**19CSE301 - COMPUTER NETWORKS**

**Socket Programming**

**LAB-4 : (10-08-2021)**

* **R.Abhinav**
* **CB.EN.U4CSE19453**

1. **Implement the simple UDP client-server**

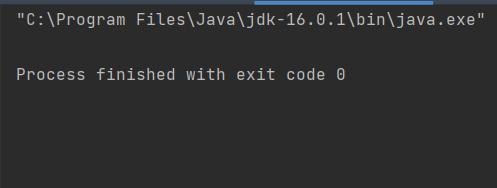
**Client:**

import java.io.\*;  
import java.net.\*;  
  
class UDPClient {  
 public static void main(String args[]) throws Exception  
 {  
  
 BufferedReader inFromUser =  
 new BufferedReader(new InputStreamReader(System.*in*));  
  
 DatagramSocket clientSocket = new DatagramSocket();  
  
 InetAddress IPAddress = InetAddress.*getByName*("hostname");  
  
 byte[] sendData = new byte[1024];  
 byte[] receiveData = new byte[1024];  
  
 String sentence = inFromUser.readLine();  
  
 sendData = sentence.getBytes();  
  
 DatagramPacket sendPacket =  
 new DatagramPacket(sendData, sendData.length, IPAddress, 9876);  
  
 clientSocket.send(sendPacket);  
  
 DatagramPacket receivePacket =  
 new DatagramPacket(receiveData, receiveData.length);  
  
 clientSocket.receive(receivePacket);  
  
 String modifiedSentence =  
 new String(receivePacket.getData());  
  
 System.*out*.println("FROM SERVER:" + modifiedSentence);  
  
 clientSocket.close();  
  
 }  
}

**Server:**

import java.io.\*;  
import java.net.\*;  
  
class UDPServer {  
 public static void main(String args[]) throws Exception  
 {  
  
 DatagramSocket serverSocket = new DatagramSocket(9876);  
  
 byte[] receiveData = new byte[1024];  
 byte[] sendData = new byte[1024];  
  
 while(true)  
 {  
  
 DatagramPacket receivePacket =  
 new DatagramPacket(receiveData, receiveData.length);  
  
 serverSocket.receive(receivePacket);  
  
 String sentence = new String(receivePacket.getData());  
  
 InetAddress IPAddress = receivePacket.getAddress();  
  
 int port = receivePacket.getPort();  
  
 String capitalizedSentence = sentence.toUpperCase();  
  
 sendData = capitalizedSentence.getBytes();  
  
 DatagramPacket sendPacket =  
 new DatagramPacket(sendData, sendData.length, IPAddress,  
 port);  
  
 serverSocket.send(sendPacket);  
 }  
 }  
}

**Output:**

****

1. **Single Datagram:**

**Code :**

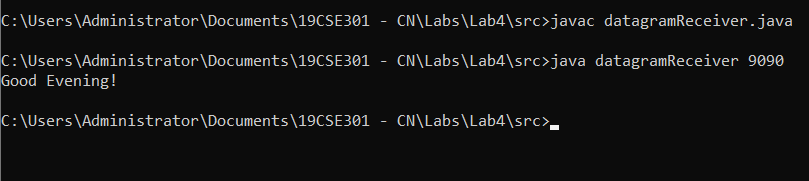
**Receiver:**

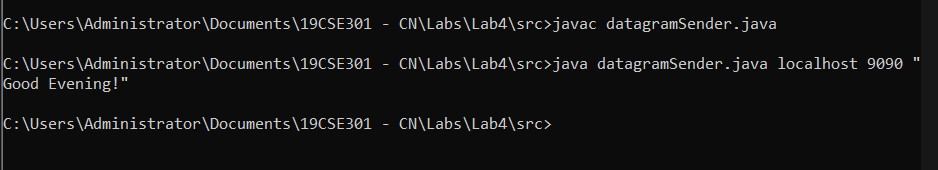
import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
  
class datagramReceiver {  
 public static void main(String[] args) {  
 try {  
 int MAX\_LEN = 40;  
 int localPortNum = Integer.*parseInt*(args[0]);  
 DatagramSocket mySocket = new DatagramSocket(localPortNum);  
 byte[] buffer = new byte[MAX\_LEN];  
 DatagramPacket packet = new DatagramPacket(buffer, MAX\_LEN);  
 mySocket.receive(packet);  
 String message = new String(buffer);  
 System.*out*.println(message);  
 mySocket.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Sender:**

import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
import java.net.InetAddress;  
  
class datagramSender {  
 public static void main(String[] args) {  
 try {  
 InetAddress receiverHost = InetAddress.*getByName*(args[0]);  
 int receiverPort = Integer.*parseInt*(args[1]);  
 String message = args[2];  
 DatagramSocket mySocket = new DatagramSocket();  
 byte[] buffer = message.getBytes();  
 DatagramPacket packet = new DatagramPacket(buffer, buffer.length, receiverHost,  
 receiverPort);  
 mySocket.send(packet);  
 mySocket.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Output:**

