异步图书
www.epubit.com.cn

BrainySoftware

Spring MVC 学习指南

（第2版）

[美] Paul Deck 著
林仪明 译

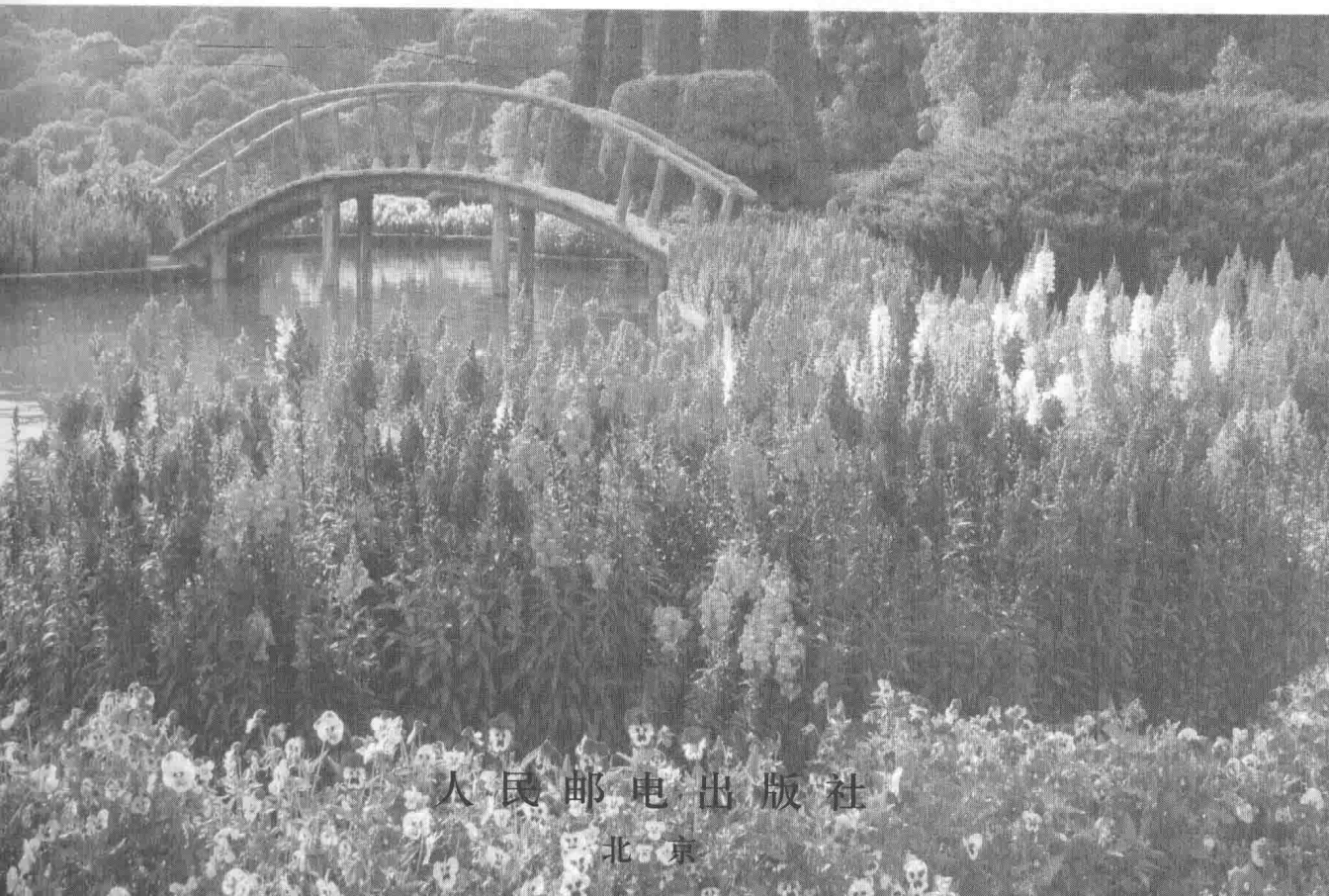
中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

BrainySoftware

Spring MVC 学习指南

（第2版）

[美] Paul Deck 著
林仪明 译



人民邮电出版社
北京

图书在版编目（CIP）数据

Spring MVC学习指南 : 第2版 / (美) 戴克 (Paul Deck) 著 ; 林仪明译. -- 北京 : 人民邮电出版社, 2017.5
ISBN 978-7-115-44759-3

