Networking in java

IP Address

An IP address is unique identification number allocated to every computer on a network or internet. IP address contains some bytes which identifies the network and the actual computer inside the network.

What is DNS (Domain name system)?

DNS is a service on Internet that maps the IP addresses with corresponding website names.

Example of IP Address : 192.168.43.1

Sockets

A socket is a program is point of connection between a server and a client on a network.

Each socket is given an identification which is called port number. Length of port is 2Byte.

Port Number	Application
13	Data and time service
21	FTP - Transfer files
23	Telnet - Remote Login
25	SMTP - Delivers mails
67	BOOTP – Boot configuration
80	HTTP – Transfer web pages
109	POP- accesses mailboxes

Program 1 : To get IP of a website

```java
import java.io.*;
import java.net.*;
class Address{
    public static void main(String[] args)throws IOException {
        InputStreamReader ISR = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(ISR);
        System.out.println("Enter a website's URL : ");
        String url = br.readLine();
        try {
            InetAddress ip = InetAddress.getByName(url);
            System.out.println("IP : "+ip);
        } catch (UnknownHostException e) {
            System.out.println("Website not found");
        }
    }
}
```

URL

URL(Uniform resource locator) represents the address that is specified to access some information or resource on Internet.

URL is represented by a class URL in java.net package. To create a object of URL class, we can use the following formats:

```
URL obj = new URL(Protocol+Host+Port+Path);
```

<OR>

```
URL obj = new URL(Protocol+Host+Path);
```

Example :

```
URL obj = new URL("https://www.facebook.com/index.html");
```

Program 2 : To retrieve different parts of the URL.

myURL.java

```
 import java.io.*;

import java.net.*;
class myURL {
    public static void main(String[] args)throws IOException {
        URL obj = new URL("https://www.facebook.com/index.html");
        System.out.println("Protocol : "+obj.getHost());
        System.out.println("Host : "+obj.getHost());
        System.out.println("File : "+obj.getFile());
        System.out.println("Port : "+obj.getPort());
        System.out.println("Path : "+obj.getPath());
        System.out.println("External form : "+obj.toExternalForm());
    }
}
```

URL Connection

The URLConnection class in Java is a class that represents a connection to a remote resource, such as a file or web page, using a URL (Uniform Resource Locator). It allows a Java program to send and receive data from a remote location and can be used to open a connection to a URL, send data to a server, and retrieve data from a servern simple words, it is a class for connecting to a specific URL for data transfer. The programs should be executed on a computer where the Internet connection is enabled.

Program 3 : To get the details of a web-page

Details.java

```java
import java.io.*;
import java.net.*;
import java.util.*;
public class Details {
    public static void main(String args[]) throws Exception {
        URL obj = new URL("http://www.ctxt.io/index.html");
        URLConnection conn = obj.openConnection();
        System.out.println("Date:" + new Date(conn.getDate()));
        System.out.println("Connect-type" + conn.getContentType());
        System.out.println("Expity:" + conn.getExpiration());
        System.out.println("Last Modified:" + new
Date(conn.getLastModified()));
        int l = conn.getContentLength();
        System.out.println("ContentLength:" + l);
        if (l == 0) {
            System.out.println("content not available");
            return;
        } else {
            int ch;
            InputStream in = conn.getInputStream();
            while ((ch = in.read()) != -1)
                System.out.print((char) ch);
        }
    }
}
```

Creating a server that sends data in java

*At Server side, create a server with some port number.

```
ServerSocket ss = new ServerSocket(3000);
```

*Now, we should make the server wait till a client accepts connection.

```
Socket s = ss.accept();
```

*Attach output stream to the server socket using getOutputStream() method.

```
OutputStream obj = new s.getOutputStream();
```

*Take PrintStream object to send data till the socket.

```
PrintStream ps = new PrintStream(obj);
```

*Finally the PrintStream is used by server to send data to the client.

```
ps.println(str);
```

*Then close all connections etc.

```
ss.close();
br.close();
s.close();
obj.close();
```

Program 4 : Creating a server for purpose of sending strings to the client

```java
import java.io.*;
import java.net.*;
public class Server1 {
    public static void main(String args[]) throws Exception {
        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        String str;
        ServerSocket ss = new ServerSocket(3020);
        Socket s = ss.accept();
        System.out.println("connection established");
        OutputStream obj = s.getOutputStream();
        PrintStream ps = new PrintStream(obj);
        System.out.println("Enter your message : ");
        str = br.readLine();
        ps.println(str);
        br.close();
        ps.close();
        ss.close();
        s.close();
    }
}
```

