DAY 2——实现双机通讯

问题 2：如何利用以太网进行双机通讯？

1.1 学习内容

双机通讯，请学习教程的第 1 章 ECHO 通讯。

2.2 任务清单

① 阅读服务器和客户端程序，划出关键语句。

② 自己实现程序，观察运行结果。

③ 深入思考每条语句的作用。

2.3 任务分解

① 详细写出 main 函数的执行步骤，用序号表示每一步执行过程。

② 特别注意并记录程序的善后工作步骤。

2.4 关键技术

2.4.1 输入/输出流

```
 in=new BufferedReader(new
InputStreamReader(toClientSocket.getInputStream(),"UTF-8"));
 out=new PrintWriter(new
OutputStreamWriter(toClientSocket.getOutputStream(),"UTF-8"),true);
```

2.4.2 服务器端关键代码

1、SocketAddress 套接字地址类

SocketAddress serverAddr=new InetSocketAddress("localhost",5000);

2、ServerSocket 服务器监听类

listenSocket = new ServerSocket();

listenSocket.bind(serverAddr) ;//绑定本地端口

3、Socket 套接字通信类

Socket clientSocket = null;

clientSocket = listenSocket.accept();

完整示例:

```
        listenSocket = new ServerSocket();

        SocketAddress serverAddr=new
InetSocketAddress("localhost",5000);

        listenSocket.bind(serverAddr);

        System.out.println("1.服务器启动成功！开始在 5000 端口侦听连
接...");

        //2.处理连接

        clientSocket = listenSocket.accept();

        System.out.println("2.客户机连接成功！客户机地址和端口：
"+clientSocket.getRemoteSocketAddress());
```

4、创建会话流

```
        //服务器与客户机会话

        in = new BufferedReader(

            new InputStreamReader(
```

```
            clientSocket.getInputStream()));

        out = new BufferedWriter(

            new OutputStreamWriter(

            clientSocket.getOutputStream()));
```

5、读写信息

```
        String recvStr=in.readLine();  //从客户机接收字符串，读

        System.out.println("3.1 服务器收到字符串："+recvStr);

        out.write(recvStr);  //向客户机回送字符串

        out.newLine();

        out.flush();
```

6、关闭

```
        //关闭套接字和流

        try {

            if (in != null)   in.close();

            if (out != null)  out.close();

            if (listenSocket != null)   listenSocket.close();

            if (clientSocket != null)   clientSocket.close();

            System.out.println("4.关闭套接字和流成功！");

        } catch (IOException ex) {

            System.out.println("异常信息"+ex.getMessage());

        }
```

2.4.3 客户端关键代码

1、SocketAddress 套接字地址类

```
SocketAddress remoteAddr=new InetSocketAddress("localhost",5000);
```

2、Socket 套接字通信类

```java
Socket clientSocket = null;

clientSocket = new Socket();

clientSocket.connect(remoteAddr);
```

3、创建会话流

```java
in = new BufferedReader(

    new InputStreamReader(

    clientSocket.getInputStream()));

out = new BufferedWriter(

    new OutputStreamWriter(

    clientSocket.getOutputStream()));
```

4、读写信息

```java
out.write(sendStr);   //向服务器发送字符串

out.newLine();

out.flush();

System.out.println("3.1向服务器发送字符串成功!"+sendStr);

String recvStr=in.readLine();   //从服务器接收字符串

System.out.println("3.2从服务器接收回送字符串成功！"+recvStr);
```

5、完整示例

```java
Socket clientSocket = null;

    BufferedReader in = null;

    BufferedWriter out = null;

    try {

        //1.创建客户机套接字

        clientSocket = new Socket();

        SocketAddress remoteAddr=new

InetSocketAddress("localhost",5000);

        System.out.println("1.创建客户机套接字成功!");

        //2.连接服务器
```

```
            clientSocket.connect(remoteAddr);

            System.out.println("2.客户机连接服务器 localhost 端口 5000 成功！
");

            System.out.println("客户机的地址和端口：
"+clientSocket.getLocalSocketAddress());

                //与服务器会话

                in = new BufferedReader(

                    new InputStreamReader(

                    clientSocket.getInputStream()));

                out = new BufferedWriter(

                    new OutputStreamWriter(

                    clientSocket.getOutputStream()));

                String sendStr="有朋自远方来";

                out.write(sendStr);  //向服务器发送字符串

                out.newLine();

                out.flush();

                System.out.println("3.1 向服务器发送字符串成功！"+sendStr);

                String recvStr=in.readLine();  //从服务器接收字符串

                System.out.println("3.2 从服务器接收回送字符串成功！"+recvStr);

        } catch (IOException ex) {

            System.out.println("异常信息："+ex.getMessage());

        }

        //关闭套接字和流

        try {

            if (in != null) in.close();

            if (out != null)  out.close();

            if (clientSocket != null)   clientSocket.close();

            System.out.println("4.关闭套接字和流成功！");

        } catch (IOException ex) {

             System.out.println("异常信息："+ex.getMessage());
```

```
            }
        }
```

2.5 问题讨论

① in.readline()返回空或不返回，分别代表什么？

② Out.flush()应在什么时候使用？

③ 如何在二台计算机上进行实验?如何知道对方的 IP ？

④ 程序如何改进？