19CSE301 - COMPUTER NETWORKS

Socket Programming

LAB-3 : (03-08-2021)

- R.Abhinav
- CB.EN.U4CSE19453

1. Accepts Client input and prints :
   - TCPServer :

```java
// A Java program for a Server
import java.net.*;
import java.io.*;

public class TCPServer
{
    //initialize socket and input stream
    private Socket      socket = null;
    private ServerSocket server = null;
    private DataInputStream in  = null;

    // constructor with port
    public TCPServer(int port)
    {
        // starts server and waits for a connection
        try
        {
            server = new ServerSocket(port);
            System.out.println("Server started");

            System.out.println("Waiting for a client ...");

            socket = server.accept();
            System.out.println("Client accepted");

            // takes input from the client socket
            in = new DataInputStream(
                new BufferedInputStream(socket.getInputStream()));

            String line = "";

            // reads message from client until "Over" is sent
            while (!line.equals("Over"))
            {
                try
                {
                    line = in.readUTF();
                    System.out.println(line);

                }
                catch(IOException i)
```

```
                {
                    System.out.println(i);
                }
            }
            System.out.println("Closing connection");

            // close connection
            socket.close();
            in.close();
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }

    public static void main(String args[])
    {
        TCPServer server = new TCPServer(5001);
    }
}
```

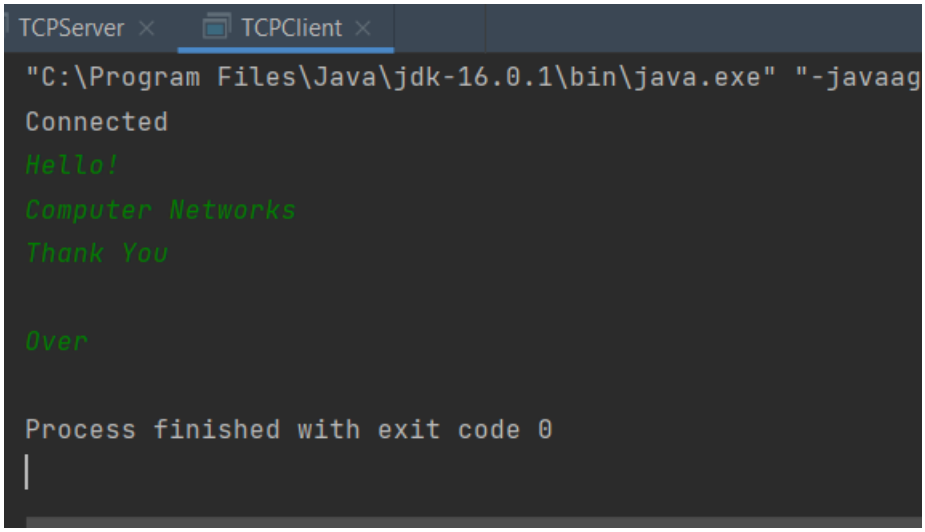
- **TCPClient :**

```
// A Java program for a Client
import java.net.*;
import java.io.*;

public class TCPClient
{
    // initialize socket and input output streams
    private Socket socket      = null;
    private DataInputStream input = null;
    private DataOutputStream out    = null;

    // constructor to put ip address and port
    public TCPClient(String address, int port)
    {
        // establish a connection
        try
        {
            socket = new Socket(address, port);
            System.out.println("Connected");

            // takes input from terminal
            input = new DataInputStream(System.in);

            // sends output to the socket
            out = new DataOutputStream(socket.getOutputStream());
        }
        catch(UnknownHostException u)
        {
            System.out.println(u);
        }
        catch(IOException i)
        {
            System.out.println(i);
```

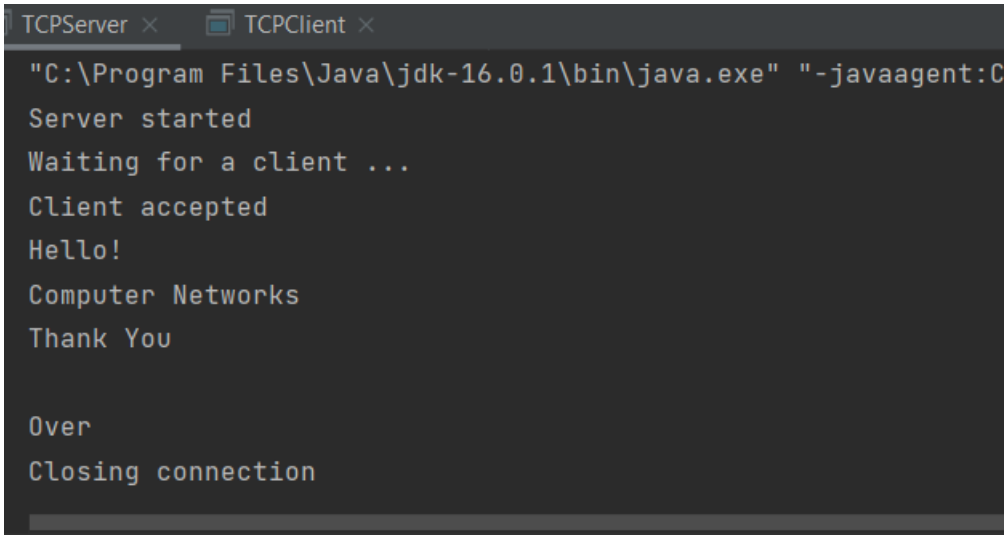
```
        }

        // string to read message from input
        String line = "";

        // keep reading until "Over" is input
        while (!line.equals("Over"))
        {
            try
            {
                line = input.readLine();
                out.writeUTF(line);
            }
            catch(IOException i)
            {
                System.out.println(i);
            }
        }

        // close the connection
        try
        {
            input.close();
            out.close();
            socket.close();
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }

    public static void main(String args[])
    {
        TCPClient client = new TCPClient("127.0.0.1", 5001);
    }
}
```

