

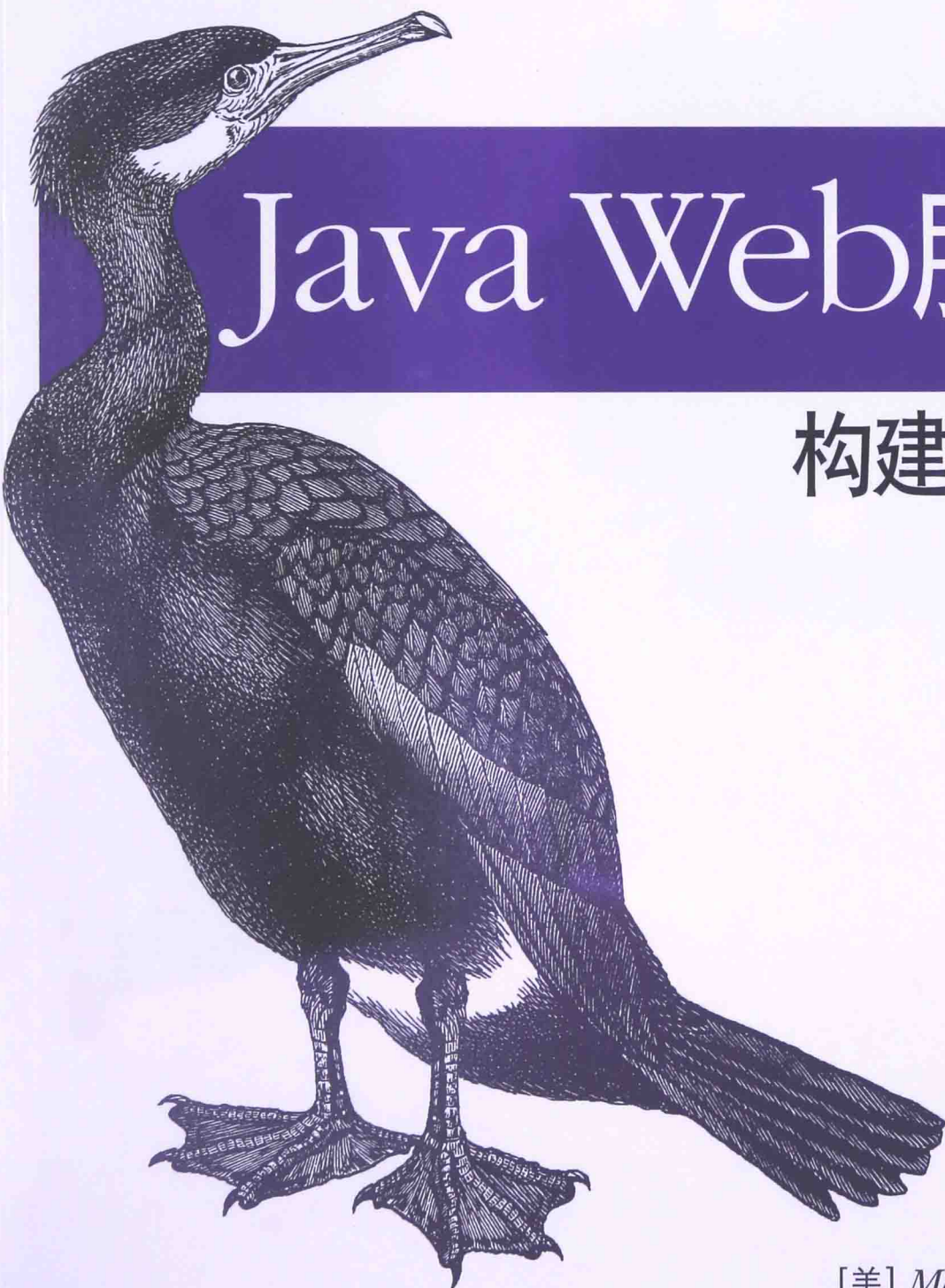
Java Web Services: Up and Running

快速、实用、深入的介绍

第2版

Java Web服务

构建与运行



[美] Martin Kalin 著

卢涛 李颖 译

O'REILLY®



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

Java Web服务: 构建与运行 (第2版)

Java Web Services: Up and Running 2nd Edition

[美] Martin Kalin 著
卢涛 李颖 译

电子工业出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书快速并详尽地介绍了如何开发 REST 式和基于 SOAP 的 Web 服务。这本动手实践型的书以清晰、务实的方法讲述 Web 服务相关技术, 提供了架构性的概述、可以运行的完整代码示例, 以及编译、部署和执行应用程序的一些短小精悍的指令。学习本书, 读者将掌握如何从头开始编写服务或将已有的服务整合到现有的 Java 应用程序中。

