

JavaSocket编程（二）PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/138/2021_2022_JavaSocket_c97_138762.htm 在Java中面向连接的类有两种形式,它们分别是客户端和服务端.客户端这一部分是最简单的,所以我们先讨论它.列表9.1列出了一个简单的客户端的程序.它向一个服务器发出一个请求,取回一个HTML文档,并把它显示在控制台上.

```
9.1 一个简单的socket客户端
import java.io.*; import java.net.*; /**
 * 一个简单的从服务器取回一个HTML页面的程序 * 注
意:merlin是本地机器的名字 */ public class SimpleWebClient {
public static void main(String args[]) { try { // 打开一个客户端socket连接
Socket clientSocket1 = new Socket("merlin", 80).
System.out.println("Client1: \" clientSocket1). // 取得一个网页
getPage(clientSocket1). } catch (UnknownHostException uhe) {
System.out.println("UnknownHostException: \" uhe). } catch
(IOException ioe) { System.err.println("IOException: \" ioe). } }
/** *通过建立的连接请求一个页面,显示回应然后关闭socket */
public static void getPage(Socket clientSocket) { try { // 需要输入和
输出流 DataOutputStream outbound = new DataOutputStream(
clientSocket.getOutputStream() ). DataInputStream inbound = new
DataInputStream( clientSocket.getInputStream() ). // 向服务器发
出HTTP请求 outbound.writeBytes("GET / HTTP/1.0 "). // 读出
回应 String responseLine. while ((responseLine =
inbound.readLine()) != null) { // 把每一行显示出来
System.out.println(responseLine). if ( responseLine.indexOf("\\") !=
-1 ) break. } // 清除 outbound.close(). inbound.close().
```

```
clientSocket.close(). } catch (IOException ioe) {
System.out.println("\nIOException: \n" ioe). } } }
```

Java面向连接的类回忆一个,一个客户端向一个正在监听的服务器socket发出一个连接.客户端的sockets是用Socket类建立的.下面的程序建立了一个客户端的socket并且连接到了一个主机: Socket

```
clientSocket = new Socket("\merlin", 80).
```

第一个参数是你想要连接的主机的名称,第二个参数是端口号.一个主机名称指定了目的的名称.端口号指定了由哪个应用程序来接收.在我们的情况下,必须指定80,因为它是默认的HTTP协议的端口.另外的知名的端口列在表9.1中,看: 知名的端口: echo 7 daytime 13 daytime 13 ftp 21 telnet 23 smtp 25 finger 79 http 80 pop3 110 因为Socket类是面向连接的,它提供了一个可供读写的流接口.java.io包中的类可以用来访问一个已连接的socket: DataOutputStream

```
outbound = new DataOutputStream(
clientSocket.getOutputStream() ). DataInputStream inbound = new
DataInputStream( clientSocket.getInputStream() ). 一旦流建立了,
一般的流操作就可以做了: outbound.writeBytes("\GET /
HTTP/1.0 ). String responseLine. while ( (responseLine =
inbound.readLine()) != null) { System.out.println(responseLine). }
```

以上的小程序请求了一个WEB页面并且把它显示出来.当程序完成之后,连接必须关闭. outbound.close(). inbound.close(). clientSocket.close(). 注意socket流必须首先关闭.所有的的socket流必须在socket关闭之前关闭.这个小程序非常地简单,但是所有的客户端程序都必须遵首下面的基本的步骤: 1.建立客户端socket连接. 2.得到socket的读和写的流. 3.利用流. 4.关闭流. 5.关闭socket. 使用一个服务器端的socket只是有一点复杂,它将在

下面讲到. 100Test 下载频道开通，各类考试题目直接下载。
详细请访问 www.100test.com