**ASSIGNMENT:9**

**Server:**

import java.io.DataInputStream;

import java.io.PrintStream;

import java.io.IOException;

import java.net.Socket;

import java.net.ServerSocket;

public class Server {

private static ServerSocket serverSocket = null;

private static Socket clientSocket = null;

private static final int maxClientsCount = 10;

private static final clientThread[] threads = new clientThread[maxClientsCount];

public static void main(String args[]) {

int portNumber = 6002;

if (args.length < 1) {

System.out.println("Usage: java MultiThreadChatServerSync <portNumber>\n"

+ "Now using port number=" + portNumber);

} else {

portNumber = Integer.valueOf(args[0]).intValue();

}

try {

serverSocket = new ServerSocket(portNumber);

} catch (IOException e) {

System.out.println(e);

}

while (true) {

try {

clientSocket = serverSocket.accept();

int i = 0;

for (i = 0; i < maxClientsCount; i++) {

if (threads[i] == null) {

(threads[i] = new clientThread(clientSocket, threads)).start();

break;

}

}

if (i == maxClientsCount) {

PrintStream os = new PrintStream(clientSocket.getOutputStream());

os.println("Server too busy. Try later.");

os.close();

clientSocket.close();

}

} catch (IOException e) {

System.out.println(e);

}

}

}

}

class clientThread extends Thread {

private String clientName = null;

private DataInputStream is = null;

private PrintStream os = null;

private Socket clientSocket = null;

private final clientThread[] threads;

private int maxClientsCount;

public clientThread(Socket clientSocket, clientThread[] threads) {

this.clientSocket = clientSocket;

this.threads = threads;

maxClientsCount = threads.length;

}

public void run() {

int maxClientsCount = this.maxClientsCount;

clientThread[] threads = this.threads;

try {

is = new DataInputStream(clientSocket.getInputStream());

os = new PrintStream(clientSocket.getOutputStream());

String name;

while (true) {

os.println("Enter your name.");

name = is.readLine().trim();

if (name.indexOf('@') == -1) {

break;

} else {

os.println("The name should not contain '@' character.");

}

}

os.println("Welcome " + name

+ " to our chat room.\nTo leave enter /quit in a new line.");

synchronized (this) {

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] != null && threads[i] == this) {

clientName = "@" + name;

break;

}

}

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] != null && threads[i] != this) {

threads[i].os.println("\*\*\* A new user " + name

+ " entered the chat room !!! \*\*\*");

}

}

}

while (true) {

String line = is.readLine();

if (line.startsWith("/quit")) {

break;

}

if (line.startsWith("@")) {

String[] words = line.split("\\s", 2);

if (words.length > 1 && words[1] != null) {

words[1] = words[1].trim();

if (!words[1].isEmpty()) {

synchronized (this) {

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] != null && threads[i] != this

&& threads[i].clientName != null

&& threads[i].clientName.equals(words[0])) {

threads[i].os.println("<" + name + "> " + words[1]);

this.os.println(">" + name + "> " + words[1]);

break;

}

}

}

}

}

} else {

synchronized (this) {

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] != null && threads[i].clientName != null) {

threads[i].os.println("<" + name + "> " + line);

}

}

}

}

}

synchronized (this) {

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] != null && threads[i] != this

&& threads[i].clientName != null) {

threads[i].os.println("\*\*\* The user " + name

+ " is leaving the chat room !!! \*\*\*");

}

}

}

os.println("\*\*\* Bye " + name + " \*\*\*");

synchronized (this) {

for (int i = 0; i < maxClientsCount; i++) {

if (threads[i] == this) {

threads[i] = null;

}

}

}

is.close();

os.close();

clientSocket.close();

