

**CS334**  
**Network Programming Lab**  
**Government Engineering College Thrissur**  
**Experiment 1**

**AMARNATH C N**  
**S6 CSE**  
**Roll no: 10**  
**TCR18CS010**

## **AIM**

Implement Client-Server communication using Socket Programming and TCP as transport layer protocol.

## **THEORY**

The Transmission Control Protocol (TCP) is one of the main protocols of the Internet Protocol Suite. It originated in the initial network implementation in which it complemented the Internet Protocol(IP).

Therefore, the entire suite is commonly referred to as TCP/IP. TCP provides reliable,ordered, and error checked delivery of a stream of bytes between applications running on hosts communicating via an IP network.TCP is connection-oriented and a connection between client and server is established before data can be sent.

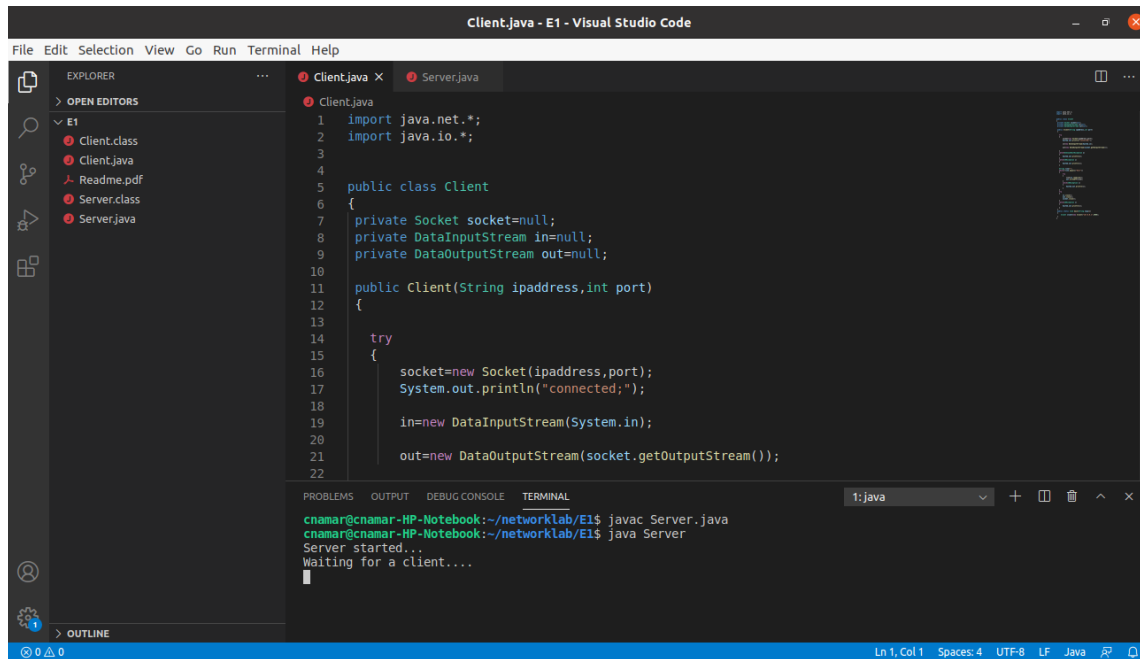
The server must be listening (passive open) for connection requests from clients before a connection is established.

## **HOW TO RUN**

- Open 2 terminal one for Client and other for Server.
- Server should be setup first and after that Client.
- Run Server application  
\$javac Server.java  
\$java Server  
Server is now started and is waiting for Client
- Run Client application  
\$javac Client.java  
\$java Client  
now connection is established

Client can give requests to server and Server can reply to it.  
Enter Over in Client to finish communication.

