Java to WSDL

Synopsis

```
java2wsdl [-?|-help|-h][-o <output-file>][-cp <class-path>][-soap12][-t <target-namespace>][-servicenam
<seservice-name>][-v][-verbose|-quiet][-s <source-directory>]
          [-classdir <compile-classes-directory>][-portname <port-name>][-createxsdimports][-d <output-
directory>] { classname }
```

Description

java2wsdl uses a compiled Web service endpoint's implementation (SEI) class and associated types classes to generate a WSDL file.

Note: java2wsdl is available only for the current production 2.0.x series of CXF. For the upcoming 2.1 versions, please use java2ws instead.

Example

java2wsdl org.apache.hello_world_soap_http.Greeter

java2wsdl -cp ./tmp org.apache.hello_world_soap_http.Greeter

java2wsdl -o hello.wsdl org.apache.hello_world_soap_http.Greeter

java2wsdl -o hello.wsdl -t http://cxf.apache.org org.apache.hello_world_soap_http.Greeter

(See below for usage with Apache Ant and Maven.)

Arguments

The arguments used to manage the WSDL file generation are reviewed in the following table.

Option	Interpretation
-?	Displays the online help for this utility.
-help	
-h	
-o	Specifies the name of the generated WSDL file.
-cp	Specify the SEI and types class search path of directories and zip/jar files.
-soap12	Specifies that the generated WSDL is to include a SOAP 1.2 binding.
-t	Specifies the target namespace to use in the generated WSDL file.
-servicename	Specifies the value of the generated service element's name attribute.
-v	Displays the version number for the tool.
-verbose	Displays comments during the code generation process.
-quiet	Suppresses comments during the code generation process.
-s	The directory in which the generated source files are placed, mostly in jaxws mode, it will generate the wrapper beans and fault beans
-classdir	The directory in which the generated sources are compiled into. If not specified, the files are not compiled.
-portname	Specify the port name to use in the generated wsdl.
-createxsdimports	Output schemas to separate files and use imports to load them instead of inlining them into the wsdl.
-d	The directory in which the output files are placed.
classname	Specifies the name of the SEI class. Note this is the compiled class, not a reference to its source file.

You must include the `classname` argument. All other arguments are optional and may be listed in any order. This tool will search and load the service endpoint class and types classes. Make certain these classes are on the `CLASSPATH` or in a location identified through the `-cp` flag.

Using java2wsdl with Ant

The java2wsdl command can be wrapped inside an Ant target as shown below:

```
<?xml version="1.0"?>
<project name="cxf java2wsdl" basedir=".">
   <property name="cxf.home" location ="/usr/myapps/cxf-2.0.1"/>
   <property name="build.classes.dir" location ="${basedir}/build/classes"/>

   <path id="cxf.classpath">
      <pathelement location="${build.classes.dir}"/>
      <fileset dir="${cxf.home}/lib">
         <include name="*.jar"/>
      </fileset>
   </path>

   <target name="cxfJavaToWSDL">
      <java classname="org.apache.cxf.tools.java2wsdl.JavaToWSDL" fork="true">
         <arg value="-o"/>
         <arg value="hello.wsdl"/>
         <arg value="service.Greeter"/>
         <classpath>
            <path refid="cxf.classpath"/>
         </classpath>
      </java>
   </target>
</project>
```

Make sure you set the "fork=true" attribute for the <java/> task as shown above. Also, remember to keep each word or flag within the command line options in its own <arg/> element (e.g., do not use <arg value="-o hello.wsdl"/>, but split them up into two <arg/> elements as done here.)

Maven Plugins

In CXF we have a Maven plugin, called "cxf-codegen-plugin", which includes a "java2wsdl" goal. You can find more information in Using CXF with maven.