1. **Design a simple client server application using socket programming in java.**

**Code:**

**Server.java:**

package edu.siescoms.simpleCS1;

import java.io.\*;

import java.net.\*;

public class Server {

public static void main(String[] args) throws IOException{

ServerSocket serversocket = new ServerSocket(1050);

System.out.println("Waiting for Client......");

Socket clientsocket = serversocket.accept();

System.out.println("Client accepted");

DataInputStream cis = new DataInputStream(clientsocket.getInputStream());

DataOutputStream cos= new DataOutputStream(clientsocket.getOutputStream());

cos.writeUTF("You are connected");

System.out.println(cis.readUTF());

}

}

**Client.java:**

package edu.siescoms.simpleCS1;

import java.io.\*;

import java.net.\*;

public class Client {

public static void main(String args[]) throws IOException

{

Socket client = new Socket("127.0.0.1",1050);

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

DataOutputStream os= new DataOutputStream(client.getOutputStream());

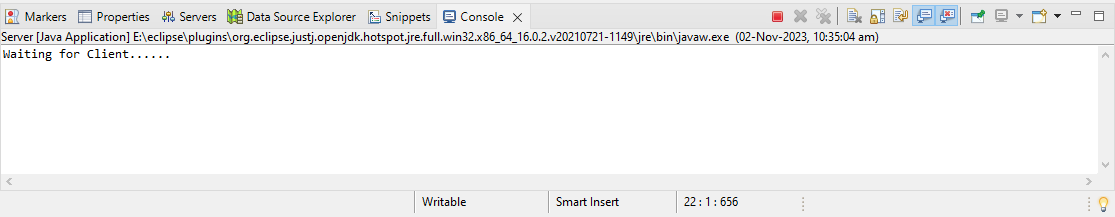
System.out.println(is.readUTF());

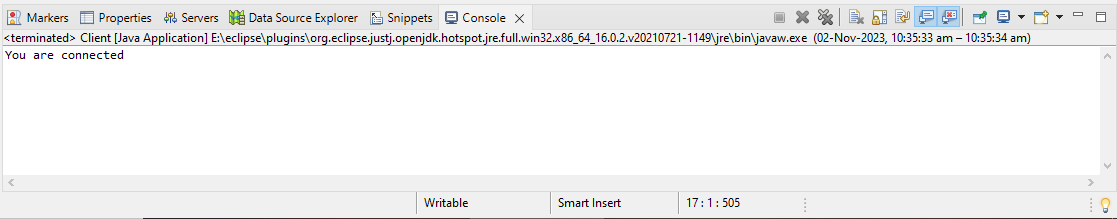
os.writeUTF("Hi Server - Thanks for accepting my connection");

}

}

**Output:**





1. **Design a simple client server application using socket programming consisting of a server and two clients.**

Code:

**Server.java:**

package edu.siescoms.multiCS1;

import java.io.\*;

import java.util.\*;

import java.net.\*;

public class Server {

public static void main(String[] args) throws IOException

{

ServerSocket serversoc = new ServerSocket(1050);

while (true)

{

Socket s = null;

s = serversoc.accept();

System.out.println("A new client is connected : " + s);

DataInputStream dis = new DataInputStream(s.getInputStream());

DataOutputStream dos = new DataOutputStream(s.getOutputStream());

Thread t = new ClientHandler(s, dis, dos);

t.start();

System.out.println("Waiting for client.....");

}

}

}

class ClientHandler extends Thread

{

DataInputStream dis;

DataOutputStream dos;

Socket s;

public ClientHandler(Socket s, DataInputStream dis, DataOutputStream dos)

{

this.s = s;

this.dis = dis;

this.dos = dos;

}

public void run()

{

try{

Scanner sc= new Scanner(System.in);

String msg=" ";

String sendmsg=" ";

while(!msg.equals("Exit"))

{

sendmsg=sc.nextLine();

dos.writeUTF(sendmsg);

msg=dis.readUTF();

System.out.println(s+msg);

}

}

catch(IOException e)

{

System.out.println("Message was not sent!!!");

