JAX-WS 简明教程

版本：20081218

陈华

clinker@163.com

2008 年 12 月

目录

1. 概述

1.1. 文档内容

本文描述了用 JAX-WS 2.x 如何创建 Web Service、如何创建 Web Service 客户端。

1.2. JAX-WS 概述

JAX-WS 2.0 的全称为 Java API for XML-Based Web services (JAX-WS) 2.0。JAX-WS 2.0 是对 JAX-RPC 1.0 规范的扩展，是 JAX-RPC 1.1 的后续版本， JAX-RPC 2.0 标准发布不久后便被重新命名为 JAX-WS 2.0。 JAX-WS 2.0 是面向 Java 5 的开发 Web services 的最新编程标准，它提供了新的编程模型和对以往的 JAX-RPC 方式的 Web services 进行了增强。

JAX-WS2.0 (JSR 224)是 Sun 新的 web services 协议栈，是一个完全基于标准的实现。在 binding 层，使用的是 the Java Architecture for XML Binding (JAXB, JSR 222)，在 parsing 层，使用的是 the Streaming API for XML (StAX, JSR 173)，同时它还完全支持 schema 规范。

1.2.1. JAX-WS 2.1 特性

- 支持 SOAP 1.1（默认）、1.2
- 支持 XML/HTTP Binding
- 支持 WS-Addressing
- 支持 document/literal 样式
- 支持 WS-I Basic Profile 1.1
- 支持消息传输优化机制（Message Transmission Optimization Mechanism，MTOM）

1.3. 缩略语

- SEI

  Service Endpoint Interface。

- JAX

Java API for XML Web Services。

- JAX-WS RI

  JAX-WS Reference Implementation。

- SAAJ

  SOAP with Attachments API for Java。

1.4. 参考资料

- 《利用 WAS 6.1 WebService 功能部件包开发 JAX-WS 2.0 Web services》，http://www.ibm.com/developerworks/cn/webservices/0801_kujg/
- Metro Users Guide

  https://jax-ws.dev.java.net/guide/index.html

- Java API for XML Web Services (JAX-WS) JAX-WS RI Extensions

  https://jax-ws.dev.java.net/nonav/2.1.5/docs/ri-features.html

2. 软件版本

2.1. JDK

java version "1.6.0_x"。

2.2. JAX-WS

JAX-WS RI 2.1.1 in JDK 6。

3. 创建 Web Service

JAX-WS 2.0 有两种开发过程：自顶向下和自底向上。自顶向下方式指通过一个 WSDL 文件来创建 Web Service，自底向上是从 Java 类出发创建 Web Service。两种开发过程最终形成的文件包括：

- SEI。一个 SEI 对应 WSDL 中 Web Service 的一个 port，在 Java 中是一个 Java 接口。
- SEI 实现类。
- WSDL 和 XSD 文件。

3.1. 从 Java 开始

3.1.1. 开发步骤

开发 Web Service 分为两个步骤：

1) 声明某个类为@WebService，即将它声明为 SEI 实现类，然后对需要暴露的方法标注为@WebMethod。

2）运行 wsgen 命令生成其他所需文件。

3.1.2. SEI 实现类代码

最简单（所有关于 Web Service 的设置均使用 JAX-WS 默认值）的 SEI 实现类代码如下：

```
package ws.server.fromjava;

import javax.jws.WebMethod;
import javax.jws.WebService;

/**
 * 演示从 Java 代码开始生成 Web Service。
 *
 */

@WebService
public class Hello {

    @WebMethod
    public String say(String name, int age) {
        return String.format("Hello,%s,you are %d years old", name, age);
    }
}
```

3.1.3. 运行 wsgen

wsgen ws.server.fromjava.Hello -wsdl -s src -d bin -r wsdl

src 是生成的源代码的存放路径。

bin 是生成的 class 文件的存放路径。

-r 后面的 wsdl 是生成的 WSDL、XSD 文件的存放路径。

3.1.4. 生成的 Java 代码

3.1.4.1. Say.java

```
package ws.server.fromjava.jaxws;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;

@XmlRootElement(name = "say", namespace = "http://fromjava.server.ws/")
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "say", namespace = "http://fromjava.server.ws/",
propOrder = {
        "arg0", "arg1" })
public class Say {

    @XmlElement(name = "arg0", namespace = "")
    private String arg0;
    @XmlElement(name = "arg1", namespace = "")
    private int arg1;

    /**
     *
     * @return returns String
     */
    public String getArg0() {
        return this.arg0;
    }

    /**
```

```
     *
     * @param arg0
     *          the value for the arg0 property
     */
    public void setArg0(String arg0) {
        this.arg0 = arg0;
    }

    /**
     *
     * @return returns int
     */
    public int getArg1() {
        return this.arg1;
    }

    /**
     *
     * @param arg1
     *          the value for the arg1 property
     */
    public void setArg1(int arg1) {
        this.arg1 = arg1;
    }

}
```