- **Output :**

  Client:



Server :



2. Echo :

- **Server :**

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

public class TcpEchoServer
{
   public static void main(String[] args)
   {
      int port = 9090;

      try {
          ServerSocket server = new ServerSocket(port);

         while(true) {
            System.out.println("Waiting for clients on port " +
port);
            Socket client = server.accept();

                 System.out.println("Got connection from
"+client.getInetAddress()+":"+client.getPort());

            BufferedReader reader = new BufferedReader(new
InputStreamReader(client.getInputStream()));
            PrintWriter    writer = new
PrintWriter(client.getOutputStream());

            writer.println("Welcome to my server");
            writer.flush();

            String message = reader.readLine();

            while (!(message == null ||
message.equalsIgnoreCase("exit"))) {
               System.out.println("MessageReceived: "+message);
               writer.println(message);
               writer.flush();
```

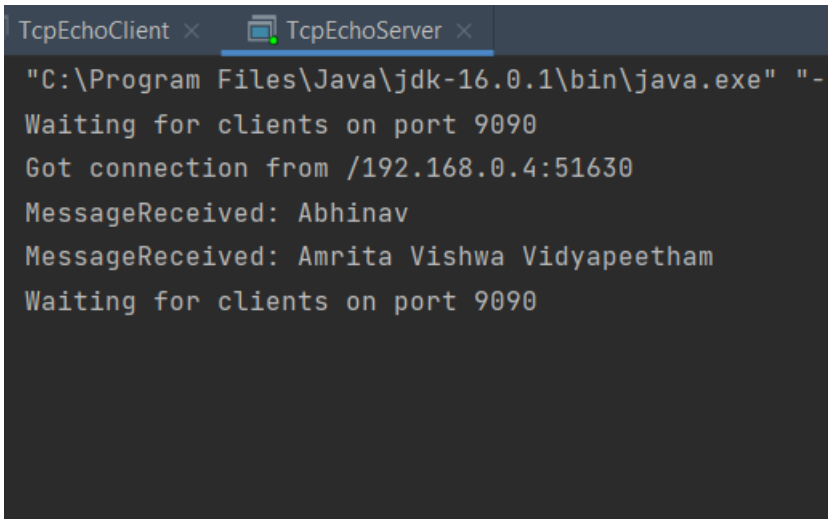
```
                message = reader.readLine();
            }
            client.close();
         }
      } catch(Exception ex) {
         System.out.println("Connection error: "+ex);
      }
   }
}
```

- **Client:**

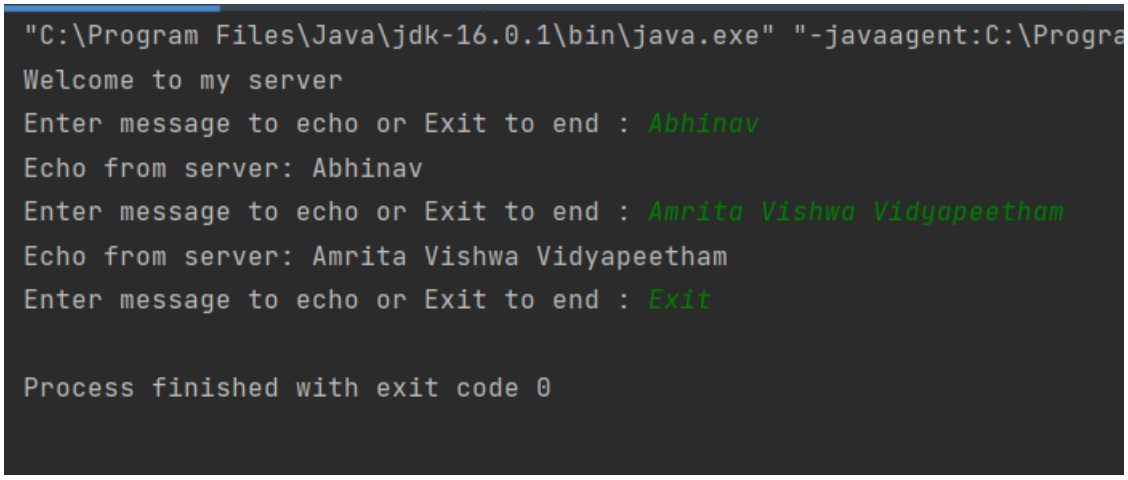
```
import java.net.*;
import java.io.*;
import java.util.*;

public class TcpEchoClient
{
   public static void main(String[] args) {

      int port = 9090;

      try {
         String host = InetAddress.getLocalHost().getHostName();
          Socket client = new Socket(host, port);

