Web Service 那点事儿（1）

2017/05/27 | 分类： 基础技术 | 0 条评论 | 标签： Service, Web

 原文出处： 黄勇

Web Service，即“Web 服务”，简写为 WS，从字面上理解，它其实就是“基于 Web 的服务”。而服务却是双方的，有服务需求方，就有服务提供方。服务提供方对外发布服务，服务需求方调用服务提供方所发布的服务。其实也就是这些了，没有多少高大上的东西。

本文将从实战的角度，描述使用 Java 开发 WS 的工具及其使用过程。

如果说得再专业一点，WS 其实就是建立在 HTTP 协议上实现异构系统通讯的工具。没错！WS 说白了还是基于 HTTP 协议的，也就是说，数据是通过 HTTP 进行传输的。

自从有了 WS，异构系统之间的通讯不再是遥不可及的梦想。比如：可在 PHP 系统中调用 Java 系统对外发布的 WS，获取 Java 系统中的数据，或者把数据推送到 Java 系统中。

如果您想了解更多关于 WS 的那些概念与术语，可以看看下面的百度百科：

```
http://baike.baidu.com/view/67105.htm
```

今天我想与大家分享的主题是，如何在 Java 中发布与调用 WS？希望本文能够对您有所帮助！

1. 使用 JDK 发布 WS

第一步：您要做的第一件事情就是，写一个服务接口。

```
<!-- lang: java -->
package demo.ws.soap_jdk;

import javax.jws.WebService;

@WebService
public interface HelloService {

    String say(String name);
}
```

在接口上放一个 WebService 注解，说明该接口是一个 WS 接口（称为“Endpoint，端点”），其中的方法是 WS 方法（称为“Operation，操作”）。

第二步：实现这个 WS 接口，在实现类中完成具体业务逻辑，为了简单，我们还是写一个 Hello World 意思一下吧。

```
<!-- lang: java -->
package demo.ws.soap_jdk;

import javax.jws.WebService;

@WebService(
    serviceName = "HelloService",
    portName = "HelloServicePort",
    endpointInterface = "demo.ws.soap_jdk.HelloService"
)
public class HelloServiceImpl implements HelloService {

    public String say(String name) {
        return "hello " + name;
    }
}
```

第三步：写一个 Server 类，用于发布 WS，直接使用 JDK 提供的工具即可实现。

```
<!-- lang: java -->
package demo.ws.soap_jdk;

import javax.xml.ws.Endpoint;

public class Server {

    public static void main(String[] args) {
```

```
        String address = "http://localhost:8080/ws/soap/hello";
        HelloService helloService = new HelloServiceImpl();

        Endpoint.publish(address, helloService);
        System.out.println("ws is published");
    }
}
```

只需使用 JDK 提供的 javax.xml.ws.Endpoint 即可发布 WS，只需提供一个 WS 的地址（address），还需提供一个服务实例（helloService）。