3.1.4.2. SayResponse.java

```
package ws.server.fromjava.jaxws;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;

@XmlRootElement(name = "sayResponse", namespace = "http://fromjava.server.ws/")
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "sayResponse", namespace = "http://fromjava.server.ws/")
public class SayResponse {

    @XmlElement(name = "return", namespace = "")
    private String _return;
```

```
    /**
     *
     * @return returns String
     */
    public String getReturn() {
        return this._return;
    }

    /**
     *
     * @param _return
     *                  the value for the _return property
     */
    public void setReturn(String _return) {
        this._return = _return;
    }

}
```

3.1.5. 生成的 WSDL 和 XSD

3.1.5.1. WSDL

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <!--
        Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is
        JAX-WS RI 2.1.1 in JDK 6.
    -->
<definitions targetNamespace="http://fromjava.server.ws/"
    name="HelloService"                         xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://fromjava.server.ws/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <types>
        <xsd:schema>
            <xsd:import namespace="http://fromjava.server.ws/"
                schemaLocation="HelloService_schema1.xsd" />
        </xsd:schema>
    </types>
    <message name="say">
        <part name="parameters" element="tns:say" />
    </message>
    <message name="sayResponse">
```

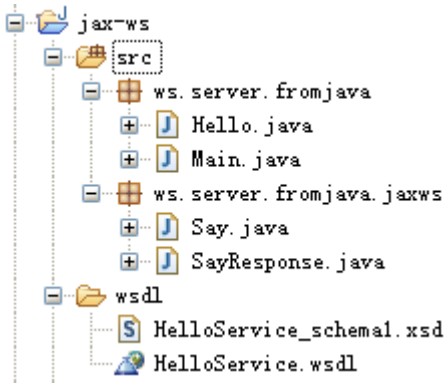
```
        <part name="parameters" element="tns:sayResponse" />
    </message>
    <portType name="Hello">
        <operation name="say">
            <input message="tns:say" />
            <output message="tns:sayResponse" />
        </operation>
    </portType>
    <binding name="HelloPortBinding" type="tns:Hello">
        <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="HelloPort" binding="tns:HelloPortBinding">
            <soap:address location="REPLACE_WITH_ACTUAL_URL" />
        </port>
    </service>
</definitions>
```

3.1.5.2. XSD

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://fromjava.server.ws/"
    xmlns:tns="http://fromjava.server.ws/" xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="say" type="tns:say" />

    <xs:element name="sayResponse" type="tns:sayResponse" />

    <xs:complexType name="say">
        <xs:sequence>
            <xs:element name="arg0" type="xs:string" minOccurs="0" />
            <xs:element name="arg1" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
```

```
    <xs:complexType name="sayResponse">
        <xs:sequence>
            <xs:element name="return" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

3.1.6. 目录结构



3.2. 从 WSDL 开始

3.2.1. 开发步骤

1) 运行 wsimport 生成 Java 代码；

2) 创建 SEI 实现类。

3.2.2. 运行 wsimport

使用上面生成的 WSDL 来反向生成 Java 代码。生成后，不需要进行改动。

wsimport -keep -d bin -s src wsdl/HelloService.wsdl

src 是生成的源代码的存放路径。

bin 是生成的 class 文件的存放路径。

wsdl/HelloService.wsdl 是 Web Service WSDL 的路径。

3.2.3. 生成的 Java 代码

3.2.3.1. Hello.java

```
package ws.server.fromjava;

import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebResult;
import javax.jws.WebService;
import javax.xml.bind.annotation.XmlSeeAlso;
import javax.xml.ws.RequestWrapper;
import javax.xml.ws.ResponseWrapper;

/**
 * This class was generated by the JAX-WS RI. JAX-WS RI 2.1.1 in JDK 6 Generated
 * source version: 2.1
 *
 */
@WebService(name = "Hello", targetNamespace = "http://fromjava.server.ws/")
@XmlSeeAlso( { ObjectFactory.class })
public interface Hello {

