ERNESTO

GRUPO:

3CM9

PRACTICA 4

**Introducción**

Los sockets multicast son de suma importancia, puesto que nos permiten enviar un mensaje por red y que este sea recibido por otros ordenadores simultáneamente.

Para estos sockets se utilizan, como su nombre lo dice, direcciones multicast, es decir direcciones en el rango 224.0.0.0 a 239.255.255.255.

Una de las aplicaciones más usadas en el mundo son las aplicaciones de mensajería o chat.

Vemos que hoy en día todas esas aplicaciones tienen funciones como el envío de archivos, emoticonos o la creación de grupos de conversación.

En esta práctica se tratará de simular una de estas con Sockets Mulitcast.

**Desarrollo**

Se desarrolló un servicio de mensajería que permite al usuario ingresar a grupos en donde el mensaje o emoticono que mande es recibido por todos los que están unidos a dicho grupo. Para lograr ello se hizo uso de sockets multicast.

A continuación se encuentra el código de la aplicación.

1. import java.awt.Color;
2. import java.awt.Dimension;
3. import java.awt.Image;
4. import java.io.DataInputStream;
5. import java.io.DataOutputStream;
6. import java.io.File;
7. import java.io.FileInputStream;
8. import java.io.FileOutputStream;
9. import java.io.IOException;
10. import java.net.DatagramPacket;
11. import java.net.InetAddress;
12. import java.net.MulticastSocket;
13. import java.net.SocketException;
14. import javax.swing.Icon;
15. import javax.swing.ImageIcon;
16. import javax.swing.JLabel;
17. import javax.swing.JPanel;
18. import javax.swing.JFileChooser;
19. import javax.swing.filechooser.FileNameExtensionFilter;
20. public class InterfazChat extends javax.swing.JFrame {
21. private static int lineafinal=0;
22. private static boolean bandera=true;
23. public InterfazChat() {
24. initComponents();
25. this.setResizable(true);
26. String rutaImg="src/iconoImagen.png";
27. jToggleButton2.setOpaque(false);
28. jToggleButton2.setContentAreaFilled(false);
29. jToggleButton2.setBorderPainted(false);
30. ImageIcon imagen=new ImageIcon(rutaImg);
31. Icon iconitoIm=new ImageIcon(imagen.getImage().getScaledInstance(jToggleButton2.getWidth(), jToggleButton2.getHeight(),Image.SCALE\_DEFAULT));
32. jToggleButton2.setIcon(iconitoIm);
34. jToggleButton2.updateUI();
35. jToggleButton2.repaint();
36. }
38. @SuppressWarnings("unchecked")
39. // <editor-fold defaultstate="collapsed" desc="Generated Code">
40. private void initComponents() {
41. jToggleButton2 = new javax.swing.JToggleButton();
42. jScrollPane2 = new javax.swing.JScrollPane();
43. textEscritura = new javax.swing.JTextPane();
44. jToggleButton1 = new javax.swing.JToggleButton();
45. jScrollPane1 = new javax.swing.JScrollPane();
46. panelChat = new javax.swing.JPanel();
47. jLabel1 = new javax.swing.JLabel();
48. jLabel3 = new javax.swing.JLabel();
49. setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);
50. setTitle("Chat");
51. setBackground(new java.awt.Color(204, 51, 0));
52. setCursor(new java.awt.Cursor(java.awt.Cursor.HAND\_CURSOR));
53. setForeground(new java.awt.Color(153, 153, 255));
54. jToggleButton2.setBackground(new java.awt.Color(255, 0, 153));
55. jToggleButton2.setForeground(new java.awt.Color(255, 0, 102));
56. jToggleButton2.setText("SUBIR");
57. jToggleButton2.addActionListener(new java.awt.event.ActionListener() {
58. public void actionPerformed(java.awt.event.ActionEvent evt) {
59. jToggleButton2ActionPerformed(evt);
60. }
61. });
62. textEscritura.setBorder(null);
63. textEscritura.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
64. textEscritura.setSelectionColor(new java.awt.Color(255, 0, 102));
65. jScrollPane2.setViewportView(textEscritura);
66. jToggleButton1.setBackground(new java.awt.Color(255, 51, 102));
67. jToggleButton1.setForeground(new java.awt.Color(255, 0, 153));
68. jToggleButton1.setText("Enviar");
69. jToggleButton1.addActionListener(new java.awt.event.ActionListener() {
70. public void actionPerformed(java.awt.event.ActionEvent evt) {
71. jToggleButton1ActionPerformed(evt);
72. }
73. });
74. panelChat.setBackground(new java.awt.Color(255, 204, 204));
75. jLabel1.setBackground(new java.awt.Color(255, 255, 255));
76. jLabel1.setFont(new java.awt.Font("Sitka Text", 1, 70)); // NOI18N
77. jLabel1.setForeground(new java.awt.Color(255, 255, 255));
78. jLabel1.setText("LoveChat");
79. jLabel1.setToolTipText("");
80. javax.swing.GroupLayout panelChatLayout = new javax.swing.GroupLayout(panelChat);
81. panelChat.setLayout(panelChatLayout);
82. panelChatLayout.setHorizontalGroup(
83. panelChatLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
84. .addGroup(panelChatLayout.createSequentialGroup()
85. .addGap(93, 93, 93)
86. .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 360, javax.swing.GroupLayout.PREFERRED\_SIZE)
87. .addContainerGap(355, Short.MAX\_VALUE))
88. );
89. panelChatLayout.setVerticalGroup(
90. panelChatLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
91. .addGroup(panelChatLayout.createSequentialGroup()
92. .addGap(167, 167, 167)
93. .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 150, javax.swing.GroupLayout.PREFERRED\_SIZE)
94. .addContainerGap(929, Short.MAX\_VALUE))
95. );
96. jScrollPane1.setViewportView(panelChat);
97. jLabel3.setIcon(new javax.swing.ImageIcon(getClass().getResource("/fondo.jpg"))); // NOI18N
98. javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
99. getContentPane().setLayout(layout);
100. layout.setHorizontalGroup(
101. layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
102. .addGroup(layout.createSequentialGroup()
103. .addGap(300, 300, 300)
104. .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 510, javax.swing.GroupLayout.PREFERRED\_SIZE))
105. .addGroup(layout.createSequentialGroup()
106. .addGap(300, 300, 300)
107. .addComponent(jToggleButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 80, javax.swing.GroupLayout.PREFERRED\_SIZE)
108. .addGap(0, 0, 0)
109. .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 360, javax.swing.GroupLayout.PREFERRED\_SIZE)
110. .addGap(0, 0, 0)
111. .addComponent(jToggleButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 70, javax.swing.GroupLayout.PREFERRED\_SIZE))
112. .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 830, javax.swing.GroupLayout.PREFERRED\_SIZE)
113. );
114. layout.setVerticalGroup(
115. layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
116. .addGroup(layout.createSequentialGroup()
117. .addGap(10, 10, 10)
118. .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 510, javax.swing.GroupLayout.PREFERRED\_SIZE)
119. .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
120. .addComponent(jToggleButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)
121. .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)
122. .addComponent(jToggleButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)))
123. .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 580, javax.swing.GroupLayout.PREFERRED\_SIZE)
124. );
125. pack();
126. }// </editor-fold>
127. private void jToggleButton1ActionPerformed(java.awt.event.ActionEvent evt) {
128. bandera=false;
129. try {
130. String dmc="228.1.1.1";
131. InetAddress grupo = InetAddress.getByName(dmc);
132. MulticastSocket socket = new MulticastSocket();
133. // Se une al grupo
134. //socket.joinGroup(grupo);
135. // Envia el mensaje
136. byte[] m = textEscritura.getText().getBytes();
137. String aux=textEscritura.getText();
138. textEscritura.setText("");
140. String i=Integer.toString(0);
141. byte [] opcc=i.getBytes();
142. DatagramPacket op=new DatagramPacket(opcc,opcc.length,grupo,1234);
143. socket.send(op);
145. DatagramPacket mensajeSalida =new DatagramPacket(m, m.length, grupo, 1234);
146. socket.send(mensajeSalida);


