# WAP to build a one way basic setup for interaction between the server and the client (socket programming)

## Server Side

import java.net.\*;

import java.io.\*;

public class Server {

private Socket socket = null;

private ServerSocket server = null;

private BufferedReader in = null;

public Server(int port) {

try {

server = new ServerSocket(port);

System.out.println("Server ready");

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

socket = server.accept();

System.out.println("Client accepted");

// takes input from the client socket

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

String line = in.readLine();

System.out.println("Client data: " + line);

// close connection

System.out.println("Closing connection");

in.close();

socket.close();

} catch(IOException i) {

System.out.println(i);

}

}

// Driver Method

public static void main(String args[]) {

Server server = new Server(5500);

}

}

## Client Side

import java.net.\*;

import java.io.\*;

public class Client {

  private Socket socket    = null;

  private DataOutputStream out   = null;

  public Client(String address, int port) {

    try {

      socket = new Socket(address, port);

      out = new DataOutputStream(socket.getOutputStream());

      out.writeBytes("I am connected");

    } catch(UnknownHostException u) {

      System.out.println(u);

    } catch(IOException i) {

      System.out.println(i);

    }

    // Close connection

    try {

      out.close();

      socket.close();

    } catch(IOException i) {

      System.out.println(i);

    }

  }

  // Driver Method

  public static void main(String args[]) {

    Client client = new Client("localhost", 5500);

  }

}

## On running [Screenshot of program execution]

