REPORT

Name : G.V.Prashanth Reddy

Roll No : SE20UARI052

TCP Server and Client Implementation :

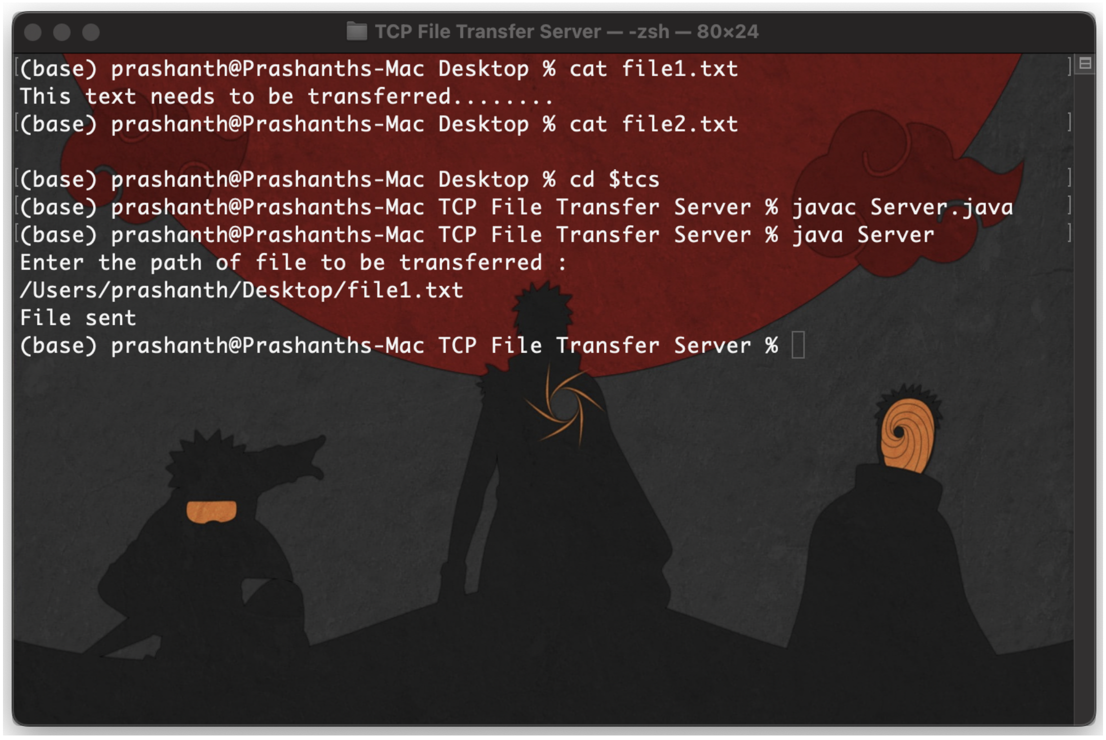
The client and server ends of the connection are identical and both can perform the same operations. The only difference between them is that the output stream of the client is the input stream of the server, and the input stream of the client is the output stream of the server.

1. The server application listens on a local port for requests for connections to be made by a client application.
2. The client application requests a connection from the server port.
3. When the server accepts the request, a port is created on the client computer and is connected to the server port.
4. A socket is created on both ends of the connection, and the details of the connection are encapsulated by the socket.
5. The server port remains available to listen for further connection requests:

Only one server application can exist, but any number of different client processes can connect to the server application.These connections can be in the same computer or in different computers.

Core idea for logic of the program : Saving the content of a file in an array (BufferReader class) and the server sends this to the client, where the client can save this into a file.