150. Dimension size=panelChat.getSize();
152. //a.setText("Agregar al carrito");
153. JLabel a=new JLabel();
155. //aqui
156. int t=aux.length();
157. a.setSize(20\*t, 30);
158. int x=(size.width-400);
159. int y=lineafinal;
160. a.setLocation(x,y);
161. lineafinal+=20;
162. a.setOpaque(true);
163. a.setBorder(null);
164. a.setBackground(Color.pink);
165. if(aux.trim().equals(":)")){
166. ImageIcon imagen=new ImageIcon("imagenes\\feliz.jpg");
167. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
168. a.setIcon(icono);
169. }
170. else if(aux.trim().equals(":(")){
171. ImageIcon imagen=new ImageIcon("imagenes\\triste.jpg");
172. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
173. a.setIcon(icono);
174. }
175. else if(aux.trim().equals(":\*")){
176. ImageIcon imagen=new ImageIcon("imagenes\\beso.jpg");
177. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
178. a.setIcon(icono);
179. }
181. else{
182. a.setText(aux);
183. }
185. a.updateUI();
186. panelChat.add(a);
187. panelChat.updateUI();
188. // Se queda a la espera de mensajes al grupo, hasta recibir "Adios"
190. // Si recibe "Adios" abandona el grupo
191. // socket.leaveGroup(grupo);
192. } catch (SocketException e) {
193. System.out.println("Socket:" + e.getMessage());
194. } catch (IOException e) {
195. System.out.println("IO:" + e.getMessage());
196. }
197. // TODO add your handling code here:
198. }
199. private void jToggleButton2ActionPerformed(java.awt.event.ActionEvent evt) {
200. bandera=false;
201. try {
202. String dmc="228.1.1.1";
203. InetAddress grupo = InetAddress.getByName(dmc);
204. MulticastSocket socket = new MulticastSocket();
205. String i=Integer.toString(1);
206. byte [] opcc=i.getBytes();
207. DatagramPacket op=new DatagramPacket(opcc,opcc.length,grupo,1234);
208. socket.send(op);
209. // filechooser
211. JFileChooser jf=new JFileChooser();
212. FileNameExtensionFilter filtroImagen=new FileNameExtensionFilter("JPG, PNG & GIF","jpg","png","gif");
213. jf.setFileFilter(filtroImagen);
214. jf.requestFocus();
215. int r=jf.showOpenDialog(null);
216. if(r==JFileChooser.APPROVE\_OPTION){
217. File f=jf.getSelectedFile();
218. String nombreA=f.getName();
219. int porcentaje;
220. long tam=f.length(),enviados=0;
221. String path=f.getAbsolutePath();
222. DataInputStream dis=new DataInputStream(new FileInputStream(path));
223. opcc=nombreA.getBytes();
224. System.out.println(nombreA);
225. op=new DatagramPacket(opcc,opcc.length,grupo,1234);
226. socket.send(op);
227. opcc=Long.toString(tam).getBytes();
228. op=new DatagramPacket(opcc,opcc.length,grupo,1234);
229. socket.send(op);
230. byte[] b = new byte[1500];
231. while(enviados<tam){
232. int n=dis.read(b);
233. enviados+=n;
234. DatagramPacket mensajeSalida =new DatagramPacket(b, b.length, grupo, 1234);
235. socket.send(mensajeSalida);
236. porcentaje=(int)((enviados\*100)/tam);
237. System.out.println(porcentaje);
238. }
239. dis.close();
240. Dimension siz=panelChat.getSize();
241. JLabel a=new JLabel();
242. a.setSize(200, 200);
243. int x=siz.width-400;
244. int y=lineafinal+40;
245. lineafinal+=240;
246. a.setLocation(x,y);
248. ImageIcon imagen=new ImageIcon("imagenes\\"+nombreA);
249. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
250. a.setIcon(icono);
251. panelChat.add(a);
252. panelChat.updateUI();
254. }
255. // Envia el mensaje