}

}

}

**Client1.java:**

package edu.siescoms.multiCS1;

import java.io.\*;

import java.net.\*;

import java.util.Scanner;

public class Client1 {

public static void main(String[] args) throws IOException

{

Scanner sc = new Scanner(System.in);

Socket client = new Socket("localhost", 1050);

DataInputStream sis = new DataInputStream(client.getInputStream());

DataOutputStream sos= new DataOutputStream(client.getOutputStream());

String rcvmsg="";

String sendmsg="";

while(!rcvmsg.equals("Exit"))

{

rcvmsg=sis.readUTF();

System.out.println(rcvmsg);

sendmsg=sc.nextLine();

sos.writeUTF(sendmsg);

}

client.close();

}

}

**Client2.java:**

package edu.siescoms.multiCS1;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.Socket;

import java.util.Scanner;

public class Client2 {

public static void main(String[] args) throws IOException

{

Scanner sc = new Scanner(System.in);

Socket client = new Socket("localhost", 1050);

DataInputStream sis = new DataInputStream(client.getInputStream());

DataOutputStream sos= new DataOutputStream(client.getOutputStream());

String rcvmsg="";

String sendmsg="";

System.out.println("Client 2");

while(!rcvmsg.equals("Exit"))

{

rcvmsg=sis.readUTF();

System.out.println(rcvmsg);

sendmsg=sc.nextLine();

sos.writeUTF(sendmsg);

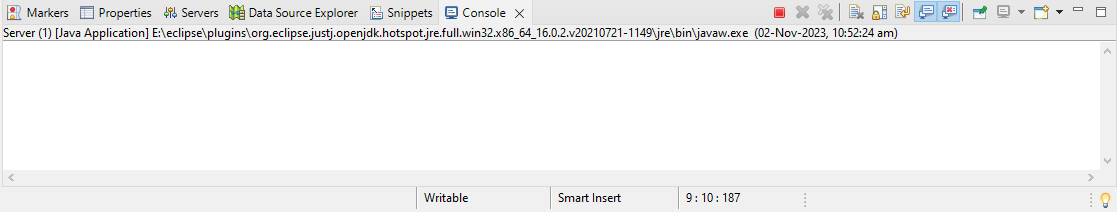
}

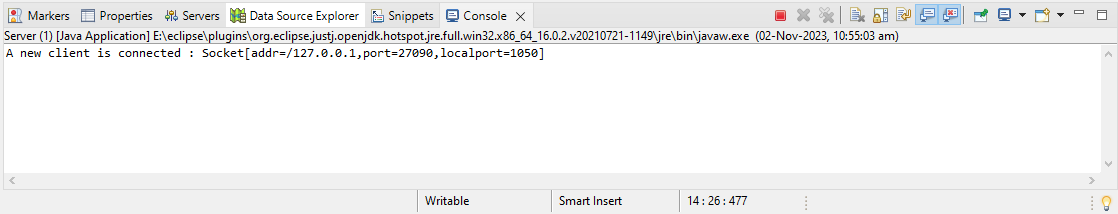
client.close();

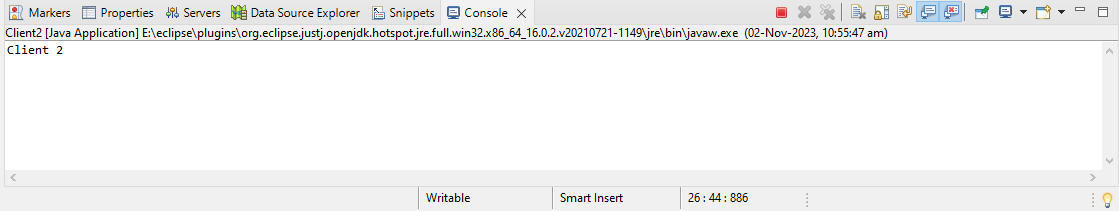
}

}

**Output:**







1. **Design a multiple clients chat server application using socket programming and multithreading mechanism in java**

**Code:**

**Server.java:**

package edu.siescoms.multiCS2;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.ArrayList;

import java.util.HashMap;

public class Server {

public static void main(String[] args) {

ArrayList<Socket> clients = new ArrayList<>();