## **Screenshots**



Client.java - E1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

E1

- Client.class
- Client.java
- Readme.pdf
- Server.class
- Server.java

Client.java

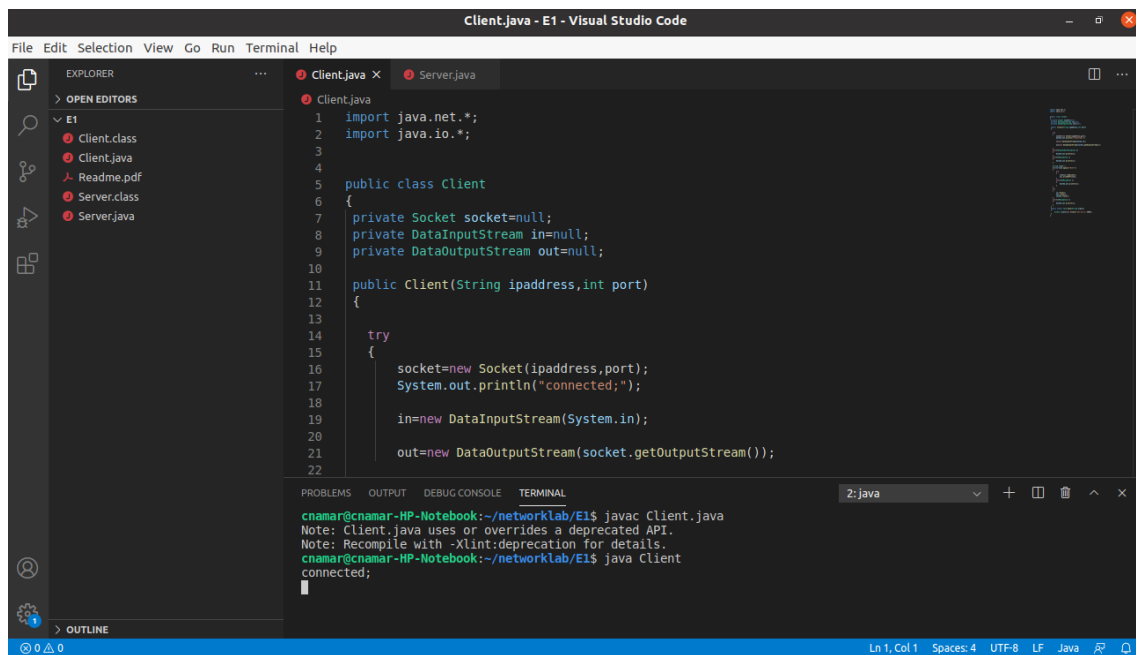
```
1 import java.net.*;
2 import java.io.*;
3
4
5 public class Client
6 {
7     private Socket socket=null;
8     private DataInputStream in=null;
9     private DataOutputStream out=null;
10
11     public Client(String ipaddress,int port)
12     {
13
14         try
15         {
16             socket=new Socket(ipaddress,port);
17             System.out.println("connected;");
18
19             in=new DataInputStream(System.in);
20
21             out=new DataOutputStream(socket.getOutputStream());
22         }
23     }
24 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1:java

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Server.java
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Server
Server started...
Waiting for a client....
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Java



Client.java - E1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

E1

- Client.class
- Client.java
- Readme.pdf
- Server.class
- Server.java

Client.java

```
1 import java.net.*;
2 import java.io.*;
3
4
5 public class Client
6 {
7     private Socket socket=null;
8     private DataInputStream in=null;
9     private DataOutputStream out=null;
10
11     public Client(String ipaddress,int port)
12     {
13
14         try
15         {
16             socket=new Socket(ipaddress,port);
17             System.out.println("connected;");
18
19             in=new DataInputStream(System.in);
20
21             out=new DataOutputStream(socket.getOutputStream());
22         }
23     }
24 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

2:java

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Client
connected;
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Java

The screenshot shows the Visual Studio Code interface with the 'Client.java' file open. The Explorer sidebar on the left shows the project structure with files 'Client.class', 'Client.java', 'Readme.pdf', 'Server.class', and 'Server.java'. The main editor displays the code for 'Client.java', which includes imports for 'java.net.\*' and 'java.io.\*', and a 'Client' class with private fields for 'socket', 'in', and 'out', and a constructor that initializes these fields. The bottom panel shows the 'TERMINAL' tab with the following output:

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Server.java
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Server
Server started...
Waiting for a client....
Accepted
```

The status bar at the bottom indicates 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'LF', 'Java'.

The screenshot shows the Visual Studio Code interface with the 'Client.java' file open. The Explorer sidebar on the left shows the project structure with files 'Client.class', 'Client.java', 'Readme.pdf', 'Server.class', and 'Server.java'. The main editor displays the code for 'Client.java', which includes imports for 'java.net.\*' and 'java.io.\*', and a 'Client' class with private fields for 'socket', 'in', and 'out', and a constructor that initializes these fields. The bottom panel shows the 'TERMINAL' tab with the following output:

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Client
connected;
hello
```

The status bar at the bottom indicates 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'LF', 'Java'.

Client.java - E1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

E1

- Client.class
- Client.java
- Readme.pdf
- Server.class
- Server.java

Client.java

```
1 import java.net.*;
2 import java.io.*;
3
4
5 public class Client
6 {
7     private Socket socket=null;
8     private DataInputStream in=null;
9     private DataOutputStream out=null;
10
11     public Client(String ipaddress,int port)
12     {
13
14         try
15         {
16             socket=new Socket(ipaddress,port);
17             System.out.println("connected");
18
19             in=new DataInputStream(System.in);
20
21             out=new DataOutputStream(socket.getOutputStream());
22         }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1:java

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Server.java
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Server
Server started...
Waiting for a client...
Accepted
message is hello
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Java

Client.java - E1 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

E1

- Client.class
- Client.java
- Readme.pdf
- Server.class
- Server.java

Client.java

```
1 import java.net.*;
2 import java.io.*;
3
4
5 public class Client
6 {
7     private Socket socket=null;
8     private DataInputStream in=null;
9     private DataOutputStream out=null;
10
11     public Client(String ipaddress,int port)
12     {
13
14         try
15         {
16             socket=new Socket(ipaddress,port);
17             System.out.println("connected");
18
19             in=new DataInputStream(System.in);
20
21             out=new DataOutputStream(socket.getOutputStream());
22         }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

2:java

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Client
connected;
hello
hii
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Java

This screenshot shows the Visual Studio Code interface with the 'Client.java' file open. The Explorer sidebar on the left shows the project structure with files 'Client.class', 'Client.java', 'Readme.pdf', 'Server.class', and 'Server.java'. The main editor displays the code for 'Client.java', which includes imports for 'java.net.\*' and 'java.io.\*', and a 'Client' class with private fields for 'socket', 'in', and 'out'. The 'main' method contains a try block that creates a 'Socket' and sets up 'DataInputStream' and 'DataOutputStream' objects. The bottom panel shows the 'TERMINAL' tab with the following output:

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Server.java
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Server
Server started...
Waiting for a client...
Accepted
message is hello
message is hii
```

The status bar at the bottom indicates the current position is 'Ln 1, Col 1' with 'Spaces: 4', 'UTF-8' encoding, and 'LF' line endings.

This screenshot shows the Visual Studio Code interface with the 'Client.java' file open. The Explorer sidebar on the left shows the project structure with files 'Client.class', 'Client.java', 'Readme.pdf', 'Server.class', and 'Server.java'. The main editor displays the code for 'Client.java', which includes imports for 'java.net.\*' and 'java.io.\*', and a 'Client' class with private fields for 'socket', 'in', and 'out'. The 'main' method contains a try block that creates a 'Socket' and sets up 'DataInputStream' and 'DataOutputStream' objects. The bottom panel shows the 'TERMINAL' tab with the following output:

```
cnamar@cnamar-HP-Notebook:~/networklab/E1$ javac Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
cnamar@cnamar-HP-Notebook:~/networklab/E1$ java Client
connected;
hello
hii
Over
cnamar@cnamar-HP-Notebook:~/networklab/E1$
```

The status bar at the bottom indicates the current position is 'Ln 1, Col 1' with 'Spaces: 4', 'UTF-8' encoding, and 'LF' line endings.