} catch (IOException e) {

}

}

}

**Client:**

**import java.io.DataInputStream;**

import java.io.PrintStream;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.IOException;

import java.net.Socket;

import java.net.UnknownHostException;

public class TcpClient1 implements Runnable {

private static Socket clientSocket = null;

private static PrintStream os = null;

private static DataInputStream is = null;

private static BufferedReader inputLine = null;

private static boolean closed = false;

public static void main(String[] args) {

int portNumber = 6002;

String host = "localhost";

if (args.length < 2) {

System.out

.println("Usage: java MultiThreadChatClient <host> <portNumber>\n"

+ "Now using host=" + host + ", portNumber=" + portNumber);

} else {

host = args[0];

portNumber = Integer.valueOf(args[1]).intValue();

}

try {

clientSocket = new Socket(host, portNumber);

inputLine = new BufferedReader(new InputStreamReader(System.in));

os = new PrintStream(clientSocket.getOutputStream());

is = new DataInputStream(clientSocket.getInputStream());

} catch (UnknownHostException e) {

System.err.println("Don't know about host " + host);

} catch (IOException e) {

System.err.println("Couldn't get I/O for the connection to the host "

+ host);

}

if (clientSocket != null && os != null && is != null) {

try {

new Thread(new TcpClient1()).start();

while (!closed) {

os.println(inputLine.readLine().trim());

}

os.close();

is.close();

clientSocket.close();

} catch (IOException e) {

System.err.println("IOException: " + e);

}

}

}

public void run() {

String responseLine;

try {

while ((responseLine = is.readLine()) != null) {

System.out.println(responseLine);

if (responseLine.indexOf("\*\*\* Bye") != -1)

break;

}

closed = true;

} catch (IOException e) {

System.err.println("IOException: " + e);

}

}

}

**Output:**

**Server:**

aj@aj:~$ cd Downloads/

aj@aj:~/Downloads$ ls

android-studio Server.java tcpserver.java

as3dbmstheory.docx TcpClient1.java UDPClient.java

eclipse-installer tcpclient.java UDPServer.java

aj@aj:~/Downloads$ javac Server.java

Note: Server.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation for details.

aj@aj:~/Downloads$ java Server

Usage: java MultiThreadChatServerSync <portNumber>

Now using port number=6002

**Client1:**

aj@aj:~$ cd Downloads/

aj@aj:~/Downloads$ javac TcpClient1.java

Note: TcpClient1.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation for details.

aj@aj:~/Downloads$ java TcpClient1

Usage: java MultiThreadChatClient <host> <portNumber>

Now using host=localhost, portNumber=6002

Enter your name.

Aj

Welcome Aj to our chat room.

To leave enter /quit in a new line.

Hii,this is aj

<Aj> Hii,this is aj

\*\*\* A new user Milind entered the chat room !!! \*\*\*

<Milind> Milind Here

Welcome

<Aj> Welcome

\*\*\* A new user Mahesh entered the chat room !!! \*\*\*

<Mahesh> Hello!!!!

<Milind> hii

bye

<Aj> bye

**Client2:**

aj@aj:~/Downloads$ java TcpClient1

Usage: java MultiThreadChatClient <host> <portNumber>

Now using host=localhost, portNumber=6002

Enter your name.

Milind

Welcome Milind to our chat room.

To leave enter /quit in a new line.

Milind Here

<Milind> Milind Here

<Aj> Welcome

\*\*\* A new user Mahesh entered the chat room !!! \*\*\*

<Mahesh> Hello!!!!

hii

<Milind> hii

<Aj> bye

**Client3:**

aj@aj:~$ cd Downloads/

aj@aj:~/Downloads$ javac TcpClient1.java

Note: TcpClient1.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation for details.

aj@aj:~/Downloads$ java TcpClient1

Usage: java MultiThreadChatClient <host> <portNumber>

Now using host=localhost, portNumber=6002

Enter your name.

Mahesh

Welcome Mahesh to our chat room.

To leave enter /quit in a new line.

Hello!!!!

<Mahesh> Hello!!!!

<Milind> hii

<Aj> bye