    /**
     *
     * @param arg1
     * @param arg0
     * @return returns java.lang.String
     */
    @WebMethod
    @WebResult(targetNamespace = "")
    @RequestWrapper(localName = "say", targetNamespace = "http://fromjava.server.ws/",
className = "ws.server.fromjava.Say")
    @ResponseWrapper(localName = "sayResponse", targetNamespace =
"http://fromjava.server.ws/", className = "ws.server.fromjava.SayResponse")
    public String say(
            @WebParam(name = "arg0", targetNamespace = "") String arg0,
            @WebParam(name = "arg1", targetNamespace = "") int arg1);

}
```

3.2.3.2. HelloService.java

```
package ws.server.fromjava;
```

```
import java.net.MalformedURLException;
import java.net.URL;
import javax.xml.namespace.QName;
import javax.xml.ws.Service;
import javax.xml.ws.WebEndpoint;
import javax.xml.ws.WebServiceClient;
import javax.xml.ws.WebServiceFeature;

/**
 * This class was generated by the JAX-WS RI. JAX-WS RI 2.1.1 in JDK 6 Generated
 * source version: 2.1
 *
 */
@WebServiceClient(name = "HelloService", targetNamespace = "http://fromjava.server.ws/",
wsdlLocation = "file:/D:/ jax-ws-server-fromwsdl/wsdl/HelloService.wsdl")
public class HelloService extends Service {

    private final static URL HELLOSERVICE_WSDL_LOCATION;

    static {
        URL url = null;
        try {
            url = new URL(

"file:/D:/mydocs/workspace/jax-ws-server-fromwsdl/wsdl/HelloService.wsdl");
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
        HELLOSERVICE_WSDL_LOCATION = url;
    }

    public HelloService(URL wsdlLocation, QName serviceName) {
        super(wsdlLocation, serviceName);
    }

    public HelloService() {
        super(HELLOSERVICE_WSDL_LOCATION, new QName(
                "http://fromjava.server.ws/", "HelloService"));
    }

    /**
     *
     * @return returns Hello
```

```
     */
    @WebEndpoint(name = "HelloPort")
    public Hello getHelloPort() {
        return (Hello) super.getPort(new QName("http://fromjava.server.ws/",
                "HelloPort"), Hello.class);
    }

    /**
     *
     * @param features
     *                  A list of {@link javax.xml.ws.WebServiceFeature} to configure
     *                  on the proxy. Supported features not in the
     *                  <code>features</code> parameter will have their default
     *                  values.
     * @return returns Hello
     */
    @WebEndpoint(name = "HelloPort")
    public Hello getHelloPort(WebServiceFeature... features) {
        return (Hello) super.getPort(new QName("http://fromjava.server.ws/",
                "HelloPort"), Hello.class, features);
    }

}
```

3.2.3.3. ObjectFactory.java

```
package ws.server.fromjava;

import javax.xml.bind.JAXBElement;
import javax.xml.bind.annotation.XmlElementDecl;
import javax.xml.bind.annotation.XmlRegistry;
import javax.xml.namespace.QName;

/**
 * This object contains factory methods for each Java content interface and Java
 * element interface generated in the ws.server.fromjava package.
 * <p>
 * An ObjectFactory allows you to programatically construct new instances of the
 * Java representation for XML content. The Java representation of XML content
 * can consist of schema derived interfaces and classes representing the binding
 * of schema type definitions, element declarations and model groups. Factory
 * methods for each of these are provided in this class.
 *
```

```
 */
@XmlRegistry
public class ObjectFactory {

    private final static QName _Say_QNAME = new QName(
            "http://fromjava.server.ws/", "say");
    private final static QName _SayResponse_QNAME = new QName(
            "http://fromjava.server.ws/", "sayResponse");

    /**
     * Create a new ObjectFactory that can be used to create new instances of
     * schema derived classes for package: ws.server.fromjava
     *
     */
    public ObjectFactory() {
    }

    /**
     * Create an instance of {@link Say }
     *
     */
    public Say createSay() {
        return new Say();
    }

    /**
     * Create an instance of {@link SayResponse }
     *
     */
    public SayResponse createSayResponse() {
        return new SayResponse();
    }

    /**
     * Create an instance of {@link JAXBElement }{@code <}{@link Say }{@code >}
     *
     */
    @XmlElementDecl(namespace = "http://fromjava.server.ws/", name = "say")
    public JAXBElement<Say> createSay(Say value) {
        return new JAXBElement<Say>(_Say_QNAME, Say.class, null, value);
    }

    /**
     * Create an instance of {@link JAXBElement }{@code <}{@link SayResponse }
```

