异步图书
www.epubit.com.cn

畅销书《Spring MVC 学习指南》姐妹篇

Servlet、JSP 和 Spring MVC 初学指南

[加] Budi Kurniawan [美] Paul Deck 著
林仪明 俞黎敏 译

中国工信出版集团 人民邮电出版社 POSTS & TELECOM PRESS

Servlet、JSP和Spring MVC

初学指南

[加] Budi Kurniawan [美] Paul Deck 著
林仪明 俞黎敏 译

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

Servlet、JSP和Spring MVC初学指南 / （加）克尼亚万（Budi Kurniawan），（美）戴克（Paul Deck）著；林仪明，俞黎敏译. -- 北京 : 人民邮电出版社，2016.11
ISBN 978-7-115-42974-2

Ⅰ. ①S… Ⅱ. ①克… ②戴… ③林… ④俞… Ⅲ. ①JAVA语言－程序设计－指南 Ⅳ. ①TP312-62

中国版本图书馆CIP数据核字(2016)第157873号

版权声明

◆ 著　[加] Budi Kurniawan　[美] Paul Deck
译　林仪明　俞黎敏
责任编辑　陈冀康
责任印制　焦志炜

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路 11 号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：24.75
字数：553 千字　2016 年 11 月第 1 版
印数：1－3 000 册　2016 年 11 月北京第 1 次印刷

著作权合同登记号　图字：01-2015-8292 号

定价：69.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前言

Java Servlet 技术简称 Servlet 技术，是 Java 开发 Web 应用的底层技术。由 Sun 公司于 1996 年发布，用来代替 CGI——当时生成 Web 动态内容的主流技术。CGI 技术的主要问题是每个 Web 请求都需要新启动一个进程来处理。创建进程会消耗不少 CPU 周期，导致难以编写可扩展的 CGI 程序。而 Servlet 有着比 CGI 程序更好的性能，因为 Servlet 在创建后（处理第一个请求时）就一直保持在内存中。此后，SUN 公司发布了 JavaServer Pages（JSP）技术，以进一步简化 servlet 程序开发。

自从 Servlet 和 JSP 技术诞生后，涌现出大量的基于 Java 的 Web 框架来帮助开发人员快速编写 Web 应用。这些框架构建于 Servlet 和 JSP 之上，帮助开发人员更加关注业务逻辑，无须编写重复性（技术）代码。目前，Spring MVC 是最为流行的可扩展 Java Web 应用开发框架。

Spring MVC 又叫 Spring Web MVC，是 Spring 框架的一个模块，用于快速开发 Web 应用。MVC 代表 Model-View-Controller，是一个广泛应用于 GUI 开发的设计模式。该模式不局限于 Web 开发，也广泛应用在桌面开发技术上，如 Java Swing 和 JavaFX。

下面将简要介绍 HTTP、基于 Servlet 和 JSP 的 Web 编程，以及本书的章节内容编排。

注意

本书中所有示例代码基于 Servlet 3.0、JSP 2.3 以及 Spring MVC 4。本书假定读者已有 Java 以及面向对象编程基础。对于 Java 新手，我们建议阅读由 Budi Kurniawan 编写的《Java : A Beginner's Tutorial (Fourth Edition)》(ISBN 9780992133047)一书。

Servlet/JSP 应用架构

Servlet 是一个 Java 程序，一个 Servlet 应用有一个或多个 Servlet 程序。JSP 页面会被转换和编译成 Servlet 程序。

Servlet 应用无法独立运行，必须运行在 Servlet 容器中。Servlet 容器将用户的请求传递给 Servlet 应用，并将结果返回给用户。由于大部分 Servlet 应用都包含多个 JSP 页面，因此更准确地说是“Servlet/JSP 应用”。

Web 用户通过 Web 浏览器例如 IE、Mozilla Firefox 或者谷歌 Chrome 来访问 Servlet 应用。通常，Web 浏览器又叫 Web 客户端。