Ⅰ. ①S… Ⅱ. ①戴… ②林… Ⅲ. ①JAVA语言－程序设计－指南 Ⅳ. ①TP312.8-62

中国版本图书馆CIP数据核字(2017)第031857号

版权声明

◆ 著　[美] Paul Deck
译　林仪明
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
北京鑫正大印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：21.5
字数：406 千字　2017 年 5 月第 1 版
印数：1－3 000 册　2017 年 5 月北京第 1 次印刷

著作权合同登记号　图字：01-2016-5102 号

定价：59.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广字第 8052 号

内容提要

Spring MVC 是 Spring 框架中用于 Web 应用快速开发的一个模块，其中的 MVC 是 Model-View-Controller 的缩写。作为当今业界最主流的 Web 开发框架，Spring MVC 已经成为当前最热门的开发技能，同时也广泛用于桌面开发领域。

本书重在讲述如何通过 Spring MVC 来开发基于 Java 的 Web 应用。全书共包括 13 章和 5 个附录，分别从 Spring 框架、模型 2 和 MVC 模式、Spring MVC 介绍、控制器、数据绑定和表单标签库、转换器和格式化、验证器、表达式语言、JSTL、国际化、上传文件、下载文件以及应用测试等多个角度介绍了 Spring MVC。附录部分分别介绍了 Tomcat、Spring Tool Suite 和 Maven 等工具的安装和配置，Servlet、JavaServer Pages 和部署描述符的相关参考资料。除此之外，本书还配有丰富的示例以供读者练习和参考。

本书是一本 Spring MVC 的教程，内容细致、讲解清晰，非常适合 Web 开发者和想要使用 Spring MVC 开发基于 Java 的 Web 应用的读者阅读。

作者简介

Paul Deck 是一位资深的 Java 和 Spring 专家，开发过大型的企业级应用，并且目前是一位独立的软件工程师。他是 《How Tomcat Works》一书的合著者。

译者简介

林仪明，男，现为 IBM 高级工程师。2004 年毕业于厦门大学软件学院，主要研究软件架构、应用中间件。目前在福州生活和工作，先后从事软件技术开发，软件架构设计以及团队管理等工作，有多年的开发设计和管理经验，目前提供 IBM 中间件产品支持工作。

前言

欢迎阅读本书。

Spring MVC 是 Spring 框架中用于 Web 应用快速开发的一个模块。Spring MVC 的 MVC 是 Model-View-Controller 的缩写。它是一个广泛应用于图形化用户交互开发中的设计模式，不仅常见于 Web 开发，也广泛应用于如 Swing 和 JavaFX 等桌面开发。

作为当今业界最主流的 Web 开发框架，Spring MVC（有时也称 Spring Web MVC）的开发技能相当热门。本书可供想要学习如何通过 Spring MVC 开发基于 Java 的 Web 应用的开发人员阅读。

Spring MVC 基于 Spring 框架、Servlet 和 JSP（JavaServer Page），在掌握这 3 种技术的基础上学习 Spring MVC 将非常容易。本书第 1 章针对 Spring 新手提供一个快速教程，附录 C 和附录 D 将帮助你快速学习 Servlet 和 JSP。如果希望深入学习 Servlet 和 JSP，推荐阅读由 Budi Kurniawan 所著的《*Servlet and JSP：A Tutorial，second Edition*》。

接下来，我们将介绍 HTTP、基于 Servlet 和 JSP 的 Web 编程，以及本书的章节结构。

HTTP

HTTP 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。作为一个制定标准的国际社区，万维网联盟（W3C）负责和维护 HTTP。HTTP 第一版是 HTTP 0.9，随后更新为 HTTP 1.0，之后的版本是 HTTP 1.1。HTTP 1.1 版本的 RFC 编号是 2616。编写本书时，HTTP 1.1 依然是当前最流行的 HTTP 版本。当前最新版本是发布于 2015 年 5 月的 HTTP/2，附表 1.1 列出了 HTTP 各个版本及其发布时间，HTTP 的第二个主要版本通常表示为 HTTP/2，而不是 HTTP2。