```
     * {@code >}
     *
     */
    @XmlElementDecl(namespace = "http://fromjava.server.ws/", name = "sayResponse")
    public JAXBElement<SayResponse> createSayResponse(SayResponse value) {
        return new JAXBElement<SayResponse>(_SayResponse_QNAME,
                SayResponse.class, null, value);
    }

}
```

3.2.3.4. package-info.java

```
@javax.xml.bind.annotation.XmlSchema(namespace = "http://fromjava.server.ws/")
package ws.server.fromjava;
```

3.2.3.5. Say.java

```
package ws.server.fromjava;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlType;

/**
 * <p>
 * Java class for say complex type.
 *
 * <p>
 * The following schema fragment specifies the expected content contained within
 * this class.
 *
 * <pre>
 * &lt;complexType name=&quot;say&quot;&gt;
 *   &lt;complexContent&gt;
 *     &lt;restriction base=&quot;{http://www.w3.org/2001/XMLSchema}anyType&quot;&gt;
 *       &lt;sequence&gt;
 *         &lt;element name=&quot;arg0&quot; type=&quot;{http://www.w3.org/2001/XMLSchema}string&quot; minOccurs=&quot;0&quot;/&gt;
 *         &lt;element name=&quot;arg1&quot; type=&quot;{http://www.w3.org/2001/XMLSchema}int&quot;/&gt;
 *       &lt;/sequence&gt;
```

```
 *         &lt;/restriction&gt;
 *     &lt;/complexContent&gt;
 * &lt;/complexType&gt;
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "say", propOrder = { "arg0", "arg1" })
public class Say {

    protected String arg0;
    protected int arg1;

    /**
     * Gets the value of the arg0 property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getArg0() {
        return arg0;
    }

    /**
     * Sets the value of the arg0 property.
     *
     * @param value
     *                  allowed object is {@link String }
     *
     */
    public void setArg0(String value) {
        this.arg0 = value;
    }

    /**
     * Gets the value of the arg1 property.
     *
     */
    public int getArg1() {
        return arg1;
    }

    /**
```

```
     * Sets the value of the arg1 property.
     *
     */
    public void setArg1(int value) {
        this.arg1 = value;
    }

}
```

3.2.3.6. SayResponse.java

```
package ws.server.fromjava;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;

/**
 * <p>
 * Java class for sayResponse complex type.
 *
 * <p>
 * The following schema fragment specifies the expected content contained within
 * this class.
 *
 * <pre>
 * &lt;complexType name=&quot;sayResponse&quot;&gt;
 *     &lt;complexContent&gt;
 *         &lt;restriction base=&quot;{http://www.w3.org/2001/XMLSchema}anyType&quot;&gt;
 *             &lt;sequence&gt;
 *                                                 &lt;element        name=&quot;return&quot;
type=&quot;{http://www.w3.org/2001/XMLSchema}string&quot;
minOccurs=&quot;0&quot;/&gt;
 *             &lt;/sequence&gt;
 *         &lt;/restriction&gt;
 *     &lt;/complexContent&gt;
 * &lt;/complexType&gt;
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "sayResponse", propOrder = { "_return" })
```

```
public class SayResponse {

    @XmlElement(name = "return")
    protected String _return;

    /**
     * Gets the value of the return property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getReturn() {
        return _return;
    }

    /**
     * Sets the value of the return property.
     *
     * @param value
     *                  allowed object is {@link String }
     *
     */
    public void setReturn(String value) {
        this._return = value;
    }

}
```

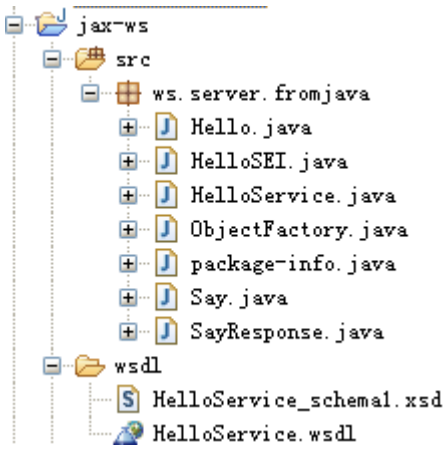
3.2.4. 创建 SEI 类

实现 Hello 接口，并添加@WebService 注释，并至少为@WebService 添加 endpointInterface 属性。

```
package ws.server.fromjava;

import javax.jws.WebService;

