# Java 与 Restful Web Service

袁 赟

（同济大学软件学院，上海 201804）

**摘要**：近年来 Web 服务领域发生着翻天覆地的变化，继传统的 XML-RPC 风格的 Web Service 之后，一种新的风格，REST 被应用于 Web Service。本文主要介绍了 Rest 构架风格，以及现在 Java 开发 Restful Web Service 的主要方式，以及如何实现 Restful Web Service 的 Java 和 Ajax 客户端，最后介绍了 Restful Web Service 的描述语言 WADL。

**关键词**：Rest；Restful Web Service；Ajax；WADL

**中图分类号**：TP311　　**文献标识码**：A　　**文章编号**：1009-3044(2007)21-40780-03

## Restful Web Service in Java

YUAN Yun

(Tongji University Software School,Shanghai 201804,China)

**Abstract**:In the past several years, there have been dramatic changes to the Web services landscape. After the traditional XML-RPC web service, a new style- Representational State Transfer (REST) is applied to Web services. This article mainly introduce REST architecture, and the ways to develop Restful Web Service in java, and how the clients to invoke the Restful web services using java or Ajax ways, and also the WSDL to Rest --Web Application Description Language (WADL).

**Key words**:Rest;Restful Web Service;Ajax;WADL

## 1 引言

近年来 Web 服务领域发生着翻天覆地的变化，继传统的 XML-RPC 风格的 Web Service 之后，一种新的风格，REST 被应用于 Web Service。

REST 是 Representational State Transfer（表述性状态转移）的缩写，它最初是 Roy Thomas Fielding 2000 年在他的博士论文中提出的，是针对分布式系统的软件架构风格。使用符合 REST 设计约束的 Web 上部署的组件可以充分利用 Web 的有用特性。

REST 是以资源为中心的，在 REST 中，认为 Web 是由一系列的抽象资源（Abstract Resource）组成，这些抽象的资源具有不同的具体表现形式((Representational State)，外界可以通过 URI 定位，修改，删除资源。通过 REST 架构，Web 应用程序可以用一致的接口（URI）暴露资源给外部世界，并对资源提供语义一致的操作服务，在网络中有很多资源（名词），而用一致的动作去访问他们。

## 2 Restful Web Service 概述

Web 服务技术让应用程序可以用平台独立或编程语言独立的方式相互通信。但是传统的 XML-RPC 风格的 Web 服务，正逐渐受到复杂性恶魔的威胁，称作 REST 风格的 Web Services 提供了更简单的替代方式。

所谓的 Restful Web Services 是指使用 REST 体系结构风格创建的轻量级的 Web 服务。REST Web 服务是面向资源的服务。可以通过统一资源标识符（Universal Resource Identifier，URI）来识别和定位资源，并且针对这些资源而执行的操作是通过 HTTP 规范定义的，通过 GET 操作检索资源，POST 操作执行资源的特定于应用程序形式的更新，PUT 操作创建新资源。DELETE 操作销毁 URI 指向的资源。现实中 HTTP 动词中的 GET 和 POST 较常使用的，而 PUT 和 DELETE 方法则相对较少使用。

Restful Web Service 充分利用现有 web 基础设施，REST 系统中所有的动作和访问的资源都可以从 HTTP 和 URI 中得到，用不同的 HTTP 请求方法来处理对资源的 CRUD（创建、读取、更新和删除）操作。而且响应可以被标示成可缓存的或是不可缓存的，从而使得代理服务器、缓存服务器和网关很好地协调工作，提高了网络效率。

## 3 Java 与 Restful Web Service

Java 中与 Restful Web Service 相关的规范有 (1) JSR 224(JAX-WS)，通过 javax.xml.ws.Provider 接口来构建 REST 风格的终端。(2) JSR311，又叫 JAX-RS (The Java API for RESTful Web Services)，正如名字所示，是 JCP 社区关于 RESTfull Web Service 的规范请求。它是一个规范，而不是真正的 API,JSR 311 有一个专门的项目 jsr311.dev.java.net，并且被列为即将到来的 Java EE 6 规范的一部分。