****

****

1. **Multi Client Server :**

**Code:**

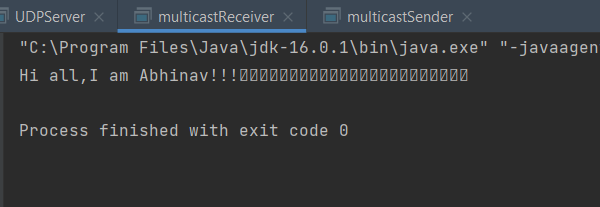
**Receiver :**

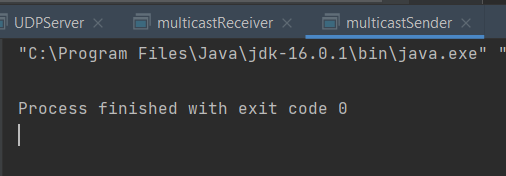
import java.net.DatagramPacket;  
import java.net.InetAddress;  
import java.net.MulticastSocket;  
  
class multicastReceiver {  
 public static void main(String[] args) {  
 try {  
 InetAddress group = InetAddress.*getByName*("224.0.0.1");  
 MulticastSocket multicastSock = new MulticastSocket(3456);  
 multicastSock.joinGroup(group);  
 byte[] buffer = new byte[45];  
 DatagramPacket packet = new DatagramPacket(buffer, buffer.length);  
 multicastSock.receive(packet);  
 System.*out*.println(new String(buffer));  
 multicastSock.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Sender:**

import java.net.DatagramPacket;  
import java.net.InetAddress;  
import java.net.MulticastSocket;  
  
class multicastSender {  
 public static void main(String[] args) {  
 try {  
 InetAddress group = InetAddress.*getByName*("224.0.0.1");  
 MulticastSocket multicastSock = new MulticastSocket(3456);  
 String msg = "Hi all,I am Abhinav!!!";  
 DatagramPacket packet = new DatagramPacket(msg.getBytes(), msg.length(), group,3456);  
 multicastSock.send(packet);  
 multicastSock.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Output :**

****

****

1. **Exercise :**

**Client:**

import java.io.IOException;  
import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
import java.net.InetAddress;  
import java.util.Scanner;  
  
public class UDPClient  
{  
 public static void main(String args[]) throws IOException  
 {  
 System.*out*.println("client started, enter input:");  
 Scanner sc = new Scanner(System.*in*);  
  
 // Step 1:Create the socket object for  
 // carrying the data.  
 DatagramSocket ds = new DatagramSocket();  
  
 InetAddress ip = InetAddress.*getLocalHost*();  
 byte buf[] = null;  
  
 // loop while user not enters "bye"  
 while (true)  
 {  
 String inp = sc.nextLine();  
  
 // convert the String input into the byte array.  
 buf = inp.getBytes();  
  
 // Step 2 : Create the datagramPacket for sending  
 // the data.  
 DatagramPacket DpSend =  
 new DatagramPacket(buf, buf.length, ip, 2345);  
  
 // Step 3 : invoke the send call to actually send  
 // the data.  
 ds.send(DpSend);  
  
 // break the loop if user enters "bye"  
 if (inp.equals("bye"))  
 break;  
 }  
 }  
}

**Server:**

//Java program to illustrate Server side  
//Implementation using DatagramSocket  
import java.io.IOException;  
import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
import java.net.InetAddress;  
import java.net.SocketException;  
  
public class UDPServer  
{  
 public static void main(String[] args) throws IOException  
 {  
 System.*out*.println("server started");  
 // Step 1 : Create a socket to listen at port 2345  
 DatagramSocket ds = new DatagramSocket(2345);  
 byte[] receive = new byte[65535];  
  
 DatagramPacket DpReceive = null;  
 while (true)  
 {  
  
 // Step 2 : create a DatgramPacket to receive the data.  
 DpReceive = new DatagramPacket(receive, receive.length);  
  
 // Step 3 : receive the data in byte buffer.  
 ds.receive(DpReceive);  
  
 System.*out*.println("Client:-" + *data*(receive));  
  
 // Exit the server if the client sends "bye"  
 if (*data*(receive).toString().equals("bye"))  
 {  
 System.*out*.println("Client sent bye.....EXITING");  
 break;  
 }  
  
 // Clear the buffer after every message.  
 receive = new byte[65535];  
 }  
 }  
  
 // A utility method to convert the byte array  
 // data into a string representation.  
 public static StringBuilder data(byte[] a)  
 {  
 if (a == null)  
 return null;  
 StringBuilder ret = new StringBuilder();  
 int i = 0;  
 while (a[i] != 0)  
 {  
 ret.append((char) a[i]);  
 i++;  
 }  
 return ret;  
 }  
}

**Output:**

|  |  |
| --- | --- |
| **Server** | **Client** |
|  |  |