@WebService(serviceName = "HelloService", portName = "HelloPort", endpointInterface = "ws.server.fromjava.Hello", targetNamespace = "http://fromjava.server.ws/")
public class HelloSEI implements Hello {

    @Override
    public String say(String arg0, int arg1) {
        return null;
```

```
    }

}
```

3.2.5. 目录结构



3.3. 发布 Web Service

3.3.1. 在应用程序中发布

JDK6 提供了发布 Web Service 的简便方法：

```
Endpoint.publish("http://localhost:8080/HelloService", new Hello());
```

如果是从 WSDL 生成的 Web Service，则为，

```
Endpoint.publish("http://localhost:8080/HelloService", new HelloSEI());
```

3.3.2. 在 Web 应用程序中发布

利用 SUN 公司提供的辅助包，可以将 Web Service 发布为 Web 应用程序。

3.3.2.1. 依赖包

- activation.jar
- FastInfoset.jar
- http.jar
- jaxb-impl.jar

- jaxws-rt.jar
- mimepull.jar
- resolver.jar
- stax-ex.jar
- streambuffer.jar

3.3.2.2. 发布步骤

1) 修改 web.xml，添加，

```
<listener>
    <listener-class>
        com.sun.xml.ws.transport.http.servlet.WSServletContextListener</listener-class>
</listener>
<servlet>
    <servlet-name>Hello</servlet-name>
    <servlet-class>com.sun.xml.ws.transport.http.servlet.WSServlet</servlet-class>
    <load-on-startup>2</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Hello</servlet-name>
    <url-pattern>/HelloService</url-pattern>
</servlet-mapping>
```

为每一个 WebService 声明一个 com.sun.xml.ws.transport.http.servlet.WSServlet。

2) 在 WEB-INF 目录下新建 sun-jaxws.xml 文件，内容如下，

```
<?xml version="1.0" encoding="UTF-8"?>
<endpoints version="2.0"
    xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime">

    <endpoint implementation="ws.server.fromjava.Hello" name="Hello"
        url-pattern="/HelloService" />

</endpoints>
```

将 ws.server.fromjava.Hello 声明为 Web Service。

如果是从 WSDL 生成的 Web Service，则为，

```
<?xml version="1.0" encoding="UTF-8"?>
<endpoints version="2.0"
        xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime">

        <endpoint implementation="ws.server.fromjava.HelloSEI" name="Hello"
                url-pattern="/HelloService" />

</endpoints>
```

4. 创建 Web Service 客户端

客户端开发的通常过程是从已有的 WSDL 出发，创建辅助类 JAXB 对象和 Service 代理类，然后基于这些类开发自己的客户端应用。

4.1. 同步调用方式的客户端

4.1.1. 开发步骤

1) 运行 wsimport 命令，生成客户端代码；

2) 修改生成的 Java 代码中的 WSDL 地址。

4.1.2. 运行 wsimport

wsimport -keep -d bin -s src wsdl/HelloService.wsdl

src 是生成的源代码的存放路径。

bin 是生成的 class 文件的存放路径。

wsdl/HelloService.wsdl 是 Web Service WSDL 的路径。

4.1.3. 修改生成的 Java 代码

编辑 HelloService.java，将

```
@WebServiceClient(name = "HelloService", targetNamespace = "http://fromjava.server.ws/",
wsdlLocation = "file:/D: /jax-ws-client/wsdl/HelloService.wsdl")
public class HelloService extends Service {

    private final static URL HELLOSERVICE_WSDL_LOCATION;

    static {
        URL url = null;
        try {
            url = new URL(
                    "file:/D:/mydocs/workspace/jax-ws-client/wsdl/HelloService.wsdl");
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
        HELLOSERVICE_WSDL_LOCATION = url;
    }
```

```
    public HelloService(URL wsdlLocation, QName serviceName) {
        super(wsdlLocation, serviceName);
    }

    public HelloService() {
        super(HELLOSERVICE_WSDL_LOCATION, new QName(
                "http://fromjava.server.ws/", "HelloService"));
    }
```

修改为，

```
@WebServiceClient(name = "HelloService", targetNamespace = "http://fromjava.server.ws/")
public class HelloService extends Service {

    public HelloService(URL wsdlLocation) {
        super(wsdlLocation, new QName("http://fromjava.server.ws/",
                "HelloService"));
    }
```

即删除 WSDL 具体地址。

4.1.4. 生成并修改后的 Java 代码

4.1.4.1. Hello.java