260. // Se queda a la espera de mensajes al grupo, hasta recibir "Adios"
262. // Si recibe "visto" abandona el grupo
263. // socket.leaveGroup(grupo);
264. } catch (SocketException e) {
265. System.out.println("Socket:" + e.getMessage());
266. } catch (IOException e) {
267. System.out.println("IO:" + e.getMessage());
268. }
270. // TODO add your handling code here:
271. }
272. /\*\*
273. \* @param args the command line arguments
274. \*/
275. public static void main(String args[]) {
277. try {
278. for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
279. if ("Nimbus".equals(info.getName())) {
280. javax.swing.UIManager.setLookAndFeel(info.getClassName());
281. break;
282. }
283. }
284. } catch (ClassNotFoundException ex) {
285. java.util.logging.Logger.getLogger(InterfazChat.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
286. } catch (InstantiationException ex) {
287. java.util.logging.Logger.getLogger(InterfazChat.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
288. } catch (IllegalAccessException ex) {
289. java.util.logging.Logger.getLogger(InterfazChat.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
290. } catch (javax.swing.UnsupportedLookAndFeelException ex) {
291. java.util.logging.Logger.getLogger(InterfazChat.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
292. }
294. java.awt.EventQueue.invokeLater(new Runnable() {
295. public void run() {
296. new InterfazChat().setVisible(true);
297. }
298. });
299. //metodo recibe mensajes
300. try {
302. String dmc="228.1.1.1";
303. InetAddress grupo = InetAddress.getByName(dmc);
304. MulticastSocket socket = new MulticastSocket(1234);
305. socket.joinGroup(grupo);
307. byte[] bufer = new byte[150000];
308. String linea;
309. // Se queda a la espera de mensajes al grupo, hasta recibir "visto"
310. while (true) {
311. DatagramPacket mensajeEntrada= new DatagramPacket(bufer,bufer.length);
312. socket.receive(mensajeEntrada);
313. linea = new String(mensajeEntrada.getData(), 0, mensajeEntrada.getLength());
314. Integer op=new Integer(linea);
315. if(op==1){
316. int recibidos=0;
317. DatagramPacket mensaje =new DatagramPacket(bufer, bufer.length);
318. socket.receive(mensaje);
319. linea = new String(mensaje.getData(), 0, mensaje.getLength());
320. socket.receive(mensaje);
321. String nombr=linea;
322. linea = new String(mensaje.getData(), 0, mensaje.getLength());
323. long tam;
324. tam=Long.parseLong(linea);
326. DataOutputStream dos=new DataOutputStream(new FileOutputStream("imagenes\\"+nombr));
327. System.out.println("preparado para recibir archivo "+nombr+" desde "+socket.getInetAddress()+" con "+ tam );
328. while(recibidos<tam){
329. byte [] b=new byte[1500];
330. socket.receive(mensaje);
331. b=mensaje.getData();
332. int n=mensaje.getLength();
333. recibidos+=n;
334. dos.write(b,0,n);
335. dos.flush();
336. int porcentaje=(int)((recibidos)/tam);
337. System.out.println("p:"+porcentaje);
338. }
339. dos.close();
340. if(bandera){
341. Dimension siz=panelChat.getSize();
342. JLabel a=new JLabel();
344. a.setSize(200, 200);
345. int x=0;
346. int y=lineafinal+40;
347. lineafinal+=240;
348. a.setLocation(x,y);
350. ImageIcon imagen=new ImageIcon("imagenes\\"+nombr);
351. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
352. a.setIcon(icono);
353. panelChat.add(a);
354. panelChat.updateUI();
355. //label.sset
356. }
357. bandera=true;
358. }else{
360. DatagramPacket mensaje =new DatagramPacket(bufer, bufer.length);
361. socket.receive(mensaje);
362. linea = new String(mensaje.getData(), 0, mensaje.getLength());
364. //
365. if(bandera){
366. Dimension siz=panelChat.getSize();