现在您就可以运行 Server 类的 main 方法了，会在控制台里看到“ws is published”的提示，此时恭喜您，WS 已成功发布了！

第四步：打开您的浏览器，在地址栏中输入以下地址：

```
http://localhost:8080/ws/soap/hello?wsdl
```

注意：以上地址后面有一个 ?wsdl 后缀，在 Server 类中的 address 里却没有这个后缀。此时，在浏览器中会看到如下 XML 文档：

```
▼<!--
   Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6.
 -->
▼<!--
   Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6.
 -->
▼<definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://soap_jdk.ws.demo/" xmlns:xsd="http://www.w3.org/2001/XM
 xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://soap_jdk.ws.demo/" name="HelloService">
 ▼<types>
  ▼<xsd:schema>
     <xsd:import namespace="http://soap_jdk.ws.demo/" schemaLocation="http://localhost:8080/ws/soap/hello?xsd=1"/>
   </xsd:schema>
  </types>
 ▼<message name="say">
    <part name="parameters" element="tns:say"/>
  </message>
 ▼<message name="sayResponse">
    <part name="parameters" element="tns:sayResponse"/>
  </message>
 ▼<portType name="HelloService">
  ▼<operation name="say">
     <input message="tns:say"/>
     <output message="tns:sayResponse"/>
   </operation>
  </portType>
 ▼<binding name="HelloServicePortBinding" type="tns:HelloService">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
  ▼<operation name="say">
     <soap:operation soapAction=""/>
   ▼<input>
       <soap:body use="literal"/>
     </input>
    ▼<output>
       <soap:body use="literal"/>
     </output>
   </operation>
  </binding>
 ▼<service name="HelloService">
  ▼<port name="HelloServicePort" binding="tns:HelloServicePortBinding">
     <soap:address location="http://localhost:8080/ws/soap/hello"/>
   </port>
  </service>
</definitions>
```

当看到这份 WSDL 文档时，也就意味着，您发布的 WS 服务现在可以被别人使用了。

2. 通过客户端调用 WS

第一步：使用 JDK 提供的命令行工具生成 WS 客户端 jar 包。

JDK 安装目录下有个 bin 目录，里面存放了大量的命令行工具，只要您的 Path 环境变量指向了该路径，就能在命令控制台上使用 JDK 提供的相关命令。

其中，有一个名为 wsimport 的命令行工具，正是用来通过 WSDL 生成 WS 客户端代码的，您只需要输入以下命令即可：

```
wsimport http://localhost:8080/ws/soap/hello?wsdl
jar -cf client.jar .
rmdir /s/q demo
```

对以上三行命令解释如下：

- 第一行：通过 WSDL 地址生成 class 文件
- 第二行：通过 jar 命令将若干 class 文件压缩为一个 jar 包
- 第三行：删除生成的 class 文件（删除根目录即可）

最终您将会得到一份名为 client.jar 的 jar 包，将这个 jar 包配置到您的 classpath 中，方便在下面的代码中使用其中的类。

技巧：可以将以上三行命令放入一个 bat 文件中，在 Windows 中双击即可运行。

第二步：写一个 Client 类，用于调用 WS，需要使用上一步生成的 WS 客户端 jar 包。

```
<!-- lang: java -->
package demo.ws.soap_jdk;

public class Client {
```

```
    public static void main(String[] args) {
        HelloService_Service service = new HelloService_Service();

        HelloService helloService = service.getHelloServicePort();
        String result = helloService.say("world");
        System.out.println(result);
    }
}
```

以上这段代码稍微有点怪异，其中 HelloService_Service 是 jar 包中类，可以将其理解为 WS 的工厂类，通过它可以生成具体的 WS 接口，比如，调用 service.getHelloServicePort() 方法，就获取了一个 HelloService 实例，正是通过这个实例来调用其中的方法。

运行 Client 类的 main 方法，就会看到您所期望的结果“hello world”了，不妨亲自尝试一下吧。

可见，这是一个典型的“代理模式”应用场景，您实际是面向代理对象来调用 WS 的，并且这是一种“静态代理”，下面我们来谈谈，如何使用“动态代理”的方式来调用 WS?