```
package ws.server.fromjava;

import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebResult;
import javax.jws.WebService;
import javax.xml.bind.annotation.XmlSeeAlso;
import javax.xml.ws.RequestWrapper;
import javax.xml.ws.ResponseWrapper;

/**
 * This class was generated by the JAX-WS RI. JAX-WS RI 2.1.1 in JDK 6 Generated
 * source version: 2.1
 *
 */
@WebService(name = "Hello", targetNamespace = "http://fromjava.server.ws/")
```

```
@XmlSeeAlso( { ObjectFactory.class })
public interface Hello {

    /**
     *
     * @param arg1
     * @param arg0
     * @return returns java.lang.String
     */
    @WebMethod
    @WebResult(targetNamespace = "")
    @RequestWrapper(localName = "say", targetNamespace = "http://fromjava.server.ws/",
className = "ws.server.fromjava.Say")
    @ResponseWrapper(localName = "sayResponse", targetNamespace =
"http://fromjava.server.ws/", className = "ws.server.fromjava.SayResponse")
    public String say(
            @WebParam(name = "arg0", targetNamespace = "") String arg0,
            @WebParam(name = "arg1", targetNamespace = "") int arg1);

}
```

4.1.4.2. HelloService.java

```
package ws.server.fromjava;

import java.net.URL;

import javax.xml.namespace.QName;
import javax.xml.ws.Service;
import javax.xml.ws.WebEndpoint;
import javax.xml.ws.WebServiceClient;
import javax.xml.ws.WebServiceFeature;

/**
 * This class was generated by the JAX-WS RI. JAX-WS RI 2.1.1 in JDK 6 Generated
 * source version: 2.1
 *
 */
@WebServiceClient(name = "HelloService", targetNamespace = "http://fromjava.server.ws/")
public class HelloService extends Service {

    public HelloService(URL wsdlLocation) {
        super(wsdlLocation, new QName("http://fromjava.server.ws/",
                "HelloService"));
```

```
    }

    /**
     *
     * @return returns Hello
     */
    @WebEndpoint(name = "HelloPort")
    public Hello getHelloPort() {
        return (Hello) super.getPort(new QName("http://fromjava.server.ws/",
                "HelloPort"), Hello.class);
    }

    /**
     *
     * @param features
     *                  A list of {@link javax.xml.ws.WebServiceFeature} to configure
     *                  on the proxy. Supported features not in the
     *                  <code>features</code> parameter will have their default
     *                  values.
     * @return returns Hello
     */
    @WebEndpoint(name = "HelloPort")
    public Hello getHelloPort(WebServiceFeature... features) {
        return (Hello) super.getPort(new QName("http://fromjava.server.ws/",
                "HelloPort"), Hello.class, features);
    }

}
```

4.1.4.3. ObjectFactory.java

```
package ws.server.fromjava;

import javax.xml.bind.JAXBElement;
import javax.xml.bind.annotation.XmlElementDecl;
import javax.xml.bind.annotation.XmlRegistry;
import javax.xml.namespace.QName;

/**
 * This object contains factory methods for each Java content interface and Java
 * element interface generated in the ws.server.fromjava package.
 * <p>
 * An ObjectFactory allows you to programatically construct new instances of the
 * Java representation for XML content. The Java representation of XML content
```

```
 * can consist of schema derived interfaces and classes representing the binding
 * of schema type definitions, element declarations and model groups. Factory
 * methods for each of these are provided in this class.
 *
 */
@XmlRegistry
public class ObjectFactory {

    private final static QName _Say_QNAME = new QName(
            "http://fromjava.server.ws/", "say");
    private final static QName _SayResponse_QNAME = new QName(
            "http://fromjava.server.ws/", "sayResponse");

    /**
     * Create a new ObjectFactory that can be used to create new instances of
     * schema derived classes for package: ws.server.fromjava
     *
     */
    public ObjectFactory() {
    }

    /**
     * Create an instance of {@link Say }
     *
     */
    public Say createSay() {
        return new Say();
    }

    /**
     * Create an instance of {@link SayResponse }
     *
     */
    public SayResponse createSayResponse() {
        return new SayResponse();
    }

    /**
     * Create an instance of {@link JAXBElement }{@code <}{@link Say }{@code >}
     *
     */
    @XmlElementDecl(namespace = "http://fromjava.server.ws/", name = "say")
    public JAXBElement<Say> createSay(Say value) {
        return new JAXBElement<Say>(_Say_QNAME, Say.class, null, value);
```

