**Multithreaded Server Socket program in Java**

import java.net.\*;

import java.io.\*;

public class MultithreadedSocketServer {

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

try{

ServerSocket server=new ServerSocket(8888);

int counter=0;

System.out.println("Server Started ....");

while(true){

counter++;

Socket serverClient=server.accept(); //server accept the client connection request

System.out.println(" >> " + "Client No:" + counter + " started!");

ServerClientThread sct = new ServerClientThread(serverClient,counter); //send the request to a separate thread

sct.start();

}

}catch(Exception e){

System.out.println(e);

}

}

}

**Server Client program**

This Server Client thread class handled the request independent of any other incoming requests. The following Java program is the part of Multithreaded Server Socket program.

class ServerClientThread extends Thread {

Socket serverClient;

int clientNo;

int squre;

ServerClientThread(Socket inSocket,int counter){

serverClient = inSocket;

clientNo=counter;

}

public void run(){

try{

DataInputStream inStream = new DataInputStream(serverClient.getInputStream());

DataOutputStream outStream = new DataOutputStream(serverClient.getOutputStream());

String clientMessage="", serverMessage="";

while(!clientMessage.equals("bye")){

clientMessage=inStream.readUTF();

System.out.println("From Client-" +clientNo+ ": Number is :"+clientMessage);

squre = Integer.parseInt(clientMessage) \* Integer.parseInt(clientMessage);

serverMessage="From Server to Client-" + clientNo + " Square of " + clientMessage + " is " +squre;

outStream.writeUTF(serverMessage);

outStream.flush();

}

inStream.close();

outStream.close();

serverClient.close();

}catch(Exception ex){

System.out.println(ex);

}finally{

System.out.println("Client -" + clientNo + " exit!! ");

}

}

}

**Client Program**

This is the real Client program that request connection to the server. For each Client, you need to open separate console window to run the client program.

import java.net.\*;

import java.io.\*;

public class TCPClient {

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

try{

Socket socket=new Socket("127.0.0.1",8888);

DataInputStream inStream=new DataInputStream(socket.getInputStream());

DataOutputStream outStream=new DataOutputStream(socket.getOutputStream());

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

String clientMessage="",serverMessage="";

while(!clientMessage.equals("bye")){

System.out.println("Enter number :");

clientMessage=br.readLine();

outStream.writeUTF(clientMessage);

outStream.flush();

serverMessage=inStream.readUTF();

System.out.println(serverMessage);

}

outStream.close();

outStream.close();

socket.close();

}catch(Exception e){

System.out.println(e);

}

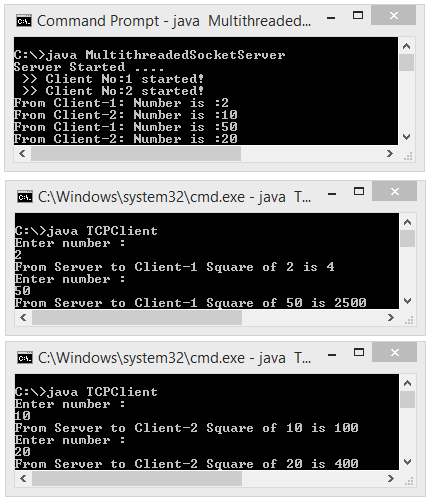
}

}

**How to run this program ?**

When you finish coding and compiled the MultithreadedSocketServer, ServerClientThread and TCPClient program, first you have to start MultithreadedSocketServer Program from **DOS prompt** (console window) , then you will get a message " Server Started..." in your DOS screen, where the server program is running .

Next step is to start Java TCPClient Socket Program in the same computer or other computers on the same network . When you start the client program , it will establish a connection to the **Server** and waiting input from client side. The **client** program repeatedly asks the user to input an integer, sends to the server, and receives the square of the integer from the server. If you want to test multiple client, for each Client, you need to open separate console window to run the client program. When the client send "bye" from client side the server closes the connection from client. From the following image, you can see how the Server and multiple clients communicate to the server.



If your Server and Client program running on same machine, then give "127.0.0.1".

Socket socket=new Socket("127.0.0.1",8888);

Otherwise give the IP Address of the machine that MultithreadedSocketServer is running on.