



**CSE 4226 — Network Programming Lab**

**Assignment 1**

**Developing a multi-threaded server-client  
application using TCP Sockets**

*Submitted by*

**Samin Shahriar Tokey**

**14.02.04.066**

**Section: B1**

Submission Date: 6th June 2018

## 1 Implementation summary

Features	Status
User Registration and Login:	Implemented
Online User Lists:	Implemented
Friend Request:	Implemented
Unicast:	Implemented
Multicast:	Implemented
Broadcast:	Implemented
Friend list:	Implemented
Create Chat Room:	Partially Implemented

## 2 Implementation Challenges

I ran into various errors while trying to implement this problem. Most of them were null pointer exceptions and server connection resets. However I managed to get past most of them by following forums on Geeks for Geeks[1] and Stackoverflow[2]

## 3 Limitation and Future Scope of Improvements

The whole system is command based, thus very limited. In future I'd like to develop a multiclient chat server with UI and more features.

## 4 Discussion

I learned quite a lot about networking with multithreaded client class.

## References

[1] Geek For Geeks,

<https://www.geeksforgeeks.org/>, 6-6-2018

[2] Stack Overflow,

<https://stackoverflow.com/>, 6-6-2018

# Appendices

## A CODE

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package chat.server;
7
8  import java.io.DataInputStream;
9  import java.io.DataOutputStream;
10 import java.io.IOException;
11 import java.net.Socket;
12 import java.util.Scanner;
13 import java.util.StringTokenizer;
14
15 /**
16  *
17  * @author
18  */
19 // ClientHandler class
20 class ClientHandler implements Runnable {
21
22     String rqst = null;
23     Scanner sc = new Scanner(System.in);
24     private String username;
25     private String password;
26     final DataInputStream dIP;
27     final DataOutputStream dOP;
28     Socket soc;
```

```

29     boolean isLoggedIn;
30     String friends[] = new String[10];
31     Room rooms[] = new Room[10];
32
33     // constructor
34     public ClientHandler(Socket soc, String name,
35         DataInputStream dIP, DataOutputStream dOP) {
36         this.dIP = dIP;
37         this.dOP = dOP;
38         this.username = name;
39         this.soc = soc;
40         this.isLoggedIn = false;
41     }
42     boolean fuse = true;
43
44     @Override
45     public void run() {
46
47         String received;
48         while (true) {
49             try {
50
51                 if (fuse) {
52                     dOP.writeUTF("Sign Up Stranger! Your Name?");
53                     username = dIP.readUTF();
54                     dOP.writeUTF("Your Password?");
55                     password = dIP.readUTF();
56                     dOP.writeUTF("Greetings! To Send Messages, you need to sign in
57 :");
58                     fuse = false;
59                 }
60                 received = dIP.readUTF();
61                 System.out.println(received);
62                 if (received.equals("signout")) {
63                     this.isLoggedIn = false;
64                 } else if (received.equals("makeRoom")) {
65                     for (int i = 0; i < rooms.length; i++) {

```

```

65         rooms[i] = new Room();
66     }
67     for (int i = 0; i < rooms.length; i++) {
68         if (rooms[i].roomName.equals("Blank")) {
69             dOP.writeUTF("Room name?");
70             String nm = dIP.readUTF();
71             rooms[i].roomName = nm;
72             rooms[i].participants.add(this.username);
73
74             dOP.writeUTF("Room Created!");
75             break;
76         }
77     }
78     } else if (received.equals("joinRoom")) {
79         dOP.writeUTF("Room name?");
80         String nm = dIP.readUTF();
81         boolean f = true;
82         for (int i = 0; i < rooms.length; i++) {
83             if (rooms[i].roomName.equals(nm)) {
84                 boolean f2 = true;
85                 f = false;
86                 for (String prt : rooms[i].participants) {
87                     if (prt.equals(this.username)) {
88                         dOP.writeUTF("You're already in the room :");
89                         f2 = false;
90                         break;
91                     }
92                 }
93             }
94             if (f2) {
95                 rooms[i].participants.add(nm);
96                 dOP.writeUTF("You have been added in the room");
97                 break;
98             }
99         }
100     }
101 }

```

