Server side program

import java.net.\*;

public class Server {

public static void main(String[] args) {

DatagramSocket socket = null;

byte[] receiveData = new byte[1024];

try {

// Create a DatagramSocket to listen on port 9876

socket = new DatagramSocket(9876);

System.out.println("Server is listening on port 9876...");

while (true) {

// Receive the message from the client

DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);

socket.receive(receivePacket); // blocking call

String message = new String(receivePacket.getData(), 0, receivePacket.getLength());

System.out.println("Client: " + message);

// Get the client's IP and port from the received packet

InetAddress clientAddress = receivePacket.getAddress();

int clientPort = receivePacket.getPort();

// Send a response to the client

String response = "Message received: " + message;

byte[] sendData = response.getBytes();

DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, clientAddress, clientPort);

socket.send(sendPacket);

}

} catch (Exception e) {

e.printStackTrace();

} finally {

if (socket != null && !socket.isClosed()) {

socket.close();

}

}

}

}

Client side program

import java.net.\*;

import java.util.Scanner;

public class Client {

public static void main(String[] args) {

DatagramSocket socket = null;

Scanner scanner = new Scanner(System.in);

try {

// Create a DatagramSocket

socket = new DatagramSocket();

// Server address and port

InetAddress serverAddress = InetAddress.getByName("localhost");

int serverPort = 9876;

while (true) {

// Take user input (message)

System.out.print("Enter message (or 'exit' to quit): ");

String message = scanner.nextLine();

// Send message to the server

byte[] sendData = message.getBytes();

DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, serverAddress, serverPort);

socket.send(sendPacket);

if (message.equalsIgnoreCase("exit")) {

System.out.println("Exiting chat...");

break;

}

// Receive the server's response

byte[] receiveData = new byte[1024];

DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);

socket.receive(receivePacket); // blocking call

String serverMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());

System.out.println("Server: " + serverMessage);

}

} catch (Exception e) {

e.printStackTrace();

} finally {

if (socket != null && !socket.isClosed()) {

socket.close();

}

scanner.close();

}

}

}

javac Server.java

javac Client.java

java server

java client