其实 JDK 已经具备了动态代理的功能，对于 WS 而言，JDK 同样也提供了很好的工具，就像下面这段代码那样：

```
<!-- lang: java -->
package demo.ws.soap_jdk;

import java.net.URL;
import javax.xml.namespace.QName;
import javax.xml.ws.Service;

public class DynamicClient {

    public static void main(String[] args) {
        try {
            URL wsdl = new URL("http://localhost:8080/ws/soap/hello?wsdl");
            QName serviceName = new QName("http://soap_jdk.ws.demo/", "HelloService");
            QName portName = new QName("http://soap_jdk.ws.demo/", "HelloServicePort");
            Service service = Service.create(wsdl, serviceName);

            HelloService helloService = service.getPort(portName, HelloService.class);
            String result = helloService.say("world");
            System.out.println(result);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

此时，只需在本地提供一个 HelloService 的接口，无需 client.jar，直接面向 WSDL 编程，只不过您需要分别定义出 serviceName 与 portName 这两个东西，最后才能调用 JDK 提供的 javax.xml.ws.Service 类生成 service 对象，它同样是一个工厂对象，通过该工厂对象获取我们需要的 HelloService 实例。貌似这种方式也不是特别动态，毕竟 HelloService 接口还是需要自行提供的。

3. 总结

通过本文，您可以了解到，不仅可以使用 JDK 发布 WS，也可以使用 JDK 调用 WS，这一切都是那么的简单而自然。但需要注意的是，这个特性是从 JDK 6 才开始提供的，如果您还在使用 JDK 5 或更低的版本，那就很遗憾了，您不得不使用以下工具来发布与调用 WS，它们分别是：

- JAX-WS RI：https://jax-ws.java.net/
- Axis：http://axis.apache.org/
- CXF：http://cxf.apache.org/

当然，发布与调用 WS 的工具不仅仅只有以上这些，而是它们是 Java 世界中最优秀的 WS 开源项目。

本文讲述的 WS 其实是一种 Java 规范，名为 JAX-WS（JSR-224），全称 Java API for XML-Based Web Services，可以将规范理解为官方定义的一系列接口。

JAX-WS 有一个官方实现，就是上面提到的 JAX-WS RI，它是 Oracle 公司提供的实现，而 Apache 旗下的 Axis 与 CXF 也同样实现了该规范。Axis 相对而言更加老牌一些，而 CXF 的前世就是 XFire，它是一款著名的 WS 框架，擅长与 Spring 集成。

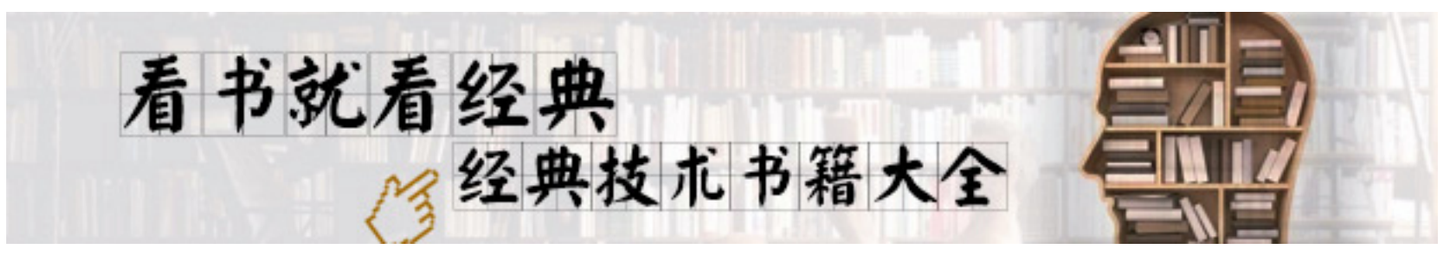
从本质上讲，JAX-WS 是基于 SOAP 的，而 SOAP 的全称是 Simple Object Access Protocol（简单对象访问协议），虽然名称里带有“简单”二字，其实并不简单，不相信您可以百度一下。

为了让 WS 的开发与使用变得更加简单、更加轻量级，于是出现了另一种风格的 WS，名为 JAX-RS（JSR-339），全称 Java API for RESTful Web Services，同样也是一种规范，同样也有若干实现，它们分别是：

- Jersey：https://jersey.java.net/
- Restlet：http://restlet.com/
- RESTEasy：http://resteasy.jboss.org/