```

102         if (f) {
103             dOP.writeUTF("ROOM NOT FOUND!");
104         }
105     } else if (received.equals("messageRoom")) {
106         dOP.writeUTF("Room name?");
107         String nm = dIP.readUTF();
108         dOP.writeUTF("Your message?");
109         String msg = dIP.readUTF();
110         boolean f1 = true;
111         for (int i = 0; i < rooms.length; i++) {
112             if (rooms[i].roomName.equals(nm)) {
113                 boolean f2 = true;
114                 f1 = false;
115                 for (String prt : rooms[i].participants) {
116                     if (prt.equals(this.username)) {
117                         for (String pr : rooms[i].participants) {
118                             if (pr.equals(this.username) == false) {
119                                 for (ClientHandler CCH : Server.
serverList) {
120                                     if (CCH.username.equals(pr) && CCH.
isloggedin) {
121                                         CCH.dOP.writeUTF("From Room : "
+ nm + ": " + msg);
122                                     }
123                                 }
124                             }
125                         }
126                         f2 = false;
127                         break;
128                     }
129                 }
130             }
131             if (f2) {
132                 dOP.writeUTF("YOU ARE NOT IN THE ROOM");
133             }
134         }
135     }

```

```

136         }
137
138         if (f1) {
139             dOP.writeUTF("ROOM NOT FOUND!");
140         }
141     } else if (received.equals("accept")) {
142         if (rqst != null) {
143             int i;
144             for (i = 0; i < friends.length; i++) {
145                 if (friends[i] == null) {
146                     friends[i] = rqst;
147                     rqst = null;
148                     this.dOP.writeUTF("Friend Request Accepted!");
149                     break;
150                 }
151             }
152         }
153     }
154
155     } else if (received.equals("frndlst")) {
156
157         int i;
158         for (i = 0; i < friends.length; i++) {
159             if (friends[i] != null) {
160
161                 this.dOP.writeUTF(friends[i]);
162
163             }
164         }
165     } else if (received.equals("denied")) {
166         for (ClientHandler CCH : Server.serverList) {
167             if (CCH.username.equals(rqst)) {
168                 CCH.dOP.writeUTF("Friend Request Denied :( ( Better
169 luck next time.");
170
171                 for (int i = 0; i <= CCH.friends.length; i++) {
172                     if (CCH.friends[i] == rqst) {
173                         CCH.friends[i] = null;

```

```

172         break;
173     }
174 }
175     rqst = null;
176     break;
177 }
178 }
179 } else if (received.equals("rqst")) {
180
181     int i;
182
183     for (i = 0; i <= friends.length; i++) {
184         if (friends[i] == null) {
185             friends[i] = dIP.readUTF();
186             break;
187         }
188     }
189
190     for (ClientHandler CCH : Server.serverList) {
191         if (CCH.username.equals(friends[i])) {
192             CCH.dOP.writeUTF("Friend Request from : " + this.
193 username + " Accept?");
194             CCH.rqst = this.username;
195             break;
196         }
197     }
198     dOP.writeUTF("REQUEST SENT!");
199 } else if (received.equals("signin")) {
200     String uname = dIP.readUTF();
201     String pass = dIP.readUTF();
202     boolean f = false;
203     if (this.username.equals(uname) == true && this.password.equals
204 (pass) == true) {
205         this.isloggedin = true;
206         f = true;
207         dOP.writeUTF(this.username + " Welcome!! You have
208 successfully signed in :)");

```



```

206
207         }
208         if (!f) {
209             dOP.writeUTF("LOGIN FAILED :@ !");
210         }
211
212     } else if (received.equals("online")) {
213
214         this.dOP.writeUTF("List of online people are: " + "\n");
215         for (ClientHandler CCH : Server.serverList) {
216             if (CCH.username.equals(this.username) == false && CCH.
217 isloggedin == true) {
218                 this.dOP.writeUTF(CCH.username + "\n");
219             }
220         }
221     } else {
222
223         StringTokenizer st = new StringTokenizer(received, ":");
224         if (st.countTokens() == 2) {
225             String first = st.nextToken();
226             String second = st.nextToken();
227
228             if (second.equals("all")) {
229                 for (ClientHandler CCH : Server.serverList) {
230                     if (CCH.username.equals(this.username) == false &&
231 CCH.isloggedin == true) {
232                         for (int i = 0; i <= CCH.friends.length; i++) {
233                             if (CCH.friends[i].equals(this.username)) {
234                                 CCH.dOP.writeUTF(this.username + " : "
235 + first);
236                             }
237                             break;
238                         }
239                     }
240                 }
241             }
242         }
243     }

```