附表 1.1　HTTP 版本与发布日期

版本	发布时间
HTTP 0.9	1991 年
HTTP 1.0	1996 年
HTTP 1.1	1997 年发布，1999 年更新
HTTP/2	2015 年 5 月

Web 服务器每天 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，客户端发起一个连接，服务端不会主动连接客户端。

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

```
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 外，URL 还可以采用其他类型的协议，以下为两个示例：

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

通常，HTTP 的 URL 格式如下：

protocol://[host.]domain[:port][/context][/resource][?query string | path variable]

或者

protocol://IP Address[:port][/context][/resource][?query string | path variable]

中括号中的内容是可选项。因此，一个最简单的 URL 是 http://yahoo.ca 或者是 http://192.168.1.9。

需要说明的是，除了输入 http://google.com 外，还可以用 http://173.194.46.35 来访问谷歌。可以用 ping 命令来获取域名对应的 IP 地址。

```
ping google.com
```

由于 IP 地址不容易记忆，所以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）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，无需输入端口号。有时，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，为了能正确访问服务器，必须提供输入端口号。

```
http://localhost:8080/index.html
```

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。若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）。一个不带资源名称的 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。

资源名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。

接下来，我们介绍 HTTP 请求和响应。

HTTP 请求

一个 HTTP 请求包含 3 部分内容。

1．方法—URI—协议/版本。

2．请求头信息。

3．请求正文。

下面为一个 HTTP 请求示例。

```
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 (Macintosh; U; Intel Mac OS X 10.5; en-US; rv:1.9.2.6)
Gecko/20100625 Firefox/3.6.6
Content-Length:30
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate

lastName=Blanks&firstName=Mike
```

请求的第一行 POST /examples/default.jsp HTTP/1.1 是方法—URI—协议/版本。

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。

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 请求包含的请求头信息，包括关于客户端环境以及实体内容等非常有用的信息。例如，浏览器设置的语言、实体内容长度等。每个 header 都用回车/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 被作为 HTTP 请求的第 4 种组件。

示例中，请求正文是 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 响应也包含 3 部分。

1．协议—状态码—描述。

2．响应头信息。

3．响应正文。

下面为一个 HTTP 响应示例。

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Date: Thu, 29 Sep 2013 13:13:33 GMT
Content-Type: text/html
Last-Modified: Web, 28 Sep 2013 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 响应报文的第一行说明了 HTTP 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网址 http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html。

Servlet 和 JSP

Java Servlet 技术是 Java 体系中用于开发 Web 应用的底层技术。1996 年，Servlet 和 JSP 由 SUN 系统公司发布，以替代 CGI 技术，作为产生 Web 动态内容的标准。CGI 技术为每一个请求创建相应的进程，但是，创建进程会耗费大量的 CPU 周期，最终导致很难编写可伸缩的 CGI 程序。相对于 CGI 程序，一个 Servlet 则快多了，这是因为当一个 Servlet 为响应第一次请求而创建后，会驻留在内存中，以便响应后续请求。

从 Servlet 技术出现的那天起，人们开发了大量的 Web 框架来帮助程序员快速编写 Web 应用程序。这些开发框架让开发人员能更关注业务逻辑，减少编写“相似”的代码片段。尽管如此，开发人员依然需要去理解 Servlet 技术的基础概念。随后发布的 JavaServer Pages（JSP）技术，是用来帮助简化 Servlet 开发。尽管实践中会应用一些诸如 Spring MVC、Struct 2 或者 JSF 等强大的开发框架，但如果没有深入理解 Servlet 和 JSP 技术，则无法有效和高效地开发。Servlet 是运行在 Servlet 容器中的 Java 程序，而 Servlet 容器或 Servlet 引擎相当于一个 Web 服务器，但是可以产生动态内容，而不仅是静态资源。Servlet 当前的版本为 3.1，其规范定义可见 JSR （Java Specification Request）340（http://jcp.org/en/jsr/detail?id=340），基于 Java 标准版本 6 及以上版本。JSP 2.3 规范定义在 JSR 245（http://jcp.org/en/jsr/detail?id=245）。本书假定读者已经了解 Java 以及面向对象编程技术。对于 Java 新手，推荐阅读《Java 7：A Beginner's Tutorial（fourth Edition）》。

一个 Servlet 是一个 Java 程序，一个 Servlet 应用包含了一个或多个 Servlet，一个 JSP 页

面会被翻译并编译成一个 Servlet。

一个 Servlet 应用运行在一个 Servlet 容器中，它无法独立运行。Servlet 容器将来自用户的请求传递给 Servlet 应用，并将 Servlet 应用的响应返回给用户。由于大部分 Servlet 应用都会包含一些 JSP 界面，故称 Java Web 应用为“Servlet/JSP”应用会更恰当些。

Web 用户通过一个诸如 IE、火狐或者 Chrome 等 Web 浏览器来访问 Servlet 应用。Web 浏览器又称为 Web 客户端。图 1 展示了一个典型的 Servlet/JSP 应用架构。

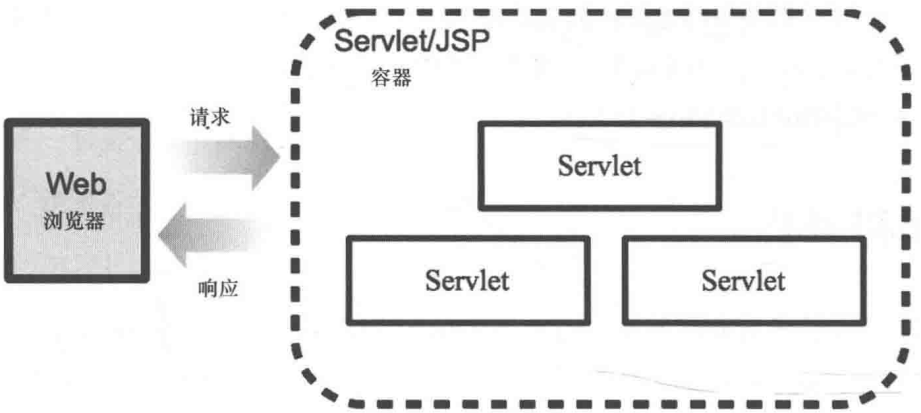


图 1 Servlet/JSP 应用架构

Web 服务端和 Web 客户端基于 HTTP 通信。因此，Web 服务端往往也称为 HTTP 服务端。

一个 Servlet/JSP 容器是一个能处理 Servlet 以及静态资源的 Web 服务端。过去，由于 HTTP 服务器更加健壮，人们更愿意将 Servlet/JSP 容器作为 HTTP 服务器（如 Apache HTTP 服务器）的一个模块来运行，在这种场景下，Servlet/JSP 容器用来产生动态内容，而 HTTP 服务器处理静态资源。今天，Servlet/JSP 容器已经更加成熟，并且被广泛地独立部署。Apache Tomcat 和 Jetty 作为最流行的 Servlet/JSP 容器，免费而且开源。下载地址为 http://tomcat.apache.org 和 http://jetty.codehaus.org。

Servlet 和 JSP 仅是 Java 企业版众多技术之一，其他 Java 企业版技术包括 JMS、EJB、JSF 和 JPA 等。Java 企业版 7（当前最新版）完整的技术列表见 Http://www.oracle.com/technetwork/java/javaee/tech/index.html。

运行一个 Java 企业版应用需要一个 Java 企业版容器，常见的容器有 GlassFish、JBoss、Oracle Weblogic 以及 IBM WebSphere。虽然可以将一个 Servlet/JSP 应用部署到 Java 企业版容

器中，但一个 Servlet/JSP 容器其实就足够了，而且更加轻量级。Tomcat 和 Jetty 不是 Java 企业版容器，故它们不能运行 EJB 或 JMS。

下载 Spring 或使用 STS 与 Maven/Gradle

Spring 网站建议使用 Maven 和 Gradle 来下载 Spring 库及其依赖包。现在的应用通常有很多依赖包，并且这些依赖包也有自己的依赖。通过一个依赖管理工具，可以把我们从解析和下载依赖的工作中解放出来。并且这些工具还可以帮助构建、测试和打包应用。Maven 和 Gradle 是当下流行的依赖管理系统，使用这两个工具是一个明智的选择。然而当下还有很多 Java 开发者要么不熟悉这两个工具，要么不喜欢使用或者不想使用这两个工具。一个好的教程不能要求他们先学习 Maven 或 Gradle，然后才能构建和测试他们的 Spring 应用程序。因此，本书所有的例子都有两种发行包：基于 Maven 的发行包和不依赖 Maven/Gradle 的发行包。

Maven 曾经是 Spring 支持的唯一的依赖管理系统。Spring 开发工具 Spring Tool Suites（STS），是一个基于 Eclipse 的 IDE，捆绑了一个最新的 Maven 版本。这对于不熟悉 Maven 的开发人员是一个好消息，只要熟悉 Eclipse，则无须学习 Maven 就可以应用其好处。然而，在写作本书的时候，最新版本的 STS 并不是没有缺点。例如，当使用 STS 创建 Spring MVC 应用程序时，生成的应用程序会因为缺少 Servlet 部署描述符（web.xml 文件）而导致 Maven 插件生成一个错误消息。因此开发人员不得不被迫处理一个 Maven 问题。当然，可以通过简单地创建一个 web.xml 文件来解决它。但是，从 Servlet 3.0 开始，web.xml 文件是可选的，并且可以构建一个没有 web.xml 文件的 Spring MVC 应用程序。

下面对于使用 Maven/Gradle 或不使用工作会分别介绍。

手动下载 Spring

如果不想通过 Gradle 和 Maven 来构建应用，则应从 Spring 仓库中下载其类库：