在突出强调 REST 式服务的同时, 此第 2 版还涵盖了 HttpServlet、Restlet、JAX-RS API, 针对 REST 式服务的 jQuery 客户端, 以及用于基于 SOAP 的服务的 JAX-WS。

©2013 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Publishing House of Electronics Industry, 2013. Authorized translation of the English edition, 2013 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

本书简体中文版专有出版权由 O'Reilly Media, Inc. 授予电子工业出版社。未经许可, 不得以任何方式复制或抄袭本书的任何部分。专有出版权受法律保护。

版权贸易合同登记号 图字: 01-2014-2673

图书在版编目 (CIP) 数据

Java Web服务: 构建与运行: 第2版 / (美) 卡林 (Kalin, M.) 著; 卢涛, 李颖译. —北京: 电子工业出版社, 2014.11

书名原文: Java web services: up and running (2nd edition)

ISBN 978-7-121-24501-5

I. ①J… II. ①卡… ②卢… ③李… III. ①JAVA语言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字(2014)第234670号

责任编辑: 付 睿

封面设计: Randy Comer 张 健

印 刷: 北京中新伟业印刷有限公司

装 订: 北京中新伟业印刷有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路173信箱 邮编: 100036

开 本: 787×980 1/16 印张: 22.25 字数: 465千字

版 次: 2014年11月第1版

印 次: 2014年11月第1次印刷

定 价: 69.00元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

O'Reilly Media, Inc. 介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始, O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来, 而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者, O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”; 创建第一个商业网站 (GNN); 组织了影响深远的开放源代码峰会, 以至于开源软件运动以此命名; 创立了 Make 杂志, 从而成为 DIY 革命的主要先锋; 公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖, 共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择, O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版、在线服务或者面授课程, 每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列 (真希望当初我也想到了) 非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人, 他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了: ‘如果你在路上遇到岔路口, 走小路 (岔路)。’ 回顾过去 Tim 似乎每一次都选择了小路, 而且有几次都是一闪即逝的机会, 尽管大路也不错。”

——Linux Journal

前言

欢迎阅读《Java Web 服务：构建和运行》的第 2 版。这个版本，与第 1 版一样，是写给有兴趣开发 Web 服务及其客户端的程序员们的。这个版本，仍与第 1 版一样，强调代码。我的目标是通过重点而现实的 Java 编程实例，使 Web 服务及其客户端变得栩栩如生，但是，当然，也包括其他语言：Web 服务被设计成语言中立的，用不同语言编写的服务端与客户端交互正是这一点最好的说明。的确，一个设计良好的 Web 服务的客户端对有关服务的实现细节，包括编写该服务的程序语言，可以一直是不了解的。为了减轻编译及发布服务的工作量，示例代码的 ZIP 文件中包含一个 Ant 脚本，它负责编译、封装和部署 Web 服务。主要的客户端例子要么包括编译和运行客户端的 Ant 脚本，要么包含所有的依赖文件在内的可执行的 JAR 文件。该代码示例可在 <https://github.com/mkalin/jwsur2> 中获得。

第 2 版中的变化

第 1 版出版距今已有四年左右的时间，这期间既有延续也有变化。分布式软件系统 (*distributed software systems*)，也就是需要位于不同物理设备上的软件进行交互的系统，对于此类系统，Web 服务仍然是流行且充满争议甚至可以说是占主导地位的方法。Web 本身是分布式系统的一个最好的例子，而目前的趋势是模糊传统的以 HTML 为中心的网站 (*websites*) 和现代化的 Web 服务 (*web services*) 之间的区别，后者通常交付 XML 或 JSON 有效载荷而不是 HTML。Web 服务是创建分布式系统的一种吸引人的方式，因为这些服务可以建立在现有的基础设施，如 HTTP (S) 传输、Web 服务器、数据库系统、各种现代编程语言、广泛的对 JSON 和 XML 进行处理的软件库、安全提供者，等等之上。事实上，Web 服务是整合不同的软件系统并且让这样的系统的功能便于取得的一种轻量级且灵活的方式。

Java 仍然是 Web 服务的主要参与者,并且 Java 以标准和第三方软件库与实用程序的形式,不断提高对这些服务的支持。然而,自这本书第 1 版出版以来,两个需要强调的重要的和相关的变化也随之出现了:

- Web 服务的消费者或客户端越来越多地采用 JavaScript 编写,特别是用 jQuery 方言编写,而这些客户端自然更愿意响应 JSON (JavaScript Object Notation, JavaScript 对象符号) 形式的有效载荷,而不是 XML 形式的有效载荷,因为 JSON 文档是一个原生的 JavaScript 对象的文本表示。例如,接收以 JSON 而不是 XML 文档形式的一个产品数组的 JavaScript 客户端,可以用通常的 JavaScript 编程结构处理该数组。与此相反,接收一个 XML 有效负载的 JavaScript 客户端将面临跨编程语言的共同挑战:转移到特定的应用程序逻辑之前解析一个 XML 文档提取其信息内容的挑战。现代的 Web 服务和 Web 服务框架通过将 JSON 和 XML 格式平等对待承认了 JSON 的日益普及。在一些框架中,如 Rails,JSON 甚至比 XML 更胜一筹。
- REST 式服务在大家都熟悉的网站,如 eBay、Facebook、LinkedIn、Tumblr 和 Twitter 中,越来越受欢迎。亚马逊是 Web 服务的先锋,它继续支持其服务 REST 式和基于 SOAP 的版本。来自较新玩家的服务往往是 REST 式的一个明显的原因是:REST 式服务都相对更易使用,其 API 也相对简单。基于 SOAP 的服务仍然大多数是通过 HTTP (S) 交付的,尽管 Java 和 .NET 继续探索使用其他传输协议,特别是 TCP。在这本书的第 1 版中强调,通过 HTTP 的以 SOAP 为基础的服务可以被看作是 REST 式服务的一个特例,第 2 版奉行同样的主题。

Web 服务的两个变化都体现在第 2 版的组织方式上。第 1 章开始于 Web 服务的概述,包括这些服务和面向服务架构 (Service-Oriented Architecture, SOA) 之间的联系,并且这一章还包括 SOA 和竞争的分布式对象架构 (Distributed Object Architecture, DOA) 的一个基于代码的对比。随后讨论转向 REST:这个缩写的含义是什么,为什么 HTTP 可以被视为一个 API,而不仅仅是一个传输工具,以及 REST 式思维方式如何持续影响着现代 Web 服务的设计和实现。第 1 章包括用 Java 编写的示例 HTTP 客户端,可以针对网站或 Web 服务中的任意一个目标的客户端。这一章以实现为带有两个后端 POJO 类进行支持的一个 JSP 脚本的 REST 式服务结束,该服务使用 Tomcat Web 服务器发布。第 1 章深入安装和运行 Tomcat 的细节,第 2 章对 Jetty Web 服务器做了同样的讲述。上述 Ant 脚本也得到解释,使得示例 Web 服务可以自动打包和部署。

虽然本书这个版本开始于 REST 式服务,但也透彻地研究了基于 SOAP 的服务。第 4 章包括在应用层 (*application level*) 的基于 SOAP 的服务,在这一层中 SOAP 仍然是透明的;第 5 章探讨了处理程序 (*handler*) 和传输 (*transport*) 层,在该层 SOAP 被公开以供检查和操纵。从 REST 式服务开始有助于解释基于 SOAP 的服务的优势,尤其是让 XML

仍然隐藏在幕后的好处。诸如跨越整个 REST/SOAP 边界的安全问题，第 6 章致力于实用的 Web 的安全性，从线路层的安全性到用户 / 角色的安全性直到 WS-Security。

Web 服务 API 和发布选项

在第 1 版中，占优势的是 JAX-WS API 及其 Metro 实现。在这一版中，这两个仍很重要，但不占主导地位。对于 REST 式服务，本书的例子基于以下 API：

HttpServlet

HttpServlet 是精心设计的 REST 式服务，因为此 API 与 HTTP 的本质是非常接近的。servlet 实例封装了回调方法，如 doPost、doGet、doPut 和 doDelete，它们涵盖了我們熟悉的 CRUD 操作：创建（*create*）（POST）、读取（*read*）（GET）、更新（*update*）（PUT）和删除（*delete*）（DELETE）。包含表明 HTTP 请求结果的 HTTP 状态码的符号版本，对 MIME 类型的支持，访问 HTTP 标头和正文的实用工具，等等。JSP 和其他基于 Java 的脚本是作为 servlet 实例执行的，因此，也被归入 servlet 的保护伞下。HttpServlet 虽然是年代久远的，但几乎没有过时。servlet 仍然是提供 REST 式服务的绝佳方式。