```

240         } else {
241             for (ClientHandler CCH : Server.serverList) {
242                 if (CCH.username.equals(second) && CCH.isloggedin
= true) {
243                     for (int i = 0; i <= CCH.friends.length; i++) {
244                         if (CCH.friends[i].equals(this.username)) {
245                             CCH.dOP.writeUTF(this.username + " : "
+ first);
246                         }
247                         break;
248                     }
249                     break;
250                 }
251             }
252         }
253     } else {
254         String first = st.nextToken();
255         while (st.hasMoreTokens()) {
256             String second = st.nextToken();
257             for (ClientHandler CCH : Server.serverList) {
258                 if (CCH.username.equals(second) && CCH.isloggedin
= true) {
259                     for (int i = 0; i <= CCH.friends.length; i++) {
260                         if (CCH.friends[i].equals(this.username)) {
261                             CCH.dOP.writeUTF(this.username + " : "
+ first);
262                         }
263                         break;
264                     }
265                     break;
266                 }
267             }
268         }
269     }
270 }
271 } catch (IOException e) {
272     e.printStackTrace();

```

```

273
274
275
276
277

```

```

    }
}
}
}

```

```

1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package chat.server;
7
8  /**
9   *
10   * @author
11   */
12  // Java implementation for multithreaded chat client
13  // Save file as Client.java
14
15  import java.io.*;
16  import java.net.*;
17  import java.util.Scanner;
18
19  public class Client
20  {
21
22
23      public static void main(String args[]) throws UnknownHostException, IOException
24      {
25          Scanner scn = new Scanner(System.in);
26          InetAddress ip = InetAddress.getByName("localhost");
27
28          Socket s = new Socket(ip, 7777);
29          DataInputStream dis = new DataInputStream(s.getInputStream());
30          DataOutputStream dos = new DataOutputStream(s.getOutputStream());
31

```

```

32     Thread sendMessage = new Thread(new Runnable()
33     {
34         @Override
35         public void run() {
36             while (true) {
37
38                 String msg = scn.nextLine();
39
40                 try {
41                     dos.writeUTF(msg);
42                 } catch (IOException e) {
43                     e.printStackTrace();
44                 }
45             }
46         }
47     });
48
49     Thread readMessage = new Thread(new Runnable()
50     {
51         @Override
52         public void run() {
53
54             while (true) {
55                 try {
56                     String msg = dis.readUTF();
57                     System.out.println(msg);
58                 } catch (IOException e) {
59
60                     e.printStackTrace();
61                 }
62             }
63         }
64     });
65
66     sendMessage.start();
67     readMessage.start();
68

```

```

69     }
70 }

```

```

1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package chat.server;
7
8  /**
9   *
10   * @author
11   */
12  // Java implementation of Server side
13  // It contains two classes : Server and ClientHandler
14  // Save file as Server.java
15
16  import java.io.*;
17  import java.util.*;
18  import java.net.*;
19
20  // Server class
21  public class Server
22  {
23
24      static Vector<ClientHandler> serverList = new Vector<>();
25
26      static int i = 0;
27
28      public static void main(String[] args) throws IOException
29      {
30          ServerSocket ss = new ServerSocket(7777);
31
32          Socket s;
33
34          while (true)

```

```

35     {
36         s = ss.accept();
37
38         System.out.println("New client request received : " + s);
39
40         DataInputStream dIP = new DataInputStream(s.getInputStream());
41         DataOutputStream dOP = new DataOutputStream(s.getOutputStream());
42
43         System.out.println("Creating a new handler for this client...");
44
45         ClientHandler CCH = new ClientHandler(s, "client " + i, dIP, dOP);
46
47         Thread t = new Thread(CCH);
48
49
50
51         serverList.add(CCH);
52
53         t.start();
54
55         i++;
56
57     }
58 }
59 }

```

```

1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package chat.server;
7
8  import java.util.Vector;
9
10
11 public class Room implements Runnable{

```

```
12     Vector<String> participants = new Vector<>();
13     public String roomName=new String("Blank");
14
15     @Override
16     public void run() {
17         throw new UnsupportedOperationException("Not supported yet."); //To change
18         body of generated methods, choose Tools | Templates.
19     }
20
21 }
```