369. JLabel a=new JLabel();
370. int t=linea.length();
371. a.setSize(10\*t, 20);
372. int x=0;
373. int y=lineafinal;
374. lineafinal+=20;
375. a.setLocation(x,y);
376. a.setOpaque(true);
377. a.setBorder(null);
378. a.setBackground(Color.blue);
379. if(linea.trim().equals(":)")){
380. ImageIcon imagen=new ImageIcon("imagenes\\feliz.jpg");
381. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
382. a.setIcon(icono);
383. }
384. else if(linea.trim().equals(":(")){
385. ImageIcon imagen=new ImageIcon("imagenes\\triste.jpg");
386. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
387. a.setIcon(icono);
388. }
389. else if(linea.trim().equals(":\*")){
390. ImageIcon imagen=new ImageIcon("imagenes\\beso.jpg");
391. Icon icono=new ImageIcon(imagen.getImage().getScaledInstance(a.getWidth(), a.getHeight(),Image.SCALE\_DEFAULT));
392. a.setIcon(icono);
393. }
395. else{
396. a.setText(linea);
397. }
399. //titlePanel.updateUI();
400. panelChat.add(a);
401. panelChat.updateUI();

404. }
405. bandera=true;
406. //
407. System.out.println("Recibido:" + linea);
408. if (linea.equals("Visto")) break;
409. }
410. }
411. // Si recibe "Adios" abandona el grupo
412. socket.leaveGroup(grupo);
413. } catch (SocketException e) {
414. System.out.println("Socket:" + e.getMessage());
415. } catch (IOException e) {
416. System.out.println("IO:" + e.getMessage());
417. }






425. //fin




431. }
432. // Variables declaration - do not modify
433. private javax.swing.JLabel jLabel1;
434. private javax.swing.JLabel jLabel3;
435. private javax.swing.JScrollPane jScrollPane1;
436. private javax.swing.JScrollPane jScrollPane2;
437. private javax.swing.JToggleButton jToggleButton1;
438. private javax.swing.JToggleButton jToggleButton2;
439. private static javax.swing.JPanel panelChat;
440. private javax.swing.JTextPane textEscritura;
441. // End of variables declaration
442. }

La aplicación nos muestra la siguiente pantalla en todos los chats.

|  |
| --- |
|  |

Nos permite enviar desde texto, caritas (emogis) hasta algún archivo.

|  |
| --- |
|  |

Como aquí podemos observar nos permite seleccionar algún archivo que deseamos mandar.

|  |
| --- |
|  |

Aquí recibe el archivo. En este caso fue una imagen.

|  |
| --- |
|  |

Conclusiones:

En esta práctica se hizo uso de Sockets Multicast, que es una herramienta que nos permite mandar a distintos cliente la misma información. La práctica fue dificl puesto que el enviar las imágenes o los emoticonos tenía que ser de manera personalizada. De igual forma los emoticonos se tuvieron que mandar como imágenes para no presentar otras complicaciones extra. La práctica nos ayudo a comprender un poco más como funcionan las aplicaciones de mensajería en grupo y sin duda fue muy útil puesto que dichas herramientas son muy utilizadas en la actualidad.