```
    }

    /**
     * Create an instance of {@link JAXBElement }{@code <}{@link SayResponse }
     * {@code >}
     *
     */
    @XmlElementDecl(namespace = "http://fromjava.server.ws/", name = "sayResponse")
    public JAXBElement<SayResponse> createSayResponse(SayResponse value) {
        return new JAXBElement<SayResponse>(_SayResponse_QNAME,
                SayResponse.class, null, value);
    }

}
```

4.1.4.4. package-info.java

```
@javax.xml.bind.annotation.XmlSchema(namespace = "http://fromjava.server.ws/")
package ws.server.fromjava;
```

4.1.4.5. Say.java

```
package ws.server.fromjava;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlType;

/**
 * <p>
 * Java class for say complex type.
 *
 * <p>
 * The following schema fragment specifies the expected content contained within
 * this class.
 *
 * <pre>
 * &lt;complexType name=&quot;say&quot;&gt;
 *     &lt;complexContent&gt;
 *         &lt;restriction base=&quot;{http://www.w3.org/2001/XMLSchema}anyType&quot;&gt;
 *             &lt;sequence&gt;
 *                                                 &lt;element        name=&quot;arg0&quot;
type=&quot;{http://www.w3.org/2001/XMLSchema}string&quot;
```

```
minOccurs=&quot;0&quot;/&gt;
 *                                                    &lt;element        name=&quot;arg1&quot;
type=&quot;{http://www.w3.org/2001/XMLSchema}int&quot;/&gt;
 *             &lt;/sequence&gt;
 *         &lt;/restriction&gt;
 *     &lt;/complexContent&gt;
 * &lt;/complexType&gt;
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "say", propOrder = { "arg0", "arg1" })
public class Say {

    protected String arg0;
    protected int arg1;

    /**
     * Gets the value of the arg0 property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getArg0() {
        return arg0;
    }

    /**
     * Sets the value of the arg0 property.
     *
     * @param value
     *                  allowed object is {@link String }
     *
     */
    public void setArg0(String value) {
        this.arg0 = value;
    }

    /**
     * Gets the value of the arg1 property.
     *
     */
    public int getArg1() {
```

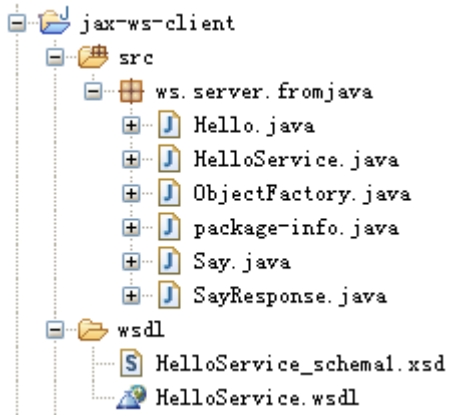
```
        return arg1;
    }

    /**
     * Sets the value of the arg1 property.
     *
     */
    public void setArg1(int value) {
        this.arg1 = value;
    }

}
```

4.1.4.6. SayResponse.java

```
package ws.server.fromjava;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;

/**
 * <p>
 * Java class for sayResponse complex type.
 *
 * <p>
 * The following schema fragment specifies the expected content contained within
 * this class.
 *
 * <pre>
 * &lt;complexType name=&quot;sayResponse&quot;&gt;
 *     &lt;complexContent&gt;
 *         &lt;restriction base=&quot;{http://www.w3.org/2001/XMLSchema}anyType&quot;&gt;
 *             &lt;sequence&gt;
 *                                                 &lt;element        name=&quot;return&quot;
type=&quot;{http://www.w3.org/2001/XMLSchema}string&quot;
minOccurs=&quot;0&quot;/&gt;
 *             &lt;/sequence&gt;
 *         &lt;/restriction&gt;
 *     &lt;/complexContent&gt;
 * &lt;/complexType&gt;
 * </pre>
 *
```

```
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "sayResponse", propOrder = { "_return" })
public class SayResponse {

    @XmlElement(name = "return")
    protected String _return;

    /**
     * Gets the value of the return property.
     *
     * @return possible object is {@link String }
     *
     */
    public String getReturn() {
        return _return;
    }

    /**
     * Sets the value of the return property.
     *
     * @param value
     *                  allowed object is {@link String }
     *
     */
    public void setReturn(String value) {
        this._return = value;
    }

