**TCP/IP Server example in Java**

The ServerSocket class is used to create servers. Typically we create the ServerSocket instance with the port number to be published for client connections. ServerSocket has the accept() method which waits for client connections. In the example below we use a ClientHandler thread which simply reads the message from client and writes back a message to client.

package com.netparam.server;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

public class JavaServer {

private static class ClientHandler extends Thread {

private Socket socket;

ClientHandler(Socket socket) {

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

this.socket = socket;

}

@Override

public void run() {

try {

// Reader and writer

BufferedReader reader = new BufferedReader

(new InputStreamReader(socket.getInputStream()));

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

// Read message from client

System.out.println(reader.readLine());

// Write a message back to client

writer.println("Hello from server");

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

socket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

public static void main ( String[] args ) {

final int port = 8888;

try ( ServerSocket ss = new ServerSocket(port) ) {

System.out.println("Listening ...");

while ( true ) {

Socket socket = ss.accept();

new ClientHandler(socket).start();

}

} catch (IOException e) {

e.printStackTrace();

}

}

}