目前有两种流行的 rest 风格的 url(1)传统以查询字符串的形式传递参数，如 richbooks.com/viewbook?isbn=12345。(2)以"/"分割参数的更 Cool 的方式，如 richbooks.com/viewbook/isbn/12345。

本文以一个虚拟的网上购书系统为例说明在 Java 中实现 Restful Web Service 的不同方式。用户通过输入 URL：richbooks.com/books,查看所有书书籍的列表，richbooks.com/viewbook/isbn/12345 来查看具体的某本书，管理员通过 richbooks.com/addbook/isbn/12345/title/ RESTful+ Web+Services 添加书籍，richbooks.com/editbook/isbn/12345 编辑书籍，richbooks.com/deletebook/isbn/12345 删除书籍。

### 3.1 使用 JAX-WS 实现 RESTful Web Service

JAX-WS 是 Java Architecture for XML Web Services 的缩写，是一种用 Java 和 XML 开发 Web Services 应用程序的

收稿日期：2007-08-19

作者简介：袁赟（1981-），男，四川平昌人，硕士研究生，研究方向：企业计算。

框架，目前版本是2.0，它是JAX-RPC 1.1的后续版本。可以通过JAX-WS的javax.xml.ws.Provider接口来构建REST风格的Web服务。

用JAX-WS开发RESTful Web Service，首先创建一个javax.xml.ws.Provider<T>接口的实现。Provider接口是标准端点实现类的另一种动态实现办法，Provider<T>是一个泛型类。它能用SOAP/HTTP绑定支持Provider<javax.xml.transform.Source>和Provider<javax.xml.soap.SOAPMessage>，或者用XML/HTTP绑定支持Provider<javax.activation.DataSource>和Provider<javax.xml.transform.Source>。当创建Provider的实现时，可以选择处理哪种形式的请求与响应消息。

具体实现代码如下：

```
@WebServiceProvider @BindingType (value = HTTPBinding.HTTP_BINDING)
public class GetBookImpl implements Provider<javax.xml.transform.Source> {
@Resource protected WebServiceContext wsContext;
public Source invoke(Source source) {
MessageContext mc = wsContext.getMessageContext();
String path = (String) mc.get (MessageContext.PATH_INFO);
StringTokenizer st = new StringTokenizer(path, "=&/");
st.nextToken();
String isbn = st.nextToken();
Book book = BookManager.getInstance().getBooksByISBN(isbn);
String body = "<ns:GetBookResponse xmlns:ns=\"http://java.duke.org\"><ns:isbn>"
+ book.getIsbn()+ "</ns:isbn><ns:title>"+ book.getTitle()+ "</ns:title><ns:author>"
+ book.getAuthor ()+ "</ns:author>"+ "</ns:return></ns:GetBookResponse>";
return new StreamSource (new ByteArrayInputStream(body.getBytes()));
}
}
```

其中@WebServiceProvider表明该类是一个基于Provider的端点而不是一个用@WebService指明的服务端点实现类，通过标记是@BindingType(value= HTTPBinding.HTTP_BINDING)指明了GetBookImpl端点将用HTTPBindingHTTP_BINDING而不是SOAPBindingSOAP11HTTP_BINDING或者SOAPBinding.SOAP12HTTP_BINDING来绑定。@Resource标记，告知JAX-WS在运行时注入WebServiceContext到我们的实例中，然后GetBookImpl从WebServiceContext中获得MessageContext，再从MessageContext中获得URL路径，最后从URL路径中获得参数，即要请求的图书的isbn号。

JAX-WS Web Service在不同的应用服务器中配置不同，在此就不再熬述。

3.2 JSR311实现RESTful Web Service

Jersey是Sun的JAX-RS(JSR311)开源参考实现，用于构建RESTful Web service，支持JSON,WADL,并且在JAXB的基础上提供了针对JSON的notations。Jersey还提供一些额外的API和扩展机制，开发人员能够按照自己的需要对Jersey进行扩展。

