SocketClinet

|  |
| --- |
| // By Linh Hoang, University of Texas at Dallas |
|  | // April, 2017 |
|  |  |
|  | import java.io.\*; |
|  | import java.util.Scanner; |
|  | import java.net.\*; |
|  |  |
|  | public class SocketClient |
|  | { |
|  | Socket socket = null; |
|  | PrintWriter out = null; |
|  | BufferedReader in = null; |
|  | Scanner sc = new Scanner(System.in); |
|  | public void communicate() |
|  | { |
|  | String name= ""; |
|  | System.out.print("Enter your name : "); // Enter the name |
|  | while(true){ |
|  | name = sc.nextLine(); // Get User's name |
|  | out.println(name); // Send user name to Server to check |
|  | String valid = readServer(); // Receive validity string from Server |
|  |  |
|  | // Check Validity of the User's name |
|  | if (valid.equals("valid")){ |
|  | break; |
|  | }else{ |
|  | System.out.print("Existing user name, enter a new name: "); // Existing user, enter again |
|  | } |
|  | } |
|  |  |
|  | String choice =""; |
|  | outerloop: |
|  | while(true){ |
|  | displayChoice(); // Display menu |
|  | innerloop: |
|  | while(true){ |
|  | choice = sc.nextLine(); // Take the user's choice |
|  | switch (choice){ |
|  | case "1": |
|  | System.out.println("Known Users:"); |
|  | sendServer(choice); |
|  | displayInfo(choice); |
|  | break innerloop; |
|  | case "2": |
|  | System.out.println("Connected Users:"); |
|  | sendServer(choice); |
|  | displayInfo(choice); |
|  | break innerloop; |
|  | case "3": |
|  | sendServer(choice); |
|  | System.out.println(readServer()); |
|  | break innerloop; |
|  | case "4": |
|  | sendServer(choice); |
|  | System.out.println(readServer()); |
|  | break innerloop; |
|  | case "5": |
|  | sendServer(choice); |
|  | System.out.println(readServer()); |
|  | break innerloop; |
|  | case "6": |
|  | System.out.println("Your Messages:"); |
|  | sendServer(choice); |
|  | displayInfo(choice); |
|  | break innerloop; |
|  | case "7": |
|  | out.println(choice); |
|  | break outerloop; |
|  | default: |
|  | System.out.print("Invalid choice (only 1~7), Enter Your Choice: "); |
|  | } |
|  | } |
|  | } |
|  | sc.close(); |
|  | } |
|  |  |
|  | // Read a single line from Server |
|  | public String readServer(){ |
|  | try { |
|  | String st = in.readLine(); |
|  | // System.out.println("Incomming data: "+st); |
|  | return (st); |
|  | } catch (NumberFormatException | IOException e) { |
|  | return null; |
|  | } |
|  | } |
|  |  |
|  | // Read multiple lines from Server and print out formatted string |
|  | public void displayInfo(String choice){ |
|  | String message; |
|  | int i = 0; |
|  | while((message = readServer())!= null){ |
|  | if (!message.equals("stop")){ |
|  | if (!choice.equals("6")) |
|  | System.out.println("\t"+(i+1)+". "+message); |
|  | else{ |
|  | String name = message; |
|  | String date = readServer(); |
|  | String inbox = readServer(); |
|  | System.out.println("\t"+(i+1)+". From "+name+", "+date+", "+inbox); |
|  | } |
|  | }else{ |
|  | break; |
|  | } |
|  | i++; |
|  | } |
|  | // String message; |
|  | // int i = 0; |
|  | // while((message = readServer())!= null){ |
|  | // if (!message.equals("stop")){ |
|  | // System.out.println("\t"+(i+1)+". "+message); |
|  | // }else{ |
|  | // break; |
|  | // } |
|  | // i++; |
|  | // } |
|  | } |
|  |  |
|  | // Send information to Server |
|  | public void sendServer(String choice){ |
|  | String name = ""; |
|  | String message = ""; |
|  | int n = Integer.parseInt(choice); |
|  |  |
|  | if ( n != 1 && n != 2 && n != 6){ |
|  | if (n==3){ |
|  | System.out.print("Enter recipient's name: "); |
|  | name = sc.nextLine()+"\n"; |
|  | } |
|  | System.out.print("Enter a message: "); |
|  | String st =""; |
|  | while(true){ |
|  | st = sc.nextLine(); |
|  | if (st.length() <= 80){ |
|  | break; |
|  | }else{ |
|  | System.out.print("Your message is too long, enter a message again: "); |
|  | } |
|  | } |
|  | message = choice+"\n"+name+st; |
|  | // System.out.println("Message sent: "+message); |
|  | out.println(message); |
|  |  |
|  | }else{ |
|  | out.println(choice); |
|  | } |
|  | } |
|  |  |
|  |  |
|  | public void displayChoice(){ |
|  | System.out.print("\n\n1. Display the names of all known users.\n" |
|  | + "2. Display the names of all currently connected users.\n" |
|  | + "3. Send a text message to a particular user.\n" |
|  | + "4. Send a text message to all currently connected users.\n" |
|  | + "5. Send a text message to all known users.\n" |
|  | + "6. Get my messages.\n" |
|  | + "7. Exit.\n" |
|  | + "Enter your choice: "); |
|  | } |
|  | public void listenSocket(String host, int port) |
|  | { |
|  | //Create socket connection |
|  | try |
|  | { |
|  | socket = new Socket(host, port); |
|  | out = new PrintWriter(socket.getOutputStream(), true); |
|  | in = new BufferedReader(new InputStreamReader(socket.getInputStream())); |
|  | } |
|  | catch (UnknownHostException e) |
|  | { |
|  | System.out.println("Unknown host"); |
|  | System.exit(1); |
|  | } |
|  | catch (IOException e) |
|  | { |
|  | System.out.println("No I/O"); |
|  | System.exit(1); |
|  | } |
|  | } |
|  |  |
|  | public static void main(String[] args) |
|  | { |
|  | if (args.length != 2) |
|  | { |
|  | System.out.println("Usage: client hostname port"); |
|  | System.exit(1); |
|  | } |
|  |  |
|  | SocketClient client = new SocketClient(); |
|  |  |
|  | String host = args[0]; |
|  | int port = Integer.valueOf(args[1]); |
|  | client.listenSocket(host, port); |
|  | client.communicate(); |
|  | } |
|  | } |