ClinetWorker

|  |
| --- |
| import java.io.\*; |
|  | import java.net.Socket; |
|  | import java.text.\*; |
|  | import java.util.\*; |
|  |  |
|  | public class ClientWorker implements Runnable { |
|  |  |
|  | private Socket client; |
|  | private static ArrayList<User> userList; |
|  | private User newUser = new User(); |
|  | private User coUser = new User(); |
|  | private String status = "known"; |
|  |  |
|  | BufferedReader in = null; |
|  | PrintWriter out = null; |
|  |  |
|  | ClientWorker(Socket client, ArrayList<User> userList) |
|  | { |
|  | this.client = client; |
|  | this.userList = userList; |
|  | } |
|  |  |
|  | public void run() { |
|  |  |
|  | String line; |
|  |  |
|  | try { |
|  |  |
|  | in = new BufferedReader(new InputStreamReader(client.getInputStream())); |
|  | out = new PrintWriter(client.getOutputStream(), true); |
|  |  |
|  | } catch (IOException e) { |
|  |  |
|  | System.out.println("in or out failed"); |
|  | System.exit(-1); |
|  |  |
|  | } |
|  |  |
|  | newUser = userConnectionCheck(in, out, newUser, status); |
|  |  |
|  | try { |
|  |  |
|  | while ((line = in.readLine()) != null) { |
|  |  |
|  | if (line.equals("1")) { |
|  |  |
|  | checkAllKnownUsers(line, null); |
|  |  |
|  | } else if (line.equals("2")) { |
|  |  |
|  | checkAllConnectedUsers(line, null); |
|  |  |
|  | } else if (line.equals("3")) { |
|  |  |
|  | String name = in.readLine(); |
|  | String message = in.readLine(); |
|  | String time = getTime(); |
|  |  |
|  | // check whether the person sending message to is a known user or not. |
|  | // If he's unknown, add him to the user list. |
|  | int index = checkAUserStatus(name); |
|  | coUser = updateUser(index, name); |
|  |  |
|  | String sendMessage = newUser.name + "\n" + time + "\n" + message; |
|  | boolean saved = storeMessage(coUser, sendMessage, out); |
|  |  |
|  | if (saved) { |
|  |  |
|  | out.println("Message posted to " + name + "\n"); |
|  |  |
|  | } else { |
|  |  |
|  | out.println("full"); |
|  |  |
|  | } |
|  |  |
|  | System.out.println(newUser.name + " posts a message for " + coUser.name + "."); |
|  |  |
|  | } else if (line.equals("4")) { |
|  |  |
|  | String message = in.readLine(); |
|  | checkAllConnectedUsers(line, message); |
|  |  |
|  | } else if (line.equals("5")) { |
|  |  |
|  | String time = getTime(); |
|  | String message = in.readLine(); |
|  | checkAllKnownUsers(line, newUser.name + "\n" + time + "\n" + message); |
|  |  |
|  | } else if (line.equals("6")) { |
|  |  |
|  | int i = 0; |
|  | while (i < newUser.messageList.size()) { |
|  |  |
|  | out.println(newUser.messageList.get(i)); |
|  | i++; |
|  |  |
|  | } |
|  |  |
|  | System.out.println(newUser.name + " gets messages."); |
|  |  |
|  | } else if (line.equals("7")) { |
|  |  |
|  | newUser.setConnected(false); |
|  | System.out.println(newUser.name + " exits."); |
|  | exitClient(); |
|  | break; |
|  |  |
|  | } |
|  |  |
|  | out.println("stop"); |
|  |  |
|  | } |
|  |  |
|  | } catch (IOException e) { |
|  | System.out.println("Read failed"); |
|  | System.exit(-1); |
|  | } |
|  | } |
|  |  |
|  | // Check whether the name provide by the client is a valid name |
|  | // and get a valid name eventually and add the user to the user list. |
|  | public static synchronized User userConnectionCheck (BufferedReader in, PrintWriter out, User user, String status) { |
|  |  |
|  | int index; |
|  | String line = new String(); |
|  |  |
|  | while (true) { |
|  |  |
|  | try { |
|  |  |
|  | line = in.readLine(); |
|  |  |
|  | } catch (IOException e) { |
|  |  |
|  | System.out.println("Read failed"); |
|  | System.exit(-1); |
|  |  |
|  | } |
|  |  |
|  | index = checkAUserStatus(line); |
|  | if (index == -1) { |
|  |  |
|  | out.println("valid"); |
|  | user.setName(line); |
|  | addUser(user, out); |
|  | break; |
|  |  |
|  | } else { |
|  |  |
|  | if(!checkAUserConnection(index)) { |
|  |  |
|  | out.println("valid"); |
|  | user = userList.get(index); |
|  | break; |
|  |  |
|  | } else { |
|  |  |
|  | out.println("invalid"); |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | System.out.println(getTime() + ", " + "Connection by " + user.status + " user " + line); |
|  | user.setStatus(status); |