```
http://repo.spring.io/release/org/springframework/spring/
```

点击以上链接将跳转到最新版本（写作本书时为版本为 4.2.5），选择一个 zip 发行包，发行包命名格式如下：

```
spring-framework-x.y.z.RELEASE-dist.zip
```

其中，x.y.z 表示 Spring 的主要版本和次要版本。

将下载好的 zip 解压到任意目录。在解压的目录中，包含相应的文档和 Java 源代码，其中 libs 文件夹下为基于 Spring 框架开发应用所需的 jar 文件。

使用 STS 和 Maven/Gradle

如果选择使用 Maven 或 Gradle，则无需手动下载 Spring。具体内容，请参考本书附录 B。

下载 Spring 源码

Spring 框架是一个开源项目，如果你喜欢冒险或者你想要的尚未发布的最新版本的 Spring，你可以使用 Git 下载源代码。克隆 Spring 源代码的命令如下：

```
git clone git://github.com/spring-projects/spring-framework.git
```

此外还需使用 Gradle 来从源代码构建 Spring。Git 和 Gradle 工具不是本书的范围。

本书内容简介

第 1 章：Spring 框架，介绍了最流行的开源框架。

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

第 3 章：Spring MVC 介绍，编写了第一个 Spring MVC 应用。

第 4 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。在本章中，我们将学习如何编写基于注解的控制器，该方式由 Spring MVC 2.5 版本引入。

第 5 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 6 章：转换器和格式化，讨论了数据绑定的辅助对象类型。

第 7 章：验证器，展示如何通过验证器来验证用户输入数据。

第 8 章：表达式语言，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 9 章：JSTL，介绍了 JSP 技术中最重要的类库——标准标签库。这是一组帮助处理常见问题的标签，诸如访问 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 10 章：国际化，将展示如何用 Spring MVC 来构建多语言网站。

第 11 章：上传文件，介绍两种不同的方式来处理文件上传。

第 12 章：下载文件，介绍如何用编程方式向客户端端传输一个资源。

第 13 章：应用测试，介绍如何使用 Junit、Mockito 和 Spring MVC Test 进行单元测试和集成测试。

附录 A：Tomcat，介绍如何安装和配置 Tomcat。

附录 B： Spring Tool Suite 和 Maven，介绍如何安装并配置 Spring Tool Suite，并且开发和运行 Spring MVC 应用。

附录C：Servlet，介绍了Servlet API并展示几个简单的Servlet应用。本附录重点关注Servlet API的4个包中的两个，即javax.servlet和javax.servlet.http。

附录D：JavaServer Pages，介绍了JSP语法，包括指令、脚本元素和actions。

附录E：部署描述符，介绍如何配置Spring MVC应用以便部署。

下载示例应用

本书所有的示例应用压缩包可以通过以下地址下载：

http://books.brainysoftware.com/download 和 http://www.epubit.com.cn。

目 录