Creating a client that Receives data in java

*First create an object of socket

```
Socket s = new Socket("IP Adress", Port number);
```

*Add inputStream to the socket so that socket will be able to receive the data.

```
InputStream obj = s.getInputStream();
```

*Add To read data from the socket into the client

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

*Now we can read data coming from server side

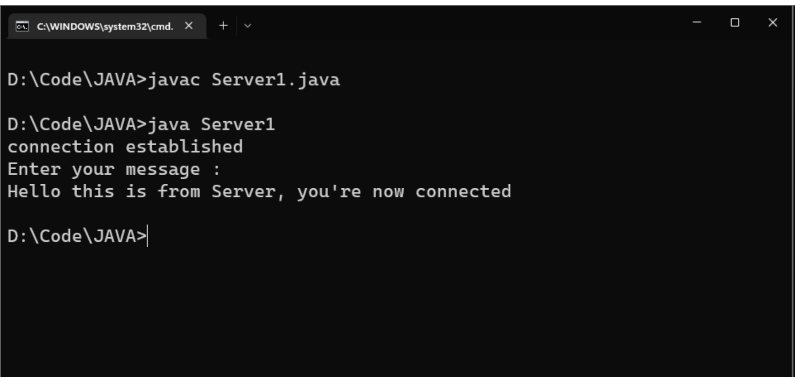
```
str = br.readLine();
```

Program 5: Creating a client-side program

```
import java.io.*;
import java.net.*;
public class Client1 {
    public static void main(String args[]) throws Exception {
        Socket s = new Socket("localhost", 3020);
        InputStream obj = s.getInputStream();
        BufferedReader br = new BufferedReader(new
InputStreamReader(obj));
        String str;
        while ((str = br.readLine()) != null)
            System.out.println(" from server :" + str);
        br.close();
        s.close();
    }
}
```

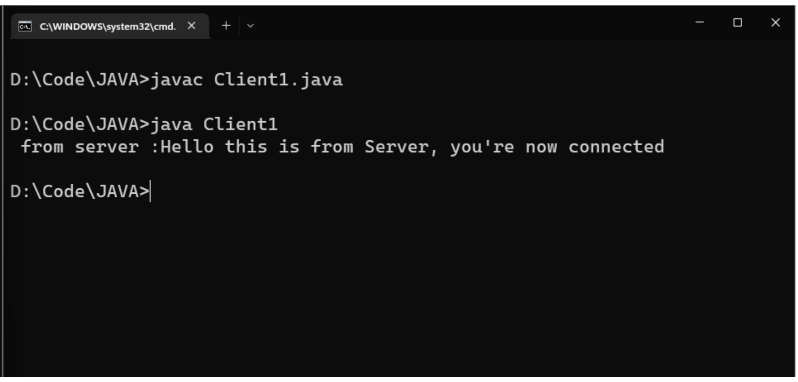
Expected output :

Server-Side



```
D:\Code\JAVA>javac Server1.java

D:\Code\JAVA>java Server1
connection established
Enter your message :
Hello this is from Server, you're now connected

D:\Code\JAVA>
```

Client side



```
D:\Code\JAVA>javac Client1.java

D:\Code\JAVA>java Client1
 from server :Hello this is from Server, you're now connected

D:\Code\JAVA>
```