JAX-RS

这是一个相对较新的和日益流行的提供 REST 式服务的 API。此 API 的中心是注解，如 @GET 和 @POST，它们把 HTTP 请求路由到特定的 Java 方法。还有一个同样方便的 @Path 注解用来识别某个请求中的特定目标资源。JAX-RS 可以被配置为自动生成 XML 和 JSON 响应。这个 API，就像下面即将描述的 Restlet API 一样，拥有现代的外观和感觉。在实现层面上，JAX-RS 代表了 servlet 之上的一层。它们的 JAX-RS 同类有发布基于 servlet 的服务的同样选项。

Restlet

这个 API 在风格上与 JAX-RS 相似，虽然这个说法很可能会激怒双方的支持者。该 Restlet API 也是以注解方式将 HTTP 请求路由到指定的 Java 方法和产生有效载荷为中心的。Restlet 鼓励与其他 API 的互动。例如，在以 Restlet 为基础的服务中使用 JAX-RS 注解，这是可能的。Restlet 提供了一个易于使用的发布者进行开发和测试。Restlet 服务，就像它们的 JAX-RS 同类，表示在 servlet 之上的实现层次。程序员应该能够在 JAX-RS 和 Restlet API 之间轻松转移。

JAX-WS @WebServiceProvider

这是一个特意以 XML 为中心的低级别 API，它可以用于任何基于 SOAP 或 REST

式服务。然而, JAX-WS 还有 `@WebService` 注解, 它正是为基于 SOAP 的服务准备的, 因此, `@WebServiceProvider` 注解最明显的使用是基于 XML 的 REST 式服务。这个 API 非常适合那些需要精细地控制 XML 生成及处理的服务。

对于基于 SOAP 的服务, 大多数例子使用 JAX-WS 的参考实现, 即 Metro。然而, 本书这个版本现在也涵盖 Axis2。Axis2 实现了 JAX-WS, 但有更多的功能。

发布选项

上述这些 API, 无论是用于 REST 式服务还是基于 SOAP 的服务, 在发布 Web 服务方面, 都遵守关注分离的原则。Web 服务是一个值得关注的问题, 而它的发布是另一个值得关注的问题。这些 API 中的任何一个开发的服务都可以使用一个独立的 Web 服务器如 Tomcat, Java 应用程序服务器 (JAS), 如 GlassFish, 甚至用一个简单的命令行实用程序, 如标准 Endpoint 发布者发布。为了强调关注分离的原则, 并强调生产级的选项, 将使用以下方式发布例子:

独立的 Web 服务器

在 Java 中有两个明显的选择是 Tomcat 和 Jetty, 虽然其中还有其他的选择。上述 Ant 脚本自动编译和打包 Web 服务用于发布, REST 式服务和基于 SOAP 的服务都是一样的。虽然 Ant 脚本是专为 Tomcat 的发布定制的, 但它生成的 WAR 文件可以不加修改地部署到 Jetty、Tomcat 或许多 JAS 中的某个之上。Tomcat 和 Jetty 可以提供通常的服务, 如线路级和用户 / 角色的安全性, 日志 / 调试和管理这些人们期望从生产级 Web 服务器得到的功能。

Java 应用程序服务器

参考实现仍然是 GlassFish, 这是基于社区的 Metro 项目的一部分。GlassFish 既可以用来发布基于 *servlet*(*servlet-based*) 的服务, 这是 Tomcat 和 Jetty 可以发布的类型, 也可以发布基于 EJB (*EJB-based*) 的服务, 这是 `@Stateless` 会话企业 JavaBean。TomEE 基本上是带有 OpenEJB 扩展的 Tomcat7, 是既能发布基于 *servlet* 的服务又能发布基于 EJB 的服务的一个重要的轻量级发布者。在 TomEE 下, 甚至基于 EJB 的服务也可以部署为标准的 WAR (Web 归档) 文件。TomEE 包括 JAX-RS 的一个实现。

命令行发布者

这类例子包括标准的 Endpoint 实用工具类和 Restlet 的 Component 类。这些发布者都用于开发、测试, 甚至还可以用于小规模的生产。

Java 一般从语言和在运行时提供的很多选项中汲取力量, 这种力量也延续到 Web 服务。

用 Java 编写 Web 服务和 Web 服务客户端有很多方法，并有各种用于发布这些服务的有吸引力的选项。没有必要声称在 Web 服务中的任何特定的方式是最佳方式。我的目标是研究和解释各种选择，使得 API、实现和发布方式最终可以通过最适合的服务来确定。