- CXF：http://cxf.apache.org/

其中，Jersey 是 Oracle 官方提供的实现，Restlet 是最老牌的实现，RESTEasy 是 JBoss 公司提供的实现，CXF 是 Apache 提供的实现（上文已做介绍）。

可见，CXF 不仅用于开发基于 SOAP 的 WS，同样也适用于开发基于 REST 的 WS，这么好的框架我们怎能错过?



相关文章

- Java Web 应用安全
- 与Service实现双向通信（下）
- 与Service实现双向通信（中）
- 与Service实现双向通信（上）
- Spring4新特性（3）：Web开发的增强
- Hadoop教程(二)
- 跟开涛学SpringMVC（5）：处理器拦截器详解
- [JAVA · 初级]：偶识【正则表达式】
- 奇怪的Java题：为什么1000 == 1000返回为False，而100 == 100会返回为True?
- zookeeper入门系列：paxos协议

发表评论

Comment form

Name*

姓名

邮箱*

请填写邮箱

网站 (请以 http://开头)

请填写网站地址

评论内容*

请填写评论内容

(*) 表示必填项

还没有评论。

« epoll 浅析以及 nio 中的 Selector

怎样写一个 RefererFilter »

Search for:

Search

Search

-
- 本月热门
- 热门标签

0 图解 CMS 垃圾回收机制，你值得...

1 2018 年 Java 平台发布计划之新...

2 Java 异常进阶

3 通向架构师的道路（第一天）之 Apache ...

4 G1 垃圾收集器之对象分配过程

5 面试必问的 volatile，你了解多少？

6 通向架构师的道路（第二天）之 apache tom...

7 通向架构师的道路（第三天）之 apach...

8 通向架构师的道路（第四天）之 Tomc...

最新评论

- Re: 成小胖学习 ActiveMQ · ...
  像是在讲故事，过程很不错 yang
- Re: 记一次 Spring Maven 打包...
  不好意思 第一次评论需要审核 怕有爬虫机器人制造垃圾评论 唐小娟
- Re: MySQL 死锁与日志二三事
  补充：update 走的是二级索引，表中有自增主键； jianhaiqing
- Re: MySQL 死锁与日志二三事
  update 假如走索引的话，索引到的数据很多，会一下锁所有行么？ 实际过程来看是不会的，过程是怎么... jianhaiqing
- Re: MySQL 死锁与日志二三事
  脚本很有用，感谢您的分享 jianhaiqing
- Re: Java String 对 null 对象...
  确实null是个问题，分开处理是最直接的。如果我们写对象去封装null->Null 这样更像... 沙漠的模样
- Re: JVM类加载的那些事
  常量在编译期间会进行替换, 输出Consts.A是会输出100的, 但是类确实没有加载, 请作者更正 Mr.Z
- Re: MySQL 死锁与日志二三事
  case1 中，假如能给出show create table 和 update sql 语句以及前... jianhaiqing

关于ImportNew

ImportNew 专注于 Java 技术分享。于2012年11月11日 11:11正式上线。是的，这是一个很特别的时刻 :)

ImportNew 由两个 Java 关键字 import 和 new 组成，意指：Java 开发者学习新知识的网站。 import 可认为是学习和吸收， new 则可认为是新知识、新技术圈子和新朋友……

联系我们

Email：ImportNew.com@gmail.com
新浪微博：@ImportNew
推荐微信号

ImportNew 安卓应用频道 Linux爱好者

反馈建议：ImportNew.com@gmail.com
广告与商务合作QQ：2302462408

推荐关注

小组 – 好的话题、有启发的回复、值得信赖的圈子
头条 – 写了文章？看干货？去头条！
相亲 – 为IT单身男女服务的征婚传播平台
资源 – 优秀的工具资源导航
翻译 – 活跃 & 专业的翻译小组
博客 – 国内外的精选博客文章
设计 – UI,网页，交互和用户体验
前端 – JavaScript, HTML5, CSS
安卓 – 专注Android技术分享
iOS – 专注iOS技术分享
Java – 专注Java技术分享
Python – 专注Python技术分享