}
```

4.1.5. 目录结构



4.1.6. 调用 Web Service

```
// Web Service  地址
URL url = new URL("http://localhost:8080/HelloService?wsdl");
//  创建客户端
HelloService service = new HelloService(url);
//  调用 Web Service 方法
String resp = service.getHelloPort().say("Jack", 28);
```

5. SOAP headers

When the WSDL you are compiling specifies that some parts of a message are bound to SOAP headers, wsimport generates the right stuff (@WebParam(header=true)), so you can pass headers as arguments to the method invocation. When starting from Java, you can use this same annotation to indicate that some parameters be sent as headers.

That said, there are many WSDLs out there that do not specify SOAP headers explicitly, yet the protocol still requires such headers to be sent, so the JAX-WS RI offers convenient ways for you to send/receive additional headers at runtime.

5.1. 非标准方法

所谓非标准方法，即使用的方法不符合 JAX-WS API 规范、但被 JAX-WS RI 支持。

5.1.1. 客户端添加 SOAP headers

The portable way of doing this is that you creaate a SOAPHandler and mess with SAAJ, but the RI provides a better way of doing this.

When you create a proxy or dispatch object, they implement BindingProvider interface. When you use the JAX-WS RI, you can downcast to WSBindingProvider which defines a few more methods provided only by the JAX-WS RI.

This interface lets you set an arbitrary number of Header object, each representing a SOAP header. You can implement it on your own if you want, but most likely you'd use one of the factory methods defined on Headers class to create one.

示例代码如下：

```
import com.sun.xml.internal.ws.api.message.Headers;
import com.sun.xml.internal.ws.developer.WSBindingProvider;
import javax.xml.namespace.QName;

// Web Service  地址
URL url = new URL("http://localhost:8080/HelloService?wsdl");
//  创建客户端
HelloService service = new HelloService(url);
Hello helloPort = service.getHelloPort();
WSBindingProvider bp = (WSBindingProvider) helloPort;
bp.setOutboundHeaders(
// simple string value as a header, like
// <simpleHeader>stringValue</simpleHeader>
        Headers.create(new QName("userid"), "123456"), Headers.create(
                new QName("password"),
```

```
                "EDD67C5E51C61C0083"), Headers.create(
                new QName("timeStamp"), "200801218111008"));
        // 调用 Web Service 方法
        String resp = helloPort.say("Jack", 28);
```

5.1.2. 访问 SOAP headers

5.1.2.1. 服务器端

Server can access all the SOAP headers of the incoming messages by using the JAXWSProperties#INBOUND_HEADER_LIST_PROPERTY property like this: Server can access all the SOAP headers of the incoming messages by using the JAXWSProperties#INBOUND_HEADER_LIST_PROPERTY property like this: Server can access all the SOAP headers of the incoming messages by using the JAXWSProperties#INBOUND_HEADER_LIST_PROPERTY property like this:

```
@WebService
public class FooService {
    @Resource
    WebServiceContext context;

    @WebMethod
    public void sayHelloTo(String name) {
        HeaderList hl = context.getMessageContext().get(
                JAXWSProperties.INBOUND_HEADER_LIST_PROPERTY);
        Header h = hl.get(MYHEADER);
    }

    private static final QName MYHEADER = new QName("myNsUri",
            "myHeader");
    }
}
```

5.1.2.2. 客户端

Clients can similarly access all the SOAP headers of the incoming messages:

```
HelloPort port = helloService.getHelloPort();  // or something like that...
port.sayHelloTo("duke");
HeaderList hl = port.getResponseContext().get(JAXWSProperties.INBOUND_HEADER_LIST_PROPERTY);
Header h = hl.get(MYHEADER);
```

See the Header interface for more details about how to access the header values.

6. 附录

6.1. 常用命令简介

6.1.1. wsgen

Usage:wsgen [options] < SEI >

主要选项：

- -d 指定生成的 class 文件的位置。
- -s 指定生成的 Java source 文件的位置。
- -r 指定生成的 resources 文件的位置。如 WSDL、XSD 的位置。

-wsdl，-servicename，-portname 三个参数指定生成的 WSDL 文件中的 service 和 port 的名称。

注意：这里的 SEI 是一个 endpoint implementation class，而不是一个接口。必须先写好一个 endpoint 的实现类，该类中用@WebService 声明好 Web Service，再将它编译成 class 文件，才能提供给 wsgen 使用。

6.1.2. wsimport

Usage: wsimport [options] <WSDL_URI>

主要选项：

- -d 指定生成的 class 文件的位置。
- -s 指定生成的 Java source 文件的位置。
- -wsdllocation 指定生成的 Java source 中@WebService.WSDLLocation 和@WebServiceClient. wsdllocation 的值