JAX-WS : wsgen tool example

By mkyong | December 31, 2010 | Updated : August 29, 2012 | Viewed : 224,830 | +252 pv/w

The `wsgen` tool is used to parse an existing web service implementation class and generates required files (JAX-WS portable artifacts) for web service deployment. This `wsgen` tool is available in `$JDK/bin` folder.

Use cases

2 common use cases for wsgen tool :

1. Generates JAX-WS portable artifacts (Java files) for web service deployment.
2. Generates WSDL and xsd files, for testing or web service client development.

Let's see a web service implementation class, quite simple, just a method to return a string.

File : ServerInfo.java

```
package com.mkyong.ws;

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

@WebService
public class ServerInfo{

    @WebMethod
    public String getIpAddress() {

        return "10.10.10.10";

    }

}
```

1. Generates JAX-WS portable artifacts (Java files)

To generate all the JAX-WS portable artifacts for above web service implementation class (`ServerInfo.java`), use following command :

Command : wsgen usage

```
D:\>wsgen -verbose -keep -cp . com.mkyong.ws.ServerInfo

Note:   ap round: 1
[ProcessedMethods Class: com.mkyong.ws.ServerInfo]
[should process method: getIpAddress hasWebMethods: true ]
[endpointReferencesInterface: false]
[declaring class has WebSevice: true]
[returning: true]
[WrapperGen - method: getIpAddress()]
[method.getDeclaringType(): com.mkyong.ws.ServerInfo]
[requestWrapper: com.mkyong.ws.jaxws.GetIpAddress]
[ProcessedMethods Class: java.lang.Object]
com\mkyong\ws\jaxws\GetIpAddress.java
com\mkyong\ws\jaxws\GetIpAddressResponse.java
Note:   ap round: 2
```

In this case, it generated four files :

1. com\mkyong\ws\jaxws\GetIpAddress.java
2. com\mkyong\ws\jaxws\GetIpAddress.class
3. com\mkyong\ws\jaxws\GetIpAddressResponse.java

4. com\mkyong\ws\jaxws\GetIpAddressResponse.class

File : GetIpAddress.java

```
package com.mkyong.ws.jaxws;

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

@XmlRootElement(name = "getIpAddress", namespace = "http://ws.mkyong.com/")
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "getIpAddress", namespace = "http://ws.mkyong.com/")
public class GetIpAddress {

}
```

File : GetIpAddressResponse.java

```
package com.mkyong.ws.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 = "getIpAddressResponse", namespace = "http://ws.mkyong.com/")
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "getIpAddressResponse", namespace = "http://ws.mkyong.com/")
public class GetIpAddressResponse {

    @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;
    }

}
```

2. Genarates WSDL and xsd

To generate WSDL and xsd files for above web service implementation class (`ServerInfo.java`), add an extra **-wsdl** in the `wsgen` command :

Command : wsgen usage

```
D:\>wsgen -verbose -keep -cp . com.mkyong.ws.ServerInfo -wsdl

Note:   ap round: 1
[ProcessedMethods Class: com.mkyong.ws.ServerInfo]
[should process method: getIpAddress hasWebMethods: true ]
[endpointReferencesInterface: false]
[declaring class has WebSevice: true]
[returning: true]
[WrapperGen - method: getIpAddress()]
[method.getDeclaringType(): com.mkyong.ws.ServerInfo]
[requestWrapper: com.mkyong.ws.jaxws.GetIpAddress]
[ProcessedMethods Class: java.lang.Object]
com\mkyong\ws\jaxws\GetIpAddress.java
com\mkyong\ws\jaxws\GetIpAddressResponse.java
Note:   ap round: 2
```

In this case, it generated six files :

1. com\mkyong\ws\jaxws\GetIpAddress.java
2. com\mkyong\ws\jaxws\GetIpAddress.class
3. com\mkyong\ws\jaxws\GetIpAddressResponse.java
4. com\mkyong\ws\jaxws\GetIpAddressResponse.class
5. ServerInfoService_schema1.xsd
6. ServerInfoService.wsdl

File : ServerInfoService_schema1.xsd

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

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

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

  <xs:complexType name="getIpAddress">
    <xs:sequence/>
  </xs:complexType>

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

File : ServerInfoService.wsdl

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<definitions targetNamespace="http://ws.mkyong.com/"
name="ServerInfoService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://ws.mkyong.com/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://ws.mkyong.com/"
             schemaLocation="ServerInfoService_schema1.xsd"/>
    </xsd:schema>
  </types>
  <message name="getIpAddress">
    <part name="parameters" element="tns:getIpAddress"/>
  </message>
  <message name="getIpAddressResponse">
    <part name="parameters" element="tns:getIpAddressResponse"/>
  </message>
  <portType name="ServerInfo">
    <operation name="getIpAddress">
      <input message="tns:getIpAddress"/>
      <output message="tns:getIpAddressResponse"/>
    </operation>
  </portType>
  <binding name="ServerInfoPortBinding" type="tns:ServerInfo">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="getIpAddress">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
  <service name="ServerInfoService">
    <port name="ServerInfoPort" binding="tns:ServerInfoPortBinding">
      <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
    </port>
  </service>
</definitions>
```

Published It!

All files are ready, publish it via endpoint publisher.

```
package com.mkyong.endpoint;

import javax.xml.ws.Endpoint;
import com.mkyong.ws.ServerInfo;

//Endpoint publisher
public class WsPublisher{

    public static void main(String[] args) {
       Endpoint.publish("http://localhost:8888/ws/server", new ServerInfo());

       System.out.println("Service is published!");
    }

}
```

Download Source Code

Download It – JAX-WS-wsgen-command-example.zip (5KB)