具体实现代码如下：

```
@UriTemplate("/") @XmlAccessorType(XmlAccessType.PROPERTY)
@XmlType(name = "bookType", propOrder = { "isbn", "title", "author" })
public class Book {
private String isbn;
private String title;
private String author;
@UriTemplate("viewbook/{isbn}/") @HttpMethod("GET")
@ProduceMime( { "application/json", "application/xml" })
public Book getBookAsXMLOrJSON(@UriParam("isbn") String isbn)
{return BookManager.getInstance().getBooksByISBN(isbn);}
@UriTemplate("viewbook/{isbn}/") @HttpMethod("GET")
@ProduceMime("text/plain")
public String getBookAsText (@UriParam("isbn")String isbn) {
Book book = BookManager.getInstance().getBooksByISBN(isbn);
return "isbn=" + book.getIsbn() + ";title=" + book.getTitle()
+ ";author=" + book.getAuthor();
}
//以下是构造函数和pulic的getter/setter
}
```

其中@UriTemplate指出一个资源类或类中的方法要serve的请求的URI。@HttpMethod指出方法可以用来处理的HTTP请求的类型。@ProduceMime指出方法serve请求之后返回的MIME类型，从上例可以看出不同的方法可以serve同一个URI，返回不同的数据格式：XML，JSON，html,普通文本，同样可以使用@ConsumeMime指出方法可以接受的MIME类型，用不同的方法来服务请求同一URI的不同MIME类型。这样的话客户端可以使用他最熟悉的格式(XML,JSON，html或者普通文本)来向服务传输数据，以及从接受服务返回的数据，这些极大地提高了Restful Web Service开发的灵活性。

3.3 比较

比较上面的实现Restful Web Service的不同方式，可以看出JSR311的方式，比较符合现有的Web体系结构，用类或方法对应请求的URI，利用annotation简化了开发门同时可以接受和返回的多种数据格式，非常灵活。是目前Java开发Restful Web Service的较好选择.

## 4 Restful Web Service客户端

4.1 JAX-WS Restful Web Service的Java客户端

JAX-WS Restful Web Service客户端是javax.xml.ws.Service的实例，实例的创建可以动态的和静态的，静态的方式就是通过相关WADL to Java类似的工具，创建相关的stub。动态调用方式是通过面向消息调用的javax.xml.ws.Dispatch实例，javax.xml.ws.Service接口作为创建Dispatch实例的工厂。

静态调用方式要利用到后文介绍的 WADL 来生成静态的 stub。下面通过代码来演示在 Java 中如何动态调用 JAX-WS Restful Web Service。

```
QName serviceQName = new QName("http://duke.example.org", "ViewBookService"); QName portQName = new QName("http://duke.example.org", "ViewBooksPort");
String endpointAddress = "http://localhost:8080/BookService/viewbook";
String pathInfo = "/BookService/viewbook/isbn/12345";
Service service = Service.create(serviceQName);
service.addPort (portQName, HTTPBinding.HTTP_BINDING, endpointAddress);
Dispatch<Source> d = service.createDispatch(portQName, Source.class,
Service.Mode.MESSAGE);
Map<String, Object> requestContext = d.getRequestContext();
requestContext.put (MessageContext.HTTP_REQUEST_METHOD, "GET");
requestContext.put (MessageContext.PATH_INFO, pathInfo);
Source result = d.invoke(null);
TransformerFactory.newInstance().newTransformer()
.transform(result, new StreamResult(System.out));
```

4.2 JSR311 Restful Web Service 的 Ajax 客户端

Ajax 表示 Asynchronous JavaScript+XML，是 JavaScript、CSS、DOM 和 XMLHttpRequest 等技术的结合。在 Ajax 中，核心的技术主要是围绕实现与服务器的异步通信，而无需页面刷新。使用 Ajax 可以(1)减轻服务器的负担。因为 Ajax 的根本理念是“按需取数据”，所以最大可能在减少了冗余请求和响影对服务器造成的负担，节约空间和带宽租用成本。(2)提高用户体验，无刷新更新页面，减少用户实际和心理等待时间。