HashMap<Socket, String> clientNameList = new HashMap<Socket, String>();

try (ServerSocket serversocket = new ServerSocket(5000)) {

System.out.println("Server is started...");

while (true) {

Socket socket = serversocket.accept();

clients.add(socket);

ThreadServer ThreadServer = new ThreadServer(socket, clients, clientNameList);

ThreadServer.start();

}

} catch (Exception e) {

System.out.println(e.getStackTrace());

}

}

}

**Client.java:**

package edu.siescoms.multiCS2;

import java.io.PrintWriter;

import java.net.Socket;

import java.util.Scanner;

public class Client {

public static void main(String[] args) {

String name = "empty";

String reply = "empty";

Scanner sc = new Scanner(System.in);

System.out.println("Enter your name (Please enter your name to join the chat): ");

reply = sc.nextLine();

name = reply;

try (Socket socket = new Socket("localhost", 5000)) {

PrintWriter cout = new PrintWriter(socket.getOutputStream(), true);

ThreadClient threadClient = new ThreadClient(socket);

new Thread(threadClient).start(); // start thread to receive message

cout.println(reply + ": has joined chat-room.");

do {

String message = (name + " : ");

reply = sc.nextLine();

if (reply.equals("logout")) {

cout.println("logout");

break;

}

cout.println(message + reply);

} while (!reply.equals("logout"));

} catch (Exception e) {

System.out.println(e.getStackTrace());

}

}

}

**ThreadServer.java:**

package edu.siescoms.multiCS2;

import java.io.\*;

import java.net.Socket;

import java.net.SocketException;

import java.util.ArrayList;

import java.util.HashMap;

public class ThreadServer extends Thread {

private Socket socket;

private ArrayList<Socket> clients;

private HashMap<Socket, String> clientNameList;

public ThreadServer(Socket socket, ArrayList<Socket> clients, HashMap<Socket, String> clientNameList) {

this.socket = socket;

this.clients = clients;

this.clientNameList = clientNameList;

}

@Override

public void run() {

try {

BufferedReader input = new BufferedReader(new InputStreamReader(socket.getInputStream()));

while (true) {

String outputString = input.readLine();

if (outputString.equals("logout")) {

throw new SocketException();

}

if (!clientNameList.containsKey(socket)) {

String[] messageString = outputString.split(":", 2);

clientNameList.put(socket, messageString[0]);

System.out.println(messageString[0] + messageString[1]);

showMessageToAllClients(socket, messageString[0] + messageString[1]);

} else {

System.out.println(outputString);

showMessageToAllClients(socket, outputString);

}

}

} catch (SocketException e) {

String printMessage = clientNameList.get(socket) + " left the chat room";

System.out.println(printMessage);

showMessageToAllClients(socket, printMessage);

clients.remove(socket);

clientNameList.remove(socket);

} catch (Exception e) {

System.out.println(e.getStackTrace());

}

}

private void showMessageToAllClients(Socket sender, String outputString) {

Socket socket;

PrintWriter printWriter;

int i = 0;

while (i < clients.size()) {

socket = clients.get(i);

i++;

try {

if (socket != sender) {

printWriter = new PrintWriter(socket.getOutputStream(), true);

printWriter.println(outputString);

}

} catch (IOException ex) {

System.out.println(ex);

}

}

}

}

**ThreadClient.java:**

package edu.siescoms.multiCS2;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.net.Socket;

import java.net.SocketException;

public class ThreadClient implements Runnable{

private Socket socket;

private BufferedReader cin;

public ThreadClient(Socket socket) throws IOException {

this.socket = socket;

this.cin = new BufferedReader(new InputStreamReader(socket.getInputStream()));

}

@Override

public void run() {

try {

while (true) {

String message = cin.readLine();

System.out.println(message);

}

} catch (SocketException e) {

System.out.println("You left the chat-room");

} catch (IOException exception) {

System.out.println(exception);

} finally {

try {

cin.close();

} catch (Exception exception) {

System.out.println(exception);

}

}

}

}

**Output:**