         PrintWriter writer = new
PrintWriter(client.getOutputStream());
         BufferedReader reader = new BufferedReader(new
InputStreamReader(client.getInputStream()));

         BufferedReader stdin = new BufferedReader(new
InputStreamReader(System.in));

         System.out.println(reader.readLine()); //read welcome
message
         String message;
         while (true) {
            System.out.print("Enter message to echo or Exit to end
: ");
            message = stdin.readLine();

            if (message == null ||
message.equalsIgnoreCase("exit"))
               break;

            writer.println(message);
            writer.flush();
            System.out.println("Echo from server:
"+reader.readLine());
         }
         client.close();

      }catch (Exception ex) {
         System.out.println("Exception: "+ex);
      }
   }
}
```

- **Output:**

Server:



Client:



3. **File Transfer :**
   - **Server :**

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

public class Server {
    private static final int PORT=1720;
    /**
     *
     * @param fileName The downloaded file save path and file name
     */
    public void downFile(String fileName) {
        try {
            ServerSocket serverSocket = new ServerSocket(PORT);
            Socket socket = serverSocket.accept();
```

```
                System.out.println("Establish a socket link");
                DataInputStream inputStream = new DataInputStream(new
   BufferedInputStream(socket.getInputStream()));
                byte[] buf = new byte[1024];
                DataOutputStream fileOut = new DataOutputStream(new
   BufferedOutputStream(new FileOutputStream(fileName)));
                System.out.println("Start receiving files!" + "\n");
                while ((inputStream.read(buf))!=-1) {
                    fileOut.write(buf, 0,buf.length);
                }
                System.out.println("Received, save the file as" +
   fileName);
                fileOut.close();
            } catch (Exception e) {
                System.out.println("Error receiving message");
                e.printStackTrace();
                return;
            }
        }

        public static void main(String[] args) {
            Server server=new Server();
            server.downFile("src\\copy.txt");
        }

    }
```

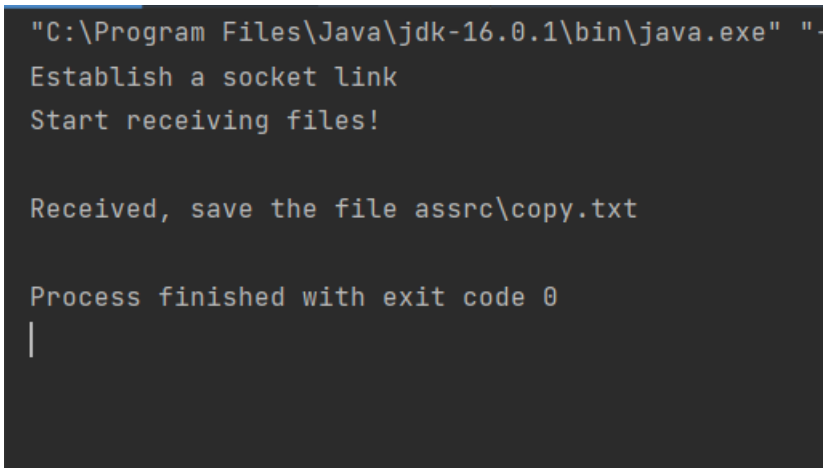
- **Client :**

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


public class Client {
    // Define the address of the local port. Because it is on this
machine, the port and address are defined as static constants
    private static final String HOST = "localhost";
    private static final int PORT = 1720;

    /**
           * @param fileName The path and file name of the uploaded
file
     */
    public void upFile(String fileName) {
        try {
            Socket socket = new Socket(HOST, PORT);
            File file = new File(fileName);
            System.out.println("File length:" + (int) file.length());
            DataInputStream fileIn = new DataInputStream(new
FileInputStream(fileName));
            DataOutputStream out = new
DataOutputStream(socket.getOutputStream());
            byte[] buf = new byte[1024];
            while ((fileIn.read(buf))!=-1) {
                out.write(buf, 0, buf.length);
            }
            out.flush();
```

```
            fileIn.close();
            out.close();
            socket.close();
            System.out.println("File transfer completed");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        Client client = new Client();
        client.upFile("C:\\first.txt");
    }
}
```

- **Output :**

Server :



Client :