由于 Restful Web Service 使用标准的 HTTP 操作 GET、PUT、POST 和 DELET 来检索和修改资源。这使得它非常适合和 Ajax 一起工作。

一般在应用中不会完全自己去写 js 来实现能跨浏览器的 ajax 代码，而是利用现有的 Ajax 框架，如 Prototype, dojo, Open Rico, DWR 等等。在以下的代码中采用 dojo1.0 release 版本。

```
<script type="text/javascript" src="js/dojo/dojo.js"></script>
< script language="javascript" type="text/javascript">
function getBook() {
var isbn= dojo.byId ("isbn").value;
dojo.xhrGet( {
url :"http://<ip>:<port>/BookService/viewbook/isbn/"+isbn,
handleAs :"xml",
load : function(response, ioArgs) {
dojo.byId ("isbn").value =response.getElementsByTagName("isbn")[0].firstChild.data;
dojo.byId ("title").value=response.getElementsByTagName("title")[0].firstChild.data;
dojo.byId ("author").value =response.getElementsByTagName("author")[0].firstChild.data;
return response;
},
error : function(response, ioArgs)
{dojo.byId ("error").innerHTML = response;}
});
}
</ script >
```

以上代码通过dojo.xhrGet 函数向 http://<ip>:<port>/BookService/viewbook/ Restful Web Service 发出一个 Ajax 请求，然后解析返回的 XML 文件，并更新页面。

## 5 WADL：Restful Web Service 的描述语言

Web Application Description Language(WADL)是由 SUN 公司提出的，旨在提供一种 Web 应用的描述语言。

客户要调用一个 web 服务，必须知道都提供了哪些服务，以及如何调用它们，传递的参数，和返回值的类型等等信息。SOAP 类型的 Web Service 采用 WSDL 描述，而在 WADL 以前，Restful Web Service 只能以文本方式描述，维护，修改都很不方便。WADL 使用 XML 以人可读，机器可以处理的方式正规化描述 Restful Web Service 提供的契约。

WADL 主要描述一个 Restful Web Service 应用的以下内容：

(1)资源列表-站点所有的资源

(2)资源之间的关系-说明资源之间的链接关系

(3)所有应用于每个资源的特定方法-应用于每个资源的 HTTP 方法，指定的输入和输出以及它们支持的数据格式

(4)资源表现的形式-支持的 MIME 类型和所用到的 XML schemas

而且 wadl.dev.java.net 项目提供了 wadl2java.jar 工具用来解析 WADL 文件并生产 Java 代码，Google 的 REST Describe & Compile 工具，可以帮助生成 WADL 文件，以及生成 Java,C#,PHP5,Ruby,Python 客户端代码。

## 6 结束语

目前越来越多企业采用的 Rest 风格的 Web Service，而不是 XML-RPC 风格的 Web Service，其中包括 Yahoo，eBay，Amazon，del.icio.us 等 IT 巨头。相信随着未来 JSR311 和 WADL 的出现和不断完善，Restful Web Service 会继续受到青睐，不断蓬勃的发展。

当然相对于传统的 XML-RPC 风格的 Web Service，Restful Web Service 也有它的不足，比如现在它还无法用于事务型的服务。因此要根据这个服务是针对资源的还是针对活动的，来选择 REST 或 XML-RPC 风格的 Web 服务。

## 参考文献：

[1]RESTful Web Services, Leonard Richardson, Sam Ruby, and David Heinemeier Hansson, 2007.5

[2]SOA Using Java Web Services, Mark D. Hansen, 2007.9

[3]Architectural Styles and the Design of Network-based Software Architectures, Roy Thomas Fielding, 2000