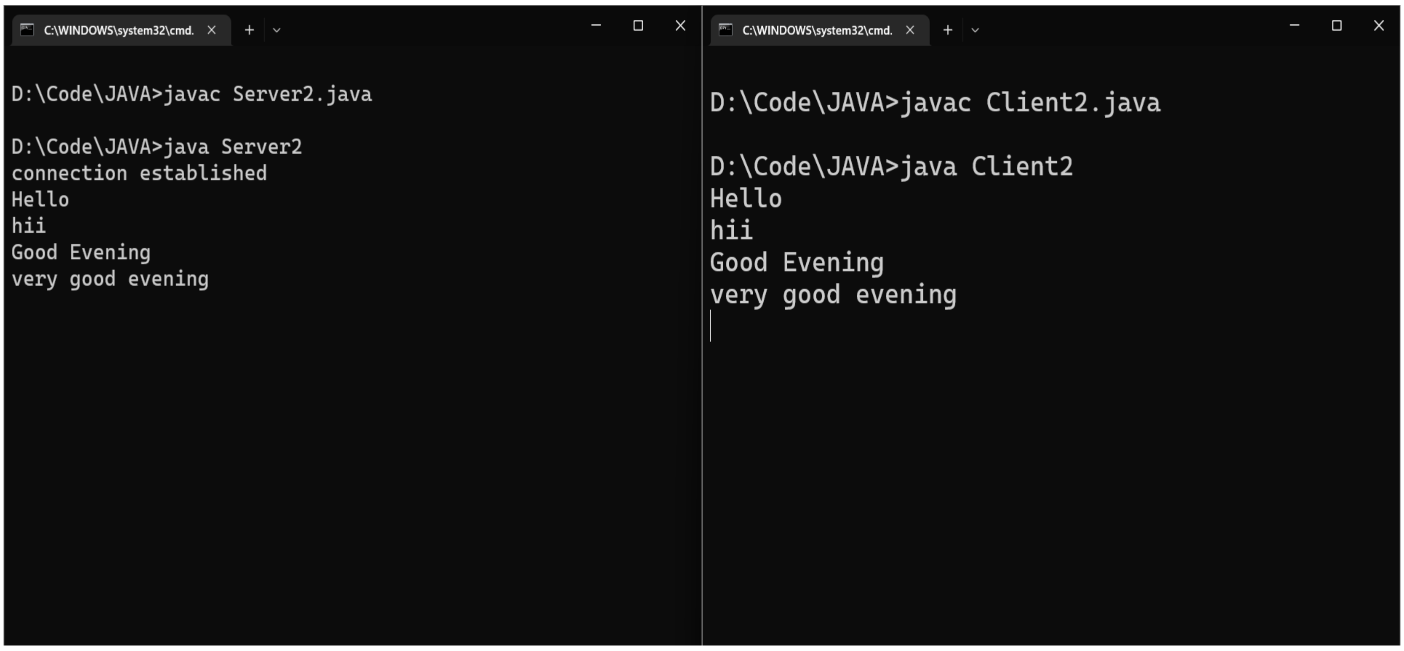
Program 6: Two-way communication between server and client(Server side)

```java
import java.io.*;
import java.net.*;
class Server2 {
    public static void main(String args[]) throws Exception {
        ServerSocket ss = new ServerSocket(1234);
        Socket s = ss.accept();
        System.out.println("connection established");
        PrintStream ps = new PrintStream(s.getOutputStream());
        BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        BufferedReader kb = new BufferedReader(new
InputStreamReader(System.in));
        while (true) {
            String str, str1;
            while ((str = br.readLine()) != null) {
                System.out.println(str);
                str1 = kb.readLine();
                ps.println(str1);
            }
            ps.close();
            br.close();
            kb.close();
            ss.close();
            s.close();
            System.exit(0);
        }
    }
}
```

Program 7: Two-way communication between server and client(Client side)

```java
import java.io.*;
import java.net.*;
class Client2 {
    public static void main(String args[]) throws Exception {
        Socket s = new Socket("localhost", 1234);
        DataOutputStream dos = new
DataOutputStream(s.getOutputStream());
        BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        BufferedReader kb = new BufferedReader(new
InputStreamReader(System.in));
        String str, str1;
        while (!(str = kb.readLine()).equals("exit")) {
            dos.writeBytes(str + "\n");
            str1 = br.readLine();
            System.out.println(str1);
        }
        dos.close();
        br.close();
        kb.close();
        s.close();
    }
}
```

OUTPUT :



```
D:\Code\JAVA>javac Server2.java

D:\Code\JAVA>java Server2
connection established
Hello
hii
Good Evening
very good evening
```

```
D:\Code\JAVA>javac Client2.java

D:\Code\JAVA>java Client2
Hello
hii
Good Evening
very good evening
```

