



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

Name: Shivam Pandey
Roll No.: 28
Experiment No.1
Implement inter-process communication
Date of Performance: 09/01/2024
Date of Submission: 01/02/2024



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

Aim: To implement inter-process communication

Objective: Develop a program to implement inter-process communication using socket programming

Theory:-

URLs and URLConnections provide a relatively high-level mechanism for accessing resources on the Internet. Sometimes your programs require lower-level network communication, for example, when you want to write a client-server application.

In client-server applications, the server provides some service, such as processing database queries or sending out current stock prices. The client uses the service provided by the server, either displaying database query results to the user or making stock purchase recommendations to an investor. The communication that occurs between the client and the server must be reliable. That is, no data can be dropped and it must arrive on the client side in the same order in which the server sent it.

TCP provides a reliable, point-to-point communication channel that client-server applications on the Internet use to communicate with each other. To communicate over TCP, a client program and a server program establish a connection to one another. Each program binds a socket to its end of the connection. To communicate, the client and the server each reads from and writes to the socket bound to the connection.

What Is a Socket?

A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program. The java.net package provides two classes--Socket and ServerSocket--that implement the client side of the connection and the server side of the connection, respectively.

Socket class

A socket is simply an endpoint for communications between the machines. The Socket class can be used to create a socket.

Important methods

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.
2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

ServerSocket class

The ServerSocket class can be used to create a server socket. This object is used to establish communication with the clients.

Important methods

Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket.

Code and output:

MyServer.java

```
import java.io.*;
import java.net.*;
public class MyServer {
    public static void main(String[] args){
        try{
            ServerSocket ss=new ServerSocket(6666);
            Socket s=ss.accept();//establishes connection
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String str=(String)dis.readUTF();
            System.out.println("message= "+str);
            ss.close();
        }catch(Exception e){
            System.out.println(e);
        }
    }
}
```

MyClient.java

```
import java.io.*;
import java.net.*;
public class MyClient {
    public static void main(String[] args) {
        try{
            Socket s=new Socket("localhost",6666);
            DataOutputStream dout=new DataOutputStream(s.getOutputStream());
            dout.writeUTF("Welcome to DC Practicals");
            dout.flush();
        }
    }
}
```



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

```
dout.close();
s.close();
}
catch(Exception e){
    System.out.println(e);
}
}
```

A screenshot of a Windows Command Prompt window. The title bar says 'Command Prompt'. The window content shows the following text:

```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Student>cd Desktop

C:\Users\Student\Desktop>javac MyClient.java

C:\Users\Student\Desktop>java MyClient

C:\Users\Student\Desktop>|
```



```
Command Prompt
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Student>cd Desktop

C:\Users\Student\Desktop>javac MyServer.java

C:\Users\Student\Desktop>java MyServer
message= Welcome to DC Practicals

C:\Users\Student\Desktop>|
```

Conclusion: In conclusion, by creating connections between client and server processes, socket programming acts as a useful tool for inter-process communication (IPC). Sockets, as opposed to other IPC technologies like pipes or shared memory, allow communication between diverse systems and across networks. Socket-based message exchange between a client and server process illustrates the ease of use and flexibility of this technology for IPC.