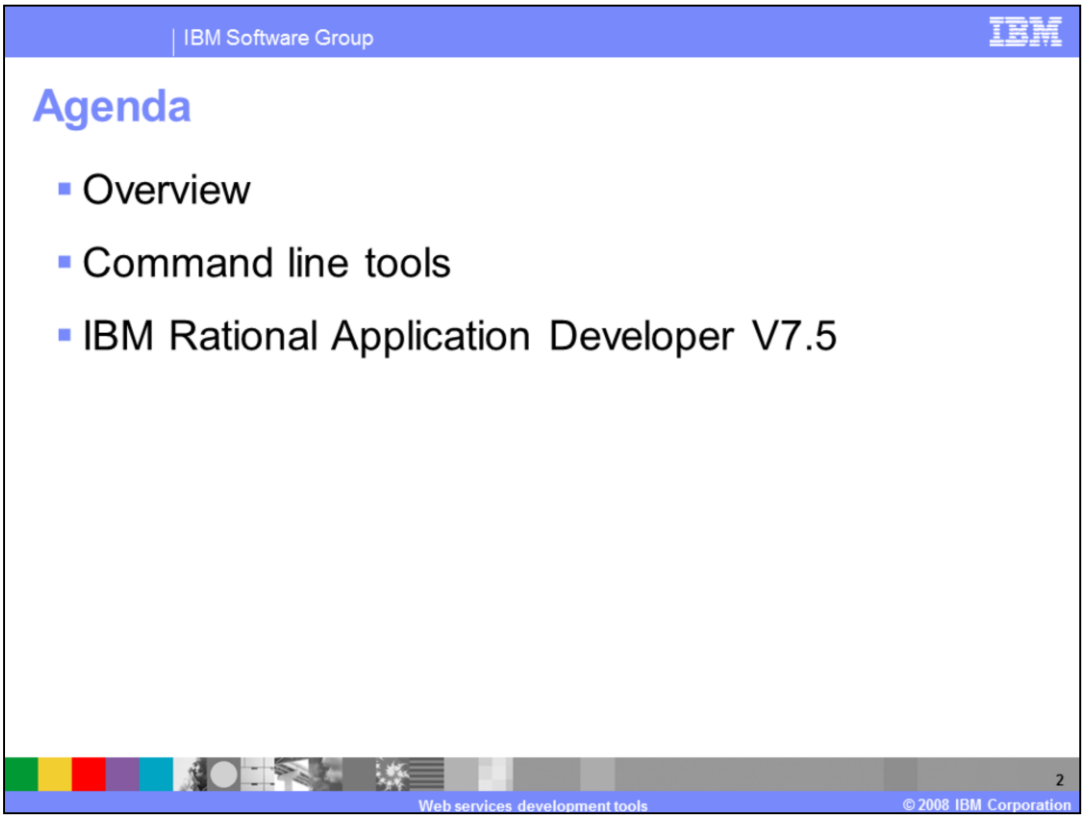
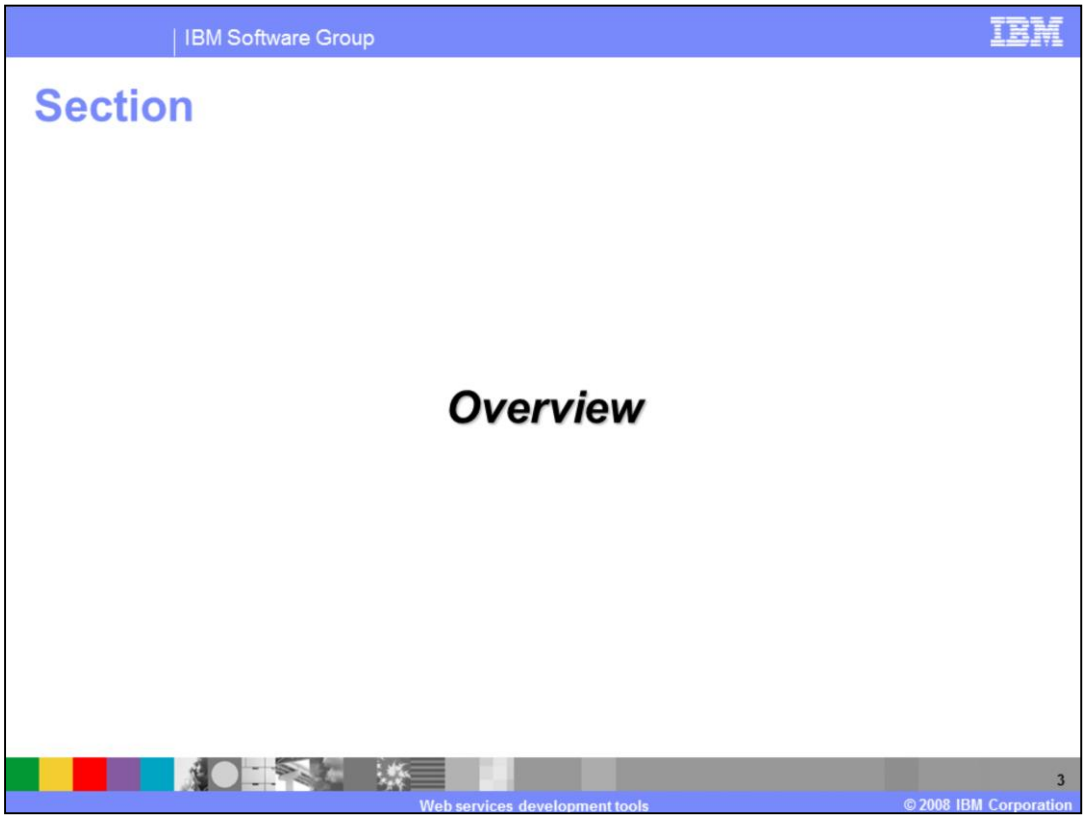
