

java认证:JavaSocket编程的一个秘密类Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/559/2021_2022_java_E8_AE_A4_E8_AF_81_c104_559556.htm 介绍 Java平台在java.net包里来实现Socket。在这本文中，我们将使用Java.net包中的下面三个类来工作：URLConnection Socket ServerSocket 在java.net包里包含有更多的类，但是这些是你最经常遇见的，让我们从URLConnection开始，这个类提供了在你的java代码里使用Socket的方法而无需了解Socket的底层机制。甚至不用尝试就可以使用sockets 连接到一个URL包括以下几个步骤：创建一个URLConnection 用不同的setter方法配置它 连接到URLConnection 与不同的getter方法进行交互 下面，我们来用一些例子示范怎样使用URLConnection从一台服务器上请求一份文档。 URLClient类 我们将从URLClient类的结构开始讲起。

```
import java.io.*; import java.net.*; public class URLClient {
protected URLConnection connection; public static void
main(String[] args) {} public String getDocumentAt(String
urlString) {} }
```

注意：必须先导入java.net和java.io包才行 我们给我们的类一个实例变量用于保存一个URLConnection 我们的类包含一个main()方法用于处理浏览一个文档的逻辑流（logic flow），我们的类还包含了getDocumentAt()方法用于连接服务器以及请求文档，下面我们将探究这些方法的细节。

浏览文档 main()方法用于处理浏览一个文档的逻辑流（logic flow）：

```
public static void main(String[] args) { URLClient
client = new URLClient(); String 100test =
client.getDocumentAt("http://www.100test.com");
```

System.out.println(100test). } 我们的main()方法仅仅创建了一个新的URLConnection类的实例并使用一个有效的URL String来调用getDocumentAt()方法。当调用返回文档，我们把它储存在一个String里并把这个String输出到控制台上。然而，实际的工作是getDocumentAt()方法当中完成的。从服务器上请求一份文档 getDocumentAt()方法处理在实际工作中如何从web上得到一份文档：

```
public String getDocumentAt(String urlString) {
    StringBuffer document = new StringBuffer(). try { URL url = new
    URL(urlString). URLConnection conn = url.openConnection().
    BufferedReader reader = new BufferedReader(new
    InputStreamReader(conn.getInputStream())). String line = null.
    while ((line = reader.readLine()) != null) document.append(line
    "\n"). reader.close(). } catch (MalformedURLException e) {
    System.out.println("Unable to connect to URL: " urlString). } catch
    (IOException e) { System.out.println("IOException when
    connecting to URL: " urlString). } return document.toString(). }
```

getDocumentAt()方法有一个String类型的参数包含我们想得到的那份文档的URL。我们先创建一个StringBuffer用于保存文档的行。接着，我们用传进去的参数urlString来创建一个新的URL。然后，我们创建一个URLConnection并打开它：

```
URLConnection conn = url.openConnection().
```

一旦有了一个URLConnection，我们就获得它的InputStream并包装成InputStreamReader，然后我们又把它进而包装成BufferedReader以至于我们能够读取从服务器获得的文档的行，我们在java代码中处理socket的时候会经常使用这种包装技术。在我们继续学习之前你必须熟悉它：BufferedReader

reader = new BufferedReader(new
InputStreamReader(conn.getInputStream())). 有了BufferedReader
，我们能够容易的读取文档的内容。我们在一个while...loop循
环里调用reader上的readLine()方法：String line = null. while
((line = reader.readLine()) != null) document.append(line "\n"). 调
用readLine()方法后从InputStream传入行终止符（例如换行符
）时才产生阻塞。如果没有得到，它将继续等待，当连接关
闭时它才会返回null，既然这样，一旦我们获得一个行，我们
连同一个换行符把它追加到一个调用的文档的StringBuffer上
。这样就保留了从服务器上原文档的格式。当我们读取所有
行以后，我们应该关闭BufferedReader: reader.close(). 如果提供
给urlString的URL构造器无效，则将会抛出一个MalformedUR
特拉LException异常。同样如果产生了其他的错误，例如从连
接获取InputStream时，将会抛出IOException。 总结 1 . 用一
个你想连接的资源的有效url String来实例化URL 2 . 连接到
指定URL 3 . 包装InputStream为连接在BufferedReader以至于你
可以读取行 4 . 用你的BufferedReader读取文档内容 5 . 关
闭BufferedReader 更多优质资料尽在百考试题论坛 百考试题在
线题库 java认证更多详细资料 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