深入剖析 Tomcat

A Fool

2019

目录

第一章　一个简单的 Web 服务器

1.1　HTTP

超文本传输协议（HyperText Transfer Protocol）是基于“请求-响应”的协议。

HTTP 使用可靠地 TCP 连接，默认使用 TCP 80 端口。

Web 服务器不负责联系客户端或建立到一个客户端的回调连接，客户端或服务端可提前关闭连接。

1.1.1　HTTP 请求

一个 HTTP 请求包含：

- 请求方法——统一资源标识符（Uniform Resource Identifier，URI）——协议/版本
- 请求头
- 实体

HTTP1.1 支持 7 种请求方法：GET、POST、HEAD、OPTIONS、PUT、DELETE 和 TRACE。统一资源定位符（Uniform Resource Locator，URL）实际上是 URI 的一种类型。

请求头包含了客户端环境、请求实体正文的相关信息。各个请求头之间使用回车/换行（Carriage Return/LineFeed，CRLF）间隔开。

在请求头和请求实体正文之间有一个空行，该行只有 CRLF 符。

1.1.2　HTTP 响应

HTTP 响应三部分：

- 协议——状态码——描述
- 响应头
- 响应式体段

响应头和响应实体正文之间只包含了 CRLF 的一个空行分隔。

1.2 Socket 类

套接字使应用程序可以从网络中读取数据，可以向网络中写入数据。

Java 中的套接字：java.net.Socket，构造函数：

public Socket(java.lang.String host, int port)

host 为远程主机名称或 IP 地址，参数 port 是远程应用程序端口号。两者通信：

```
1  Socket socket = new Socket("127.0.0.1", 8080);
2  OutputStream os = socket.getOutputStream();
3  boolean autoflush = true;
4  PrintWriter out = new PrintWriter(socket.getOutputStream(),
       autoflush);
5  BufferedReader in = new BufferedReader(new InputStreamReader(socket
       .getInputStream()));
6
7  out.println("GET /index.jsp HTTP/1.1");
8  out.println("Host: localhost:8080");
9  out.println("Connection: Close");
10 out.println();
11
12 boolean loop =true;
13 StringBuffer sb = new StringBuffer(8096);
14 while (loop) {
15     if (in.ready()) {
16         int i = 0;
17         while (i != -1) {
18             i = in.read();
19             sb.append((char) i);
20         }
21         loop = false;
22     }
23     Thread.currentThread().sleep(50);
24 }
25 System.out.println(sb.toString());
26 socket.close();
```