逐章概述

本书共分为 7 章。下面提供了每章的摘要。

第 1 章，Web 服务快速入门

本章用对 REST 和 SOAP 之间的差异和相似性的概述开始了代码驱动的 Web 服务旅程。为什么 Web 服务如此有用？这个问题用例子得到了解决：一个例子侧重于使用 Web 服务来自动获取在 Web 上可用的数据和功能，而另一个例子侧重于 Web 服务作为一种方法来集成不同的软件系统。贯穿全书都使用实例追求互操作性（*interoperability*）的主题。第 1 章包括 Web 服务的简史，并侧重于讲解分布式系统的 SOA 方法与更早出现但还在继续与 Web 服务进行竞争的 DOA 方法的显著不同之处。本章随后重点介绍，在基于网络的分布式软件系统的 REST 式的方法中，HTTP 本身是如何处于中心地位的。引入 XML 和 JSON 作为对 REST 式服务特别重要的文件交换格式。本章包括如下代码例子：用来说明 HTTP 的主要功能的一对 Java HTTP 客户端，以及由一个 JSP 脚本和两个后端 POJO 类组成的第一个 REST 式服务。*curl* 工具是用来针对第一个服务做客户端的示例调用的，包括失败的那些调用。本章涵盖实际问题，如安装 Tomcat Web 服务器，并使用提供的 Ant 脚本编译、打包和部署 Web 服务。

第 2 章，REST 式的 Web 服务：服务端

本章介绍各种可用于编写和提供 REST 式 Java Web 服务的实现和 API。通过完整的代码示例探讨了 `HttpServlet`、JAX-RS、Restlet 和 JAX-WS `@WebServiceProvider` APIs。本章同时阐明使用标准的 Java 类和不同的第三方类生成 XML 和 JSON 的有效载荷的各种方法。这些代码示例都坚持 REST 式的原则，如遵守每个 CRUD 动词的含义、用直观的 URI 式来命名资源、依靠 MIME 数据类型来描述资源的表示形式，以及利用那些报告对 REST 式服务的 HTTP 请求结果的 HTTP 状态代码。第 2 章连同后面的章节，都着眼于发布 REST 式服务的选项。这些选项包括独立的 Web 服务器，如 Tomcat 和 Jetty，还包括命令行的发布者，如 Endpoint、HttpServer 和 Restlet Component。本章深入介绍了多线程及采用一个 Web 服务器，如 Tomcat 或 Jetty 部署的服务中的线程同步的技术细节。Jetty 的安装和管理也包括在内。第 2 章还初步涉及了强大的 JAX-B（用于 XML 绑定的 Java API）和 JAX-P（用于 XML 处理的 Java API）实用工具，这在第 3 章涵盖的 REST 式服务的客户端中是特别重要的。

第3章, REST 式的 Web 服务: 客户端

本章重点从服务端转移到 REST 式服务的客户端。既有用久经风霜的但仍然值得信赖的 `URLConnection` 类编写的示例客户端, 也有使用 REST 特定的 API 编写的客户端。(JAX-RS、Restlet 和 JAX-WS 都同时提供服务端和客户端 API。)作为互操作性的证据, 本章提供针对 Java 服务的 jQuery 和 Perl 的客户端与针对未公开实现语言的商业性服务的 Java 客户端。该代码示例探讨处理 XML 和 JSON 的有效载荷的各种可能性, 特别是标准的 JAX-B 包和第三方的贡献, 如 XStream。这些实用工具在将 XML 文档转换成原生的 Java 对象时特别有用, 因为这避免了对明确解析的要求。大多数 REST 式服务现在都以一个 XML 模式或等效的形式为服务提供一个语法, 核心 Java 有诸如转换 XML 模式到 Java 类的 `xjc` 等实用工具。第3章介绍了针对 Twitter、亚马逊和芝加哥高速运输管理局的实际的 REST 式服务的客户端。本章特别重视 JavaScript 客户端日益增加的重要性, 这突出体现在几个使用 jQuery 的例子中。最后, 本章还介绍了如何精心策划不同的 Web 服务, 使之形成一个单一的综合服务。