图 I.1 展示了 Servlet/JSP 应用的架构。

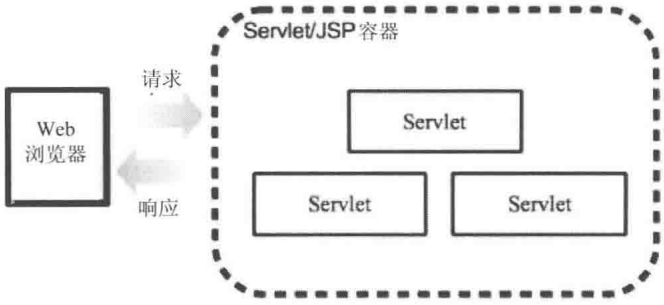

图 I.1 Servlet/JSP 应用架构

Web 服务器和 Web 客户端间通过 HTTP 协议通信，因此 Web 服务器也叫 HTTP 服务器。下面会详细讨论 HTTP 协议。

Servlet/JSP 容器是一个可以同时处理 Servlet 和静态内容的 Web 容器。过去，由于通常认为 HTTP 服务器比 Servlet/JSP 容器更加可靠，因此人们习惯将 Servlet/JSP 容器作为 HTTP 服务器如 Apache HTTP 服务器的一个模块。这种模式下，HTTP 服务器用来处理静态资源，而 Servlet/JSP 容器则负责生成动态内容。如今，Servlet/JSP 容器更加成熟可靠，并被广泛地独立部署。Apache Tomcat 和 Jetty 是当前最流行的 Servlet/JSP 容器，并且它们是免费而且开源的。你可以访问 http://tomcat.apache.org 以及 http://www.eclipse.org/jetty 下载。

Servlet 和 JSP 只是 Java 企业版中众多技术中的两个，其他 Java EE 技术还有 Java 消息服务，企业 Java 对象、JavaServer Faces 以及 Java 持久化等，完整的 Java EE 技术列表可以访问如下地址：

```
http://www.oracle.com/technetwork/java/javaee/tech/index.html
```

要运行 Java EE 应用，需要一个 Java EE 容器，例如 GlassFish、JBoss、Oracle Weblogic 或者 IBM WebSphere。诚然，我们可以将一个 Servlet/JSP 应用部署到一个 Java EE 容器上，但一个 Servlet/JSP 容器就已经满足需要了，并且更加轻量。当然，Tomcat 和 Jetty 不是 Java EE 容器，因此无法运行 EJB 或 JMS 技术。

HTTP

HTTP 协议使得 Web 服务器与浏览器之间可以通过互联网或内网进行数据交互。万维网联盟（W3C），作为一个制定标准的国际社区，负责和维护 HTTP 协议。HTTP 第一版是 0.9，之后是 HTTP 1.0，当前最新版本是 HTTP 1.1。HTTP 1.1 版本的 RFC 编号是 2616，下载地址为 http://www.w3.org/Protocols/HTTP/1.1/rfc2616.pdf。按计划，HTTP 的下一个版本是 HTTP/2。

Web 服务器 7×24 小时不间断运行，并等待 HTTP 客户端（通常是 Web 浏览器）来连接

并请求资源。通常，由客户端发起一个连接，服务端不会主动连接客户端。

注意：

2011 年，标准化组织 IETF 发布了 WebSocket 协议，即 RFC 6455 规范。该协议允许一个 HTTP 连接升级为 WebSocket 连接，支持双向通信，这就使得服务端可以通过 WebSocket 协议主动发起同客户端的会话通信。

互联网用户需要通过点击或者输入一个 URL 链接或地址来访问一个资源，如下为两个示例：

```
http://google.com/index.html
http://facebook.com/index.html
```

URL 的第一个部分是 http，代表所采用的协议。除 HTTP 协议外，URL 还可以采用其他类型的协议，如下为两个示例：

```
mailto:joe@example.com
ftp://marketing@ftp.example.org
```

通常，HTTP 的 URL 格式如下：

```
protocol://[host.]domain[:port][/context][/resource][?query string]
```

或者

```
protocol://IP address[:port][/context][/resource][?query string]
```

中括号中的内容是可选的，因此一个最简的 URL 是 http://yahoo.ca 或者 http://192.168.1.9。

需要说明的是，除了输入 http://google.com，你还可以用 http://209.85.143.99 来访问谷歌。可以用 ping 命令来获取域名所对应的 IP 地址：

```
ping google.com
```

由于 IP 地址不容易记忆，实践中更倾向于使用域名。一台计算机可以托管不止一个域名，因此不同的域名可能指向同一个 IP。另外，example.com 或者 example.org 无法被注册，因为它们被保留作为各类文档手册举例使用。

URL 中的 Host 部分用来表示在互联网或内网中一个唯一的地址，例如：http://yahoo.com（没有 host）所访问的地址完全不同于 http://mail.yahoo.com（有 host）。多年以来，作为最受欢迎的主机名，www 是默认的主机名，通常，http://www.domainName 会被映射到 http://domainName。