|  | user.setConnected(true); |
|  | return user; |
|  |  |
|  | } |
|  |  |
|  |  |
|  | public static String getTime(){ |
|  | DateFormat df = new SimpleDateFormat("MM/dd/yy HH:mm a"); |
|  | Date dateobj = new Date(); |
|  | String time = df.format(dateobj); |
|  | return time; |
|  | } |
|  |  |
|  | public synchronized static void addUser(User currentUser, PrintWriter out) { |
|  |  |
|  | if (userList.size() < 5) { |
|  |  |
|  | userList.add(currentUser); |
|  |  |
|  | } else { |
|  |  |
|  | out.println("full"); |
|  | } |
|  |  |
|  |  |
|  | } |
|  |  |
|  | public synchronized static boolean storeMessage(User user, String message, PrintWriter out) { |
|  |  |
|  | if (userList.size() < 5) { |
|  |  |
|  | user.setMessage(message); |
|  | return true; |
|  |  |
|  | } else { |
|  |  |
|  | out.println("full"); |
|  | return false; |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | public static int checkAUserStatus(String name) { |
|  |  |
|  | int index = -1; |
|  |  |
|  | for(int i = 0; i < userList.size(); i++) { |
|  |  |
|  | if(userList.get(i).name.equals(name)) { |
|  | index = i; |
|  | break; |
|  | } |
|  |  |
|  | } |
|  |  |
|  | return index; |
|  |  |
|  | } |
|  |  |
|  | public static boolean checkAUserConnection(int index) { |
|  |  |
|  | boolean connection = false; |
|  |  |
|  | if (userList.get(index).connected) { |
|  |  |
|  | connection = true; |
|  |  |
|  | } |
|  |  |
|  | return connection; |
|  |  |
|  | } |
|  |  |
|  |  |
|  | public void checkAllKnownUsers (String choice, String message) { |
|  |  |
|  | int i = 0; |
|  | while(i < userList.size()) { |
|  |  |
|  | if (choice.equals("1")) { |
|  |  |
|  | out.println(userList.get(i).name); |
|  |  |
|  | } else { |
|  |  |
|  | if (!userList.get(i).name.equals(newUser.name)) { |
|  |  |
|  | boolean saved = storeMessage(userList.get(i), message, out); |
|  | if(!saved) { |
|  |  |
|  | out.println(userList.get(i) + "'s message is full and can't be stored"); |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | i++; |
|  |  |
|  | } |
|  |  |
|  | if(choice.equals("1")) { |
|  |  |
|  | System.out.println(newUser.name + " displays all known users."); |
|  |  |
|  | } else { |
|  |  |
|  | System.out.println(newUser.name + " posts a message for all known users."); |
|  | out.println("Message posted to all known users."); |
|  |  |
|  | } |
|  |  |
|  |  |
|  | } |
|  |  |
|  | public void checkAllConnectedUsers(String choice, String message) { |
|  |  |
|  | int i = 0; |
|  | while(i < userList.size()) { |
|  |  |
|  | if(userList.get(i).connected) { |
|  |  |
|  | if(choice.equals("2")) { |
|  |  |
|  | out.println(userList.get(i).name); |
|  |  |
|  | } else { |
|  |  |
|  | if (!userList.get(i).name.equals(newUser.name)) { |
|  |  |
|  | String time = getTime(); |
|  | String sendMessage = newUser.name + "\n" + time + "\n" + message; |
|  | boolean saved = storeMessage(userList.get(i), sendMessage, out); |
|  | if(!saved) { |
|  |  |
|  | out.println(userList.get(i) + ".s message is full and can't be stored."); |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | i++; |
|  |  |
|  | } |
|  |  |
|  | if(choice.equals("2")) { |
|  |  |
|  | System.out.println(newUser.name + " displays all connected users."); |
|  |  |
|  | } else { |
|  |  |
|  | System.out.println(newUser.name + " posts a message for all currently connected users."); |
|  | out.println("Message posted to all currently connected users."); |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | // Check whether it's a new user, if it, add him to the user List, otherwise get the |
|  | // user from the user list. |
|  | public User updateUser (int index, String name) { |
|  |  |
|  | User user = new User(); |
|  | if (index == -1) { |
|  |  |
|  | user.setName(name); |
|  | user.setStatus(status); |
|  | addUser(user, out); |
|  |  |
|  | } else { |
|  |  |
|  | user = userList.get(index); |
|  |  |
|  | } |
|  |  |
|  | return user; |
|  |  |
|  | } |
|  |  |
|  | public void exitClient() { |
|  | try |
|  | { |
|  | client.close(); |
|  | } |
|  | catch (IOException e) |
|  | { |
|  | System.out.println("Close failed"); |
|  | System.exit(-1); |
|  | } |
|  | } |
|  |  |
|  | } |