第4章, 基于 SOAP 的 Web 服务

本章将从 REST 式服务转到基于 SOAP 的服务, 特别是对 JAX-WS 的 API 及其中心注解 `@WebService`。本章以前面章节的 REST 式服务转换为基于 SOAP 的服务开篇。本章的重点是在应用层, 在这一层上, 在基于 SOAP 的消息中的 XML 仍然是透明的。事实上, 基于 SOAP 的服务的主要吸引力在于, 无论是服务, 还是它们的客户都不需要对底层的 XML 有任何关注: 在 Java 中, 服务操作最好用 `@WebMethod` 注解的 `public` 方法, 而远程客户端直截了当地调用了这些操作。参数和返回值的数据类型包括所有的基本类型和它们的包装、String 和 Calendar 类型、其他各种标准类型、任何可接受类型的数组, 以及程序员定义的任何属性最终简化为上述任何一种类型的类。本章详细介绍了对程序员友好的 `wsimport` 工具, 它用 Web 服务合同, 即 WSDL (Web Service Description Language Web 服务描述语言) 文件生成客户端支持代码。本章通过编码实例阐明 WSDL 的结构、用途, 以及它的各种用法, 并对 WSDL 文档中的 XML 模式或等效的角色给予了特别的重视。本章包括针对 Amazon 电子商务服务的两个 Java 客户端, 以及针对 Java 服务的 C# 和 Perl 客户端。这些例子强调, 基于 SOAP 的服务, 就像它们的 REST 式同类一样, 都是语言中立的。针对 Amazon 电子商务服务的基于 SOAP 的版本的客户端介绍了基于 SOAP 的服务的处理程序层, 但不对它进行深入研究, 这些例子还提供了一个对安全问题的初步观察。

第5章, SOAP 处理程序和错误

本章考察了 SOAP 消息中的处理程序层和传输层, 在这些层上, 在 SOAP 消息中的 XML 就可以用于检查和处理。本章首先观察 SOAP 消息的架构, 它区分了消息发送

者 (*sender*)，应将其活动限制在 SOAP 标头；而不处理 SOAP 正文或消息附件的中介人 (*intermediary*)；以及应该能够访问整个 SOAP 消息的最终接收者 (*receiver*)。SOAP 处理程序可以访问 SOAP 消息的不同部分，包括原始的 XML 和任何附件，它有两种形式：消息处理程序 (*message handlers*) 可以访问整个 SOAP 消息 (标头、正文和附件)，而逻辑处理程序 (*logical handlers*) 只能访问在正文内的有效载荷。WS-Security 和超出了基本配置 (*basic profile*) 的 SOAP 相关的扩展可以使用这种访问注入或检查 SOAP 标头中的安全元素。处理程序是类似于网站上的过滤器 (*Filter*) 实例，处理程序本质上是双向的，可以出现在客户端或服务端任一方。本章同时涵盖单独的处理程序和处理程序链，在这个例子中，链是由消息和逻辑处理程序组成的。处理程序链的例子模仿在 Amazon 的 Web 服务中工作的用户身份验证。本章也探讨 SOAP 错误 (*faults*)，即可以在应用层或处理程序层上产生特殊的错误信息的相关主题。本章还着眼于 SOAP 消息如何可以将任意的二进制数据作为附件传输。最后一个话题是传输层，通常是 HTTP，这一层对将在第 6 章中介绍的用户/角色的安全性检查特别有用。

第 6 章，Web 服务安全性

本章涵盖安全性，这个主题需跨越 REST/SOAP 的边界，在三个层：即 HTTPS 提供的类型线路层的安全性、用户/角色的安全性和在基于 SOAP 的消息中的 WS-Security。本章从线路层的安全性和组成它的对等身份验证 (*peer authentication*)，消息机密性 (*message confidentiality*) 和消息完整性 (*message integrity*) 服务开始阐述。这些术语的解释，反过来又要求对诸如对称 (*symmetric*) 和非对称 (*asymmetric*) 加密/解密、公钥的安全性 (*public key security*)、加密散列函数 (*cryptographic hash function*) 和加密套件 (*cryptographic suite*) 等概念进行研究。通过编写代码的例子详细研究 HTTPS：针对谷歌网站以及核心 Java JDK 中自带的 `HttpsServer` 类构建的一个轻量级的 HTTPS 服务器的两个 HTTPS 客户端。本章还介绍使用 REST 式服务的示例 HTTPS 客户端。这些和其他的编写代码例子阐明额外的安全性构件，如密钥库 (*keystore*) 和信任库 (*truststore*)，数字证书 (*digital certificates*) 和证书颁发机构 (*certificate authority*)。有一节专门解释如何在如 Tomcat 的 Web 服务器上进行设置处理，甚至强制 HTTPS 连接。第 6 章还包括容器管理的 (*container-managed*) 用户/角色的安全性，又具有涉及生产级 Web 服务器如 Tomcat 的编码实例。本章对用户身份验证和角色授权过程这两个阶段进行了详细的研究，并列举了 HTTP 基本 (BASIC) 和摘要 (DIGEST) 式身份验证的例子。在本章结束前介绍了一个专注于 WS-Security 及其端到端的安全方法的代码示例。

