**Create a folder ( chatapp) in your machine and create Server Class.**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

public class Server {

Socket socket;

ServerSocket server;

BufferedReader br;

PrintWriter out;

public Server() {

try {

server = new ServerSocket(8012);

System.out.println("Server is ready to accept the connection!");

System.out.println("Waiting!");

socket = server.accept();

//Read the data from one socket end

br = new BufferedReader(

new InputStreamReader(socket.getInputStream()));

//Write data

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

StartReading();

StartWriting();

}

catch(Exception ex) {

ex.printStackTrace();

}

}

public void StartReading() {

System.out.println("Server - Reader starting");

Runnable read = () -> {

try {

while(true) {

String msg = br.readLine();

if(msg.equals("exit")) {

System.out.println("Client terminated !");

socket.close();

break;

}

System.out.println("Client :" + msg);

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println("Connection is closed");

}

};

new Thread(read).start();

}

public void StartWriting() {

System.out.println("Server - Writer starting");

Runnable write = () -> {

try {

while(true && !socket.isClosed()) {

BufferedReader bris = new BufferedReader(new

InputStreamReader(System.in));

String recdata = bris.readLine();

out.println(recdata);

out.flush();

if(recdata.equals("exit")) {

socket.close();

break;

}

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println("Connection is closed");

}

};

new Thread(write).start();

}

public static void main(String [] args) {

System.out.println("Server Start!");

new Server();

}

}

**Create a folder ( chatapp) in your machine and create Client Class.**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

public class Client {

Socket socket;

BufferedReader br;

PrintWriter out;

public Client() {

try {

System.out.println("Sending request to the server!");

socket = new Socket("127.0.0.1",8012);

System.out.println("Connection done!");

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

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

StartReading();

StartWriting();

} catch(Exception ex) {

ex.printStackTrace();

}

}

public void StartReading() {

System.out.println("Client - Reader starting");

Runnable read = () -> {

try {

while(true) {

String msg = br.readLine();

if(msg.equals("exit")) {

System.out.println("Server terminated the chat!");

socket.close();

break;

}

System.out.println("Server :" + msg);

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println("Connection is closed");

}

};

new Thread(read).start();

}

public void StartWriting() {

System.out.println("Client - Reader starting");

Runnable write = () -> {

try {

while(true && !socket.isClosed()){

BufferedReader bris = new BufferedReader(new

InputStreamReader(System.in));

String recdata = bris.readLine();

out.println(recdata);

out.flush();

if(recdata.equals("exit")) {

socket.close();

break;

}

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println("Connection is closed");

}

};

new Thread(write).start();

}

public static void main(String[] args) {

System.out.println("Client start!");

new Client();

}

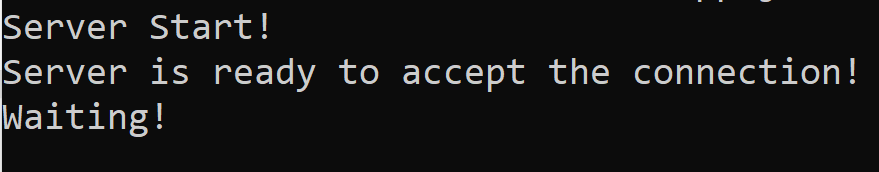
}

**Compile the both classes**

**chatapp>javac Server.java**

**chatapp>javac Client.java**

**chatapp>java Server**



**chatapp>java Client**