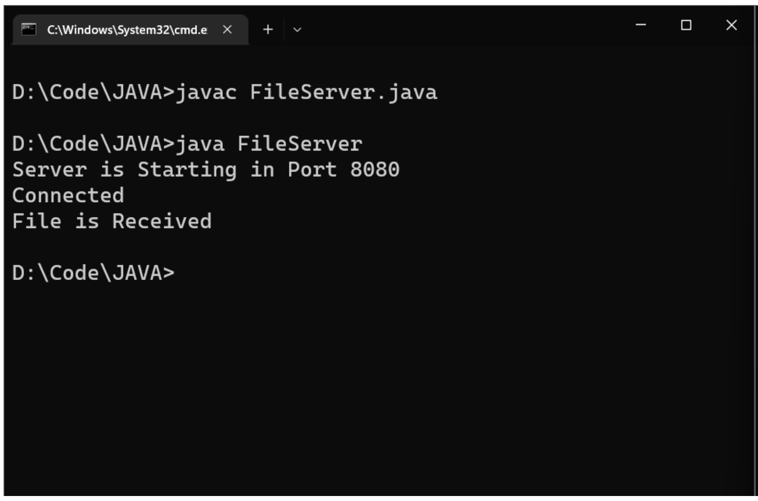
Program 8 : Transferring files using the sockets (Server side)

```java
import java.io.*;
import java.net.*;

public class FileServer {

    private static DataOutputStream dataOutputStream = null;
    private static DataInputStream dataInputStream = null;

    public static void main(String[] args) {
        try (ServerSocket serverSocket = new ServerSocket(8080)) {
            System.out.println("Server is Starting in Port 8080");
            Socket clientSocket = serverSocket.accept();
            System.out.println("Connected");
            dataInputStream = new DataInputStream(clientSocket.getInputStream());
            dataOutputStream = new DataOutputStream(clientSocket.getOutputStream());
            receiveFile("Received_file");

            dataInputStream.close();
            dataOutputStream.close();
            clientSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void receiveFile(String fileName) throws Exception {
        int bytes = 0;
        FileOutputStream fileOutputStream = new FileOutputStream(fileName);

        long size = dataInputStream.readLong();
        byte[] buffer = new byte[4 * 1024];
        while (size > 0 && (bytes = dataInputStream.read(buffer, 0,
                (int) Math.min(buffer.length, size))) != -1) {
            fileOutputStream.write(buffer, 0, bytes);
            size -= bytes;
        }
        System.out.println("File is Received");
        fileOutputStream.close();
    }
}
```

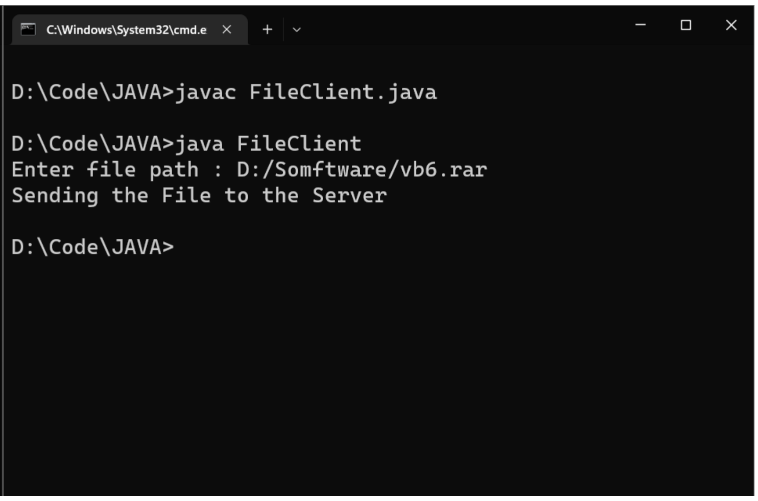
Program 9 : Transferring files using the sockets (Client side)

```
import java.io.*;
import java.net.*;

public class FileServer {

    private static DataOutputStream dataOutputStream = null;
    private static DataInputStream dataInputStream = null;

    public static void main(String[] args) {
        try (ServerSocket serverSocket = new ServerSocket(8080)) {
            System.out.println("Server is Starting in Port 8080");
            Socket clientSocket = serverSocket.accept();
            System.out.println("Connected");
            dataInputStream = new DataInputStream(clientSocket.getInputStream());
            dataOutputStream = new DataOutputStream(clientSocket.getOutputStream());
            receiveFile("Received_file");

            dataInputStream.close();
            dataOutputStream.close();
            clientSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void receiveFile(String fileName) throws Exception {
        int bytes = 0;
        FileOutputStream fileOutputStream = new FileOutputStream(fileName);

        long size = dataInputStream.readLong();
        byte[] buffer = new byte[4 * 1024];
        while (size > 0 && (bytes = dataInputStream.read(buffer, 0,
                (int) Math.min(buffer.length, size))) != -1) {
            fileOutputStream.write(buffer, 0, bytes);
            size -= bytes;
        }
        System.out.println("File is Received");
        fileOutputStream.close();
    }
}
```

OUTPUT



```
D:\Code\JAVA>javac FileServer.java

D:\Code\JAVA>java FileServer
Server is Starting in Port 8080
Connected
File is Received

D:\Code\JAVA>
```



```
D:\Code\JAVA>javac FileClient.java

D:\Code\JAVA>java FileClient
Enter file path : D:/Somftware/vb6.rar
Sending the File to the Server

D:\Code\JAVA>
```