第 7 章，Web 服务和 Java 应用程序服务器

本章介绍 Java 应用程序服务器作为独立的 Web 服务器部署 REST 式服务和基于

SOAP 的服务。JAS 作为服务发布者也带来了作为一个 EJB 部署 Web 服务的选项，特别是作为一个 @Stateless Session（无状态会话）EJB 的 Web 服务。本章从列举流行的 JAS 开始，然后解释了 JAS 的核心部件及其功能，并回顾了前面章节中的服务，从 HttpServlet 的实例到 JAX-RS、Restlet、@WebServiceProvider 和 @WebService 的实例，以及如何将这些服务移植到一个 JAS 中。在所有情况下，几乎无须任何更改。第 7 章还介绍选择使用 JAS 部署服务的动机，特别是在线程安全的 EJB 容器中部署服务所附带的好处。一个典型的 JAS 不仅是发布者，还是开发、测试和管理环境。本章有一个可以深入了解的 Web 服务和网站如何互动的编码例子，并且通过使用两个在不同的数据库（HSQLDB 和 Java Derby）中持久化数据的编码示例介绍了 JPA（Java Persistence API，Java 持久性 API）。本章还通过代码示例介绍了两个 JAS：GlassFish，它是参考实现；以及 TomEE，这是 JAS 中强调轻便的选项。各种边栏说明安装和管理的细节。本章在结束前评论并建议程序员接受 Java 在 Web 服务领域提供的许多优秀的 API 和实现。

工具和 IDE（集成开发环境）

Java 程序员有大量的生产力工具可供选择。其中生成工具有 Ant、Maven 和 Meister，测试工具有 EasyMock、JMockit、JUnit、Mockito 和 TestNG。Java 同样提供 IDE 的选择，包括 Eclipse、IntelliJ IDEA 和 NetBeans。在生产环境中，最好是使用工具和 IDE，因为它们隐藏了从最初的设计到部署和维护的繁杂细节。在一个学习环境中，使用构建工具有道理的，因为它有利于实验。我的 Ant 脚本正是为了服务于这个目的：它允许用单个命令对 Web 服务及其所有的依赖文件进行编译、构建和部署：

```
% ant deploy -Dwar.name=myFirstService
```

第 1 章进入细节设置，这是最简化的。

对于 IDE，像第 1 版一样，本书第 2 版是不带倾向性的。非常繁杂的细节在生产环境中是障碍物，但对学习环境却是至关重要的。出于这个原因，我的代码示例包括所有的 import 语句，使得依赖关系是明确的。每当使用第三方库时，都会对包 / 目录结构进行说明。为便于参考和解释，代码清单通常包含带编号的语句和表达式。这些例子本身旨在突出对任何编程固有的挑战，这些例子还注重用模式化的方法来迎接这些挑战。Web 服务及其客户端最终是代码，而这本书主要讲的就是代码。

本书中所使用的约定

本书使用以下排版约定：

斜体 (*Italic*)

表示新的术语、URL、文件名、文件扩展名和重点。

等宽字体 (Constant Width)

用于程序清单以及段落中引用的程序元素，如变量或方法名、数据类型、环境变量、语句和关键字。

边栏

书中使用边栏（参见下面的“这是一个边栏”），以专注于特定的主题。边栏通常包含有关安装和运行应用程序的内容，如一个独立的 Web 服务器或 Java 应用服务器的实用信息。

书中切口处的“”表示原书页码。

这是一个边栏

一个特别感兴趣的话题。

使用代码示例

这本书的目的是帮助你完成工作。一般来说，如果这本书包括代码示例，那么你可以在你的程序和文档中使用这些代码。你不需要就许可问题与我们联系，除非你直接复制大部分的代码。例如，使用这本书的几段代码编写程序不需要获得许可。销售或用 CD-ROM 分发 O'Reilly 书籍中的例子，则需要获得许可。通过引用本书和引用示例代码来回答案问题不需要获得许可。将本书大量的示例代码加入到你的产品文档则需要获得许可。

我们对署名表示赞赏及感谢，但不强制要求这么做。一个署名通常包括书名、作者、出版商和 ISBN。例如：“Java Web 服务：构建和运行，第 2 版，Martin Kalin 著。版权所有 2013 Martin Kalin, 978-1-449-36511-0”。

如果当你觉得对代码示例的使用超出合理使用范围或上面给出的权限时，请随时通过 permissions@oreilly.com 与我们联系。

Safari® 联机丛书



Safari 联机丛书 (*Safari Books Online*) 是一家点播数字图书馆，它同时提供来自世界各地领先的技术和业务作者的书籍和视频两种形式的专业内容。