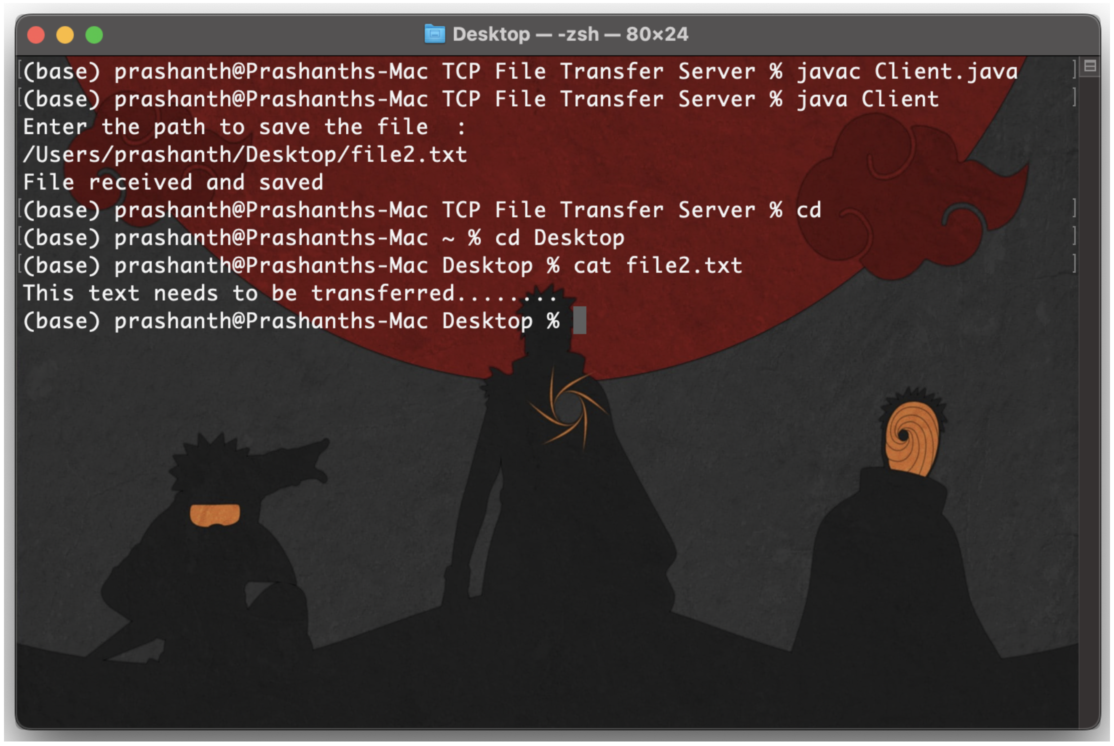
Over Server side :



```
(base) prashanth@Prashanths-Mac Desktop % cat file1.txt
This text needs to be transferred........
(base) prashanth@Prashanths-Mac Desktop % cat file2.txt

(base) prashanth@Prashanths-Mac Desktop % cd $tcs
(base) prashanth@Prashanths-Mac TCP File Transfer Server % javac Server.java
(base) prashanth@Prashanths-Mac TCP File Transfer Server % java Server
Enter the path of file to be transferred :
/Users/prashanth/Desktop/file1.txt
File sent
(base) prashanth@Prashanths-Mac TCP File Transfer Server %
```

Over Client Side :



```
(base) prashanth@Prashanths-Mac TCP File Transfer Server % javac Client.java
(base) prashanth@Prashanths-Mac TCP File Transfer Server % java Client
Enter the path to save the file  :
/Users/prashanth/Desktop/file2.txt
File received and saved
(base) prashanth@Prashanths-Mac TCP File Transfer Server % cd
(base) prashanth@Prashanths-Mac ~ % cd Desktop
(base) prashanth@Prashanths-Mac Desktop % cat file2.txt
This text needs to be transferred........
(base) prashanth@Prashanths-Mac Desktop %
```

TCP Server

1. Import java packages and create class file servers.

```
import java.util.*;

import java.io.FileInputStream;

import java.io.BufferedInputStream;

import java.net.InetAddress;

import java.net.ServerSocket;

import java.net.Socket;

import java.io.File;

import java.io.OutputStream;
```

Java.io , java.util and java.net are Java packages.

The java.io package contains classes for input and output streams.i.e., BufferedReader and DataOutputStream classes (which are used for three streams)

The java.util package is necessary to create an applet that contains all the classes for creating user interfaces.

The java.net package provides classes for network support.i.e., it contains ServerSocket classes.

2. Create a new server socket and bind it to the port.

```
// Initialization of  Sockets

ServerSocket sock = new ServerSocket(5001);

Socket socket = sock.accept();
```

Here ServerSocket is like a WelcomeSocket that is a sort of door that listens for a knock from some client, listens on port number 5001.

Second line creates a new socket which is connectionSocket. TCP establishes a direct virtual pipe between clientSocket at the client and connectionSocket at the server.

With connectionSocket established, the server can continue to listen for requests from other clients for the applications using welcomeSocket.

3. Accept the client connection

```
//The InetAddress specification

InetAddress IA = InetAddress.getByName("localhost");
```

Here both server and client are over same end systems

4. Get the file name and store it into the BufferedReader.

```java
//File Provision

Scanner sc= new Scanner(System.in);

System.out.print("Enter the path of file to be transferred : \n");

String str= sc.nextLine();

File file = new File(str);

FileInputStream fis = new FileInputStream(file);

BufferedInputStream bis = new BufferedInputStream(fis);
```

5. Copy file Contents into array

```java
while(current!=fileLength){
int size = 10000;
if(fileLength - current >= size)
current += size; else
{
size = (int)(fileLength - current);
current = fileLength;
}
contents = new byte[size];
bis.read(contents, 0, size);
os.write(contents);
}
os.flush();
```

6. Close Socket connection

```java
socket.close();
sock.close();
```

TCP Client

1. **Import java packages and create class client**

```
import java.util.*;
import java.io.FileOutputStream;
import java.io.BufferedOutputStream;
import java.net.Socket;
import java.io.InputStream;
import java.net.InetAddress;
```

2. **Initialization of socket and bind it to the port**

```
//Initialization of socket
Socket socket = new Socket(InetAddress.getByName("localhost"), 5001);
byte[] contents = new byte[10000];
```

3. **Initialization of OutPut file**

```
Scanner sc= new Scanner(System.in);
System.out.print("Enter the path to save the file  : \n");
String str= sc.nextLine();
```

4. **Implementation of FileOutputStream. BufferReader class is used for storing data.**

```
//The FileOutputStream to the file's complete path

FileOutputStream fos = new FileOutputStream(str);

BufferedOutputStream bos = new BufferedOutputStream(fos);

InputStream is = socket.getInputStream();
```

5. **Close connection**

```
socket.close();
```