HTTP 的默认端口是 80 端口。因此，对于采用 80 端口的 Web 服务器，可以无须输入端口号。但有时候，Web 服务器并未运行在 80 端口上，此时必须输入相应的端口号。例如：Tomcat 服务器的默认端口号是 8080，为了能正确访问，必须提供输入端口号：

```
http://localhost:8080
```

localhost 作为一个保留关键字，用于指向本机。

URL 中的 context 部分用来代表应用名称，该部分也是可选的。一台 Web 服务器可以运行多个上下文（应用），其中一个可以配置为默认上下文，对于访问默认上下文中的资源，可以跳过 context 部分。

最后，一个 context 可以有一个或多个默认资源（通常为 index.html，index.htm 或者 default.htm）。一个没有带资源名称的 URL 通常指向默认资源。当存在多个默认资源时，其中最高优先级的资源将被返回给客户端。

在资源名之后可以有一个或多个查询语句或者路径参数。查询语句是一个 Key/Value 组，多个查询语句间用“&”符号分隔。路径参数类似于查询语句，但只有 value 部分，多个 value 部分用“/”符号分隔。

HTTP 请求

一个 HTTP 请求包含三部分内容：

- 方法-URI-协议/版本
- 请求头信息
- 请求正文

如下为一个具体示例：

```
POST /examples/default.jsp HTTP/1.1
Accept: text/plain; text/html
Accept-Language:en-gb
Connection:Keep-Alive
Host:localhost
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36
➥(KHTML, like Gecko) Chrome/37.0.2049.0 Safari/537.36
Content-Length:30
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate

lastName=Blanks&firstName=Mike
```

请求的第一行即是：方法-URI-协议/版本

```
POST /examples/default.jsp HTTP/1.1
```

请求方法为 POST，URI 为/examples/default.jsp，而协议/版本为 HTTP/1.1。

HTTP 1.1 规范定义了 7 种类型的方法，包括 GET、POST、HEAD、OPTIONS、PUT、DELETE 以及 TRACE，其中 GET 和 POST 广泛应用于互联网应用。

URI 定义了一个互联网资源，通常解析为服务器根目录的相对路径。因此，通常用/符号打头。另外 URL 是 URI 的一个具体类型。（详见 http://www.ietf.org/rfc/rfc2396.txt。）

HTTP 请求所包含的请求头信息包含关于客户端环境以及实体内容等非常有用的信息。例如，浏览器所设置的语言实体内容长度等。每个 header 用回车/换行（即 CRLF）分隔。

HTTP 请求头信息和请求正文用一行空行分隔，HTTP 服务器据此判断请求正文的起始位置。因此在一些关于互联网的书籍中，CRLF 作为 HTTP 请求的第四种组件。

在此前所举的例子中，请求正文如下行：

```
lastName=Blanks&firstName=Mike
```

在正常的 HTTP 请求中，请求正文的内容不止如此。

HTTP 响应

同 HTTP 请求一样，HTTP 响应包含三部分：

- 协议—状态码—描述
- 响应头信息
- 响应正文

如下是一个 HTTP 响应实例：

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Date: Thu, 8 Jan 2015 13:13:33 GMT
Content-Type: text/html
Last-Modified: Wed, 7 Jan 2015 13:13:12 GMT
Content-Length: 112

<html>
<head>
<title>HTTP Response Example</title>
</head>
<body>
Welcome to Brainy Software
</body>
</html>
```

类似于 HTTP 请求报文，HTTP 响应报文第一行说明了 HTTP 协议的版本是 1.1，并且请求结果是成功的（状态代码 200 为响应成功）。

同 HTTP 请求报文头信息一样，HTTP 响应报文头信息也包含了大量有用的信息。HTTP

响应报文的响应正文是 HTML 文档。HTTP 响应报文的头信息和响应正文也是用 CRLF 分隔的。

状态代码 200 表示 Web 服务器能正确响应所请求的资源。若一个请求的资源不能被找到或者理解，则 Web 服务器将返回不同的状态代码。例如：访问未授权的资源将返回 401，而使用被禁用的请求方法将返回 405。完整的 HTTP 响应状态代码列表详见如下网址：