专业技术人员、软件开发人员、网页设计师，以及商业和创意专业人士使用 Safari 联机丛书作为研究、解决问题、学习和认证培训的主要资源。

Safari 联机丛书提供了一系列的产品组合和针对组织、政府机构和个人的定价方案。用户有机会在一个完全可搜索的数据库中访问成千上万的书籍、培训视频和即将出版的原稿，它们来自 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及大量其他出版商。有关 Safari 联机丛书的更多详细信息，请访问我们的网站。

如何联系我们

对于本书的评论或问题请联系出版商：

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室 (100035)
奥莱利技术咨询(北京)有限公司

有关本书的网页、勘误表、样例和其他信息，请登录：

<http://oreil.ly/Java-web-services>

关于本书的评论或技术讨论可发送邮件至：

bookquestions@oreilly.com

如果想了解关于本书、会议、资源中心或 O'Reilly 网络的其他相关信息，请登录网站：

<http://www.oreilly.com>

在 Facebook 上找到我们：<http://facebook.com/oreilly>

在 Twitter 上关注我们：<http://twitter.com/oreillymedia>

在 YouTube 上观看我们：<http://www.youtube.com/oreillymedia>

致谢

Edward Yue Shung Wong 和 Ken Yu 都慷慨地审查了这本书，并提出许多有见地的改善建议。他们使本书变得更好。我衷心感谢他们为这个项目投入的时间和精力。当然，本书其余的不足之处都是我一个人的责任。我的编辑 Meghan Blanchette 为我提供了宝贵的支持，没有她的帮助就不会有这本书。我还要感谢 O'Reilly Media 公司曾为这个项目工作的许多幕后工作人员。

这个版本，与第 1 版一样，是特别献给 Janet 的。

在本书的翻译过程中，李绿霞、卢林、陈克非、张慧珍、李又及、卢晓瑶、陈克翠、汤有四、刘雯、李阳、贾书民、苏旭辉、李洪秋也参与了部分翻译工作，在此表示诚挚的感谢。

目录

前言	v
第 1 章 Web 服务快速入门	1
1.1 Web 服务杂项	3
1.2 Web 服务有什么好处	4
1.3 Web 服务和面向服务的架构	6
1.4 Web 服务简史	8
1.4.1 从 DCE/RPC 到 XML-RPC	8
1.4.2 分布式对象架构：Java 示例	10
1.4.3 用 Web 服务来救援	12
1.5 什么是 REST	13
1.5.1 动词和不透明名词	16
1.6 回顾 HTTP 请求和响应	17
1.7 HTTP 作为 API	19
1.7.1 用 Java 编写的两个 HTTP 客户端	20
1.8 第一个 REST 式的例子	23
1.8.1 Predictions Web 服务的工作原理	24
1.8.2 针对 Predictions Web 服务的客户端	32
1.9 为什么 REST 式 Web 服务要使用 servlet	33
1.10 下一步是什么	35

第 2 章 REST 式的 Web 服务：服务端	39
2.1 REST 式服务作为一个 HttpServlet 类	40
2.1.1 实现细节	41
2.1.2 针对 predictions2 服务的示例客户端调用	52
2.2 REST 式 Web 服务作为一个 JAX-RS 资源	53
2.2.1 第一个使用 Jersey 的 JAX-RS Web 服务	54
2.2.2 用 Java 应用程序发布 JAX-RS 资源	55
2.2.3 用 Tomcat 发布 JAX-RS 资源	56
2.2.4 Adage 类	58
2.2.5 JAX-RS 生成的 XML 和 JSON 响应	62
2.2.6 将 Predictions Web 服务移植到 JAX-RS	66
2.3 REST 式 Web 服务作为 Restlet 的资源	75
2.3.1 对 adages2 服务的示例调用	83
2.3.2 在不使用 Web 服务器的情况下发布 adages2 Restlet 服务	84
2.3.3 REST 式服务作为一个 @WebServiceProvider	85
2.4 下一步是什么	94
 第 3 章 REST 式的 Web 服务：客户端	 95
3.1 针对 Java REST 式 Web 服务的 Perl 客户端	96
3.2 针对亚马逊电子商务服务的客户端	101
3.3 一个独立的 JAX-B 示例	110
3.3.1 XStream 选项	114
3.4 针对亚马逊电子商务服务的另一个客户端	119
3.5 CTA 公共汽车跟踪服务	123
3.6 REST 式客户端和 WADL 文档	126
3.7 JAX-RS 客户端 API	132
3.8 用于 JavaScript 客户端的 JSON	134
3.8.1 JSONP 和 Web 服务	135
3.8.2 使用 jQuery 编写的组合 REST 式服务	136
3.8.3 一个 Ajax 轮询的例子	139
3.9 下一步是什么	142