

## JAVA编写的一个简单的Socket实现的HTTP响应服务器

2013年09月24日 15:45:59 阅读数：6932

---

JAVA编写的一个简单的Socket实现的HTTP响应服务器，看后就很容易理解Web服务器的原理了。

[java]  

```
1. package test.io;
2.
3. import java.net.*;
4. import java.io.*;
5.
6. /**
7.  * 一个简单的Socket实现的HTTP响应服务器。<br>
8.  * 只用于熟悉HTTP协议的目的，可以看到浏览器发过来的数据格式。
9.  *
10.  * @author */
11. public class MyWebServer {
12.     public static void main(String[] args) {
13.         Socket socket = null;
14.         try {
15.             // 创建一个监听8000端口的服务器Socket
16.             ServerSocket s = new ServerSocket(8000, 3);
17.             System.out.println("MyWebServer等待来自浏览器的连接\n");
18.             while (true) {
19.                 socket = s.accept();
20.                 System.out.println("连接已建立。端口号：" + socket.getPort());
21.                 new MyWebServerThread(socket).start();
22.             }
23.         } catch (IOException e) {
24.             e.printStackTrace();
25.         }
26.     }
27. }
28.
29. class MyWebServerThread extends Thread {
30.     private Socket socket;
31.
32.     MyWebServerThread(Socket socket) {
33.         this.socket = socket;
34.     }
35.
36.     @Override
37.     public void run() {
38.         try {
39.             InputStreamReader is = new InputStreamReader(socket.getInputStream());
40.             char[] bs = new char[2048];
41.             PrintStream out;
42.             out = new PrintStream(socket.getOutputStream());
43.             StringBuilder msg = new StringBuilder();
44.             // 如果10毫秒还没有数据，则视同没有新的数据了。
45.             // 因为有Keep-Alive的缘故，浏览器可能不主动断开连接的。
46.             // 实际应用，会根据协议第一行是GET还是 POST确定。
47.             socket.setSoTimeout(10);
48.             //
49.             // 此处读入请求数据并做相应的处理
50.             //
51.             int len = -1;
52.             try {
53.                 while ((len = is.read(bs)) != -1) {
54.                     msg.append(bs, 0, len);
55.                     msg.append("\n");
56.                 }
57.             } catch (Exception ex) {
58.                 // ex.printStackTrace();
59.             }
60.             // 下面是由服务器直接生成的主页内容
61.             // 1、首先向浏览器输出响应头信息
62.             out.println("HTTP/1.1 200 OK");
63.             out.println("Content-Type:text/html;charset:GBK");
64.             out.println();
65.             // 2、输出主页信息
66.             out
67.                 .println("<HTML><BODY>"
68.                     + "<center>"
69.                     + "<H1>HTTP协议测试服务器,当前时间："
70.                     + new java.util.Date()
71.                     + "</H1>"
72.                     + "<form method='get'>username:<input type='text' name='username'/>password:<input type='text' name='password'/><input"
73.                     + "<form method='post'>username:<input type='text' name='username'/>password:<input type='text' name='password'/><input"
74.                     + "<form method='post' enctype='multipart/form-data'>phototitle:<input type='text' name='phototitle'/>photo:<input ty"
75.                     + "</center>您提交的数据如下:<pre>" + msg.toString() + "</pre></BODY></HTML>");
76.             out.flush();
77.             out.close();
78.             is.close();
79.             System.out.println("close");
80.             // 关闭连接
81.             socket.close();
82.         } catch (IOException e) {
83.             e.printStackTrace();
84.         }
85.     }
86. }
```

文章标签：[JAVA](#) [Socket](#) [HTTP](#) [服务器](#)

个人分类：[J2EE](#) [计算机网络](#)

---

此PDF由[spygg](#)生成,请尊重原作者版权!!!

我的邮箱:[liushidc@163.com](mailto:liushidc@163.com)