```
http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
```

本书内容简介

第一部分：Servlet 和 JSP

第 1 章："Servlets"，介绍 Servlet API，本章重点关注两个 java 包：javax.servlet 和 javax.servlet.http packages。

第 2 章："会话管理"，讨论了会话管理——在 Web 应用开发中非常重要的主题（因为 HTTP 是无状态的），本章比较了 4 种不同的状态保持技术：URL 重写、隐藏域、Cookies 和 HTTPSession 对象。

第 3 章："JavaServer Pages（JSP）"，JSP 是 Servlet 技术的补充完善，是 Servlet 技术的重要组成部分，本章包括了 JSP 语法、指令、脚本元素和动作。

第 4 章："表达式语言"，本章介绍了 JSP 2.0 中最重要的特性"表达式语言"。该特性的目标是帮助开发人员编写无脚本的 JSP 页面，让 JSP 页面更加简洁而且有效。本章将帮助你学会通过 EL 来访问 Java Bean 和上下文对象。

第 5 章："JSTL"，本章介绍了 JSP 技术中最重要的类库：标准标签库——一组帮助处理常见问题的标签。具体内容包括访问 Map 或集合对象、条件判断、XML 处理，以及数据库访问和数据处理。

第 6 章："自定义标签"，大多数时候，JSTL 用于访问上下文对象并处理各种任务，但对于特定的任务，我们需要编写自定义标签，本章将介绍如何编写标签。

第 7 章："标签文件"，本章介绍在 JSP 2.0 中引入的新特性——标签文件，标签文件可以简化自定义标签的编写。

第 8 章："监听器"，本章介绍了 Servlet 中的事件驱动编程，展示了 Servlet API 中的事件类以及监控器接口，以及如何应用。

第 9 章："Filters"，本章介绍了 Filter API，包括 Filter、FilterConfig 和 FilterChain 接口，并展示了如何编写一个 Filter 实现。

第 10 章：“修饰 Requests 和 Responses”，本章介绍如何用修饰器模式来包装 Servlet 请求和响应对象，并改变 Servlet 请求和响应的行为。

第 11 章：“异步处理”，本章主要讨论 Servlet 3.0 引入的新特性——异步处理。该特性非常适合于当 Servlet 应用负载较高且有一个或多个耗时操作。该特性允许由一个新线程来运行耗时操作，使得当前的 Web 请求处理线程可以处理新的 Web 请求。

第 12 章：“安全”，介绍了如何通过声明式以及编程式来保护 Java Web 应用，本章覆盖四个主题：认证、授权、加密和数据完整性。

第 13 章：“部署”，介绍了 Servlet/JSP 应用的部署流程，以及部署描述符。

第 14 章：“动态加载以及 Servlet 容器加载器”介绍了 Servlet 3.0 中的两个新特性，动态注册支持在无须重启 Web 应用的情况下注册新的 Web 对象，以及框架开发人员最关心的容器初始化。

第二部分：Spring MVC

第 15 章：“Spring 框架”，介绍了最流行的开源框架。

第 16 章：“模型 2 和 MVC 模式”，讨论了 Spring MVC 所实现的设计模式。

第 17 章：“Spring MVC 介绍”，Spring MVC 概述。本章编写了第一个 Spring MVC 应用。

第 18 章：“基于注解的控制器”，讨论了 MVC 模式中最重要的一个对象——控制器。本章，我们将学会如何编写基于注解的控制器，这是 Spring MVC 2.5 版本引入的方法。

第 19 章：“数据绑定和表单标签库”，讨论 Spring MVC 最强大的一个特性，并利用它来展示表单数据。

第 20 章：“转换器和格式化”，讨论了数据绑定的辅助对象类型。

第 21 章：“验证器”，本章将展示如何通过验证器来验证用户输入数据。

第 22 章：“国际化”，本章将展示如何用 Spring MVC 来构建多语言网站。

第 23 章：“上传文件”，介绍两种不同的方式来处理文件上传。

第 24 章：“下载文件”，介绍如何用编程方式向客户端传输一个资源。

附录

附录 A：“Tomcat”，介绍如何安装和配置 Tomcat。

附录 B：“Web Annotations”，列出所有可用配置 Web 对象，如 Servlet、Listener 或 Filter 的注解。这些来自 Servlet 3.0 规范的注解可以帮助减少部署描述配置。

附录 C：“SSL 证书”，介绍了如何用 KeyTool 工具生成公钥/私钥对，并生成数字证书。

下载示例应用

本书所有的示例应用压缩包可以通过如下地址下载：

http://books.brainysoftware.com/download

目 录

第一部分 Servlets 和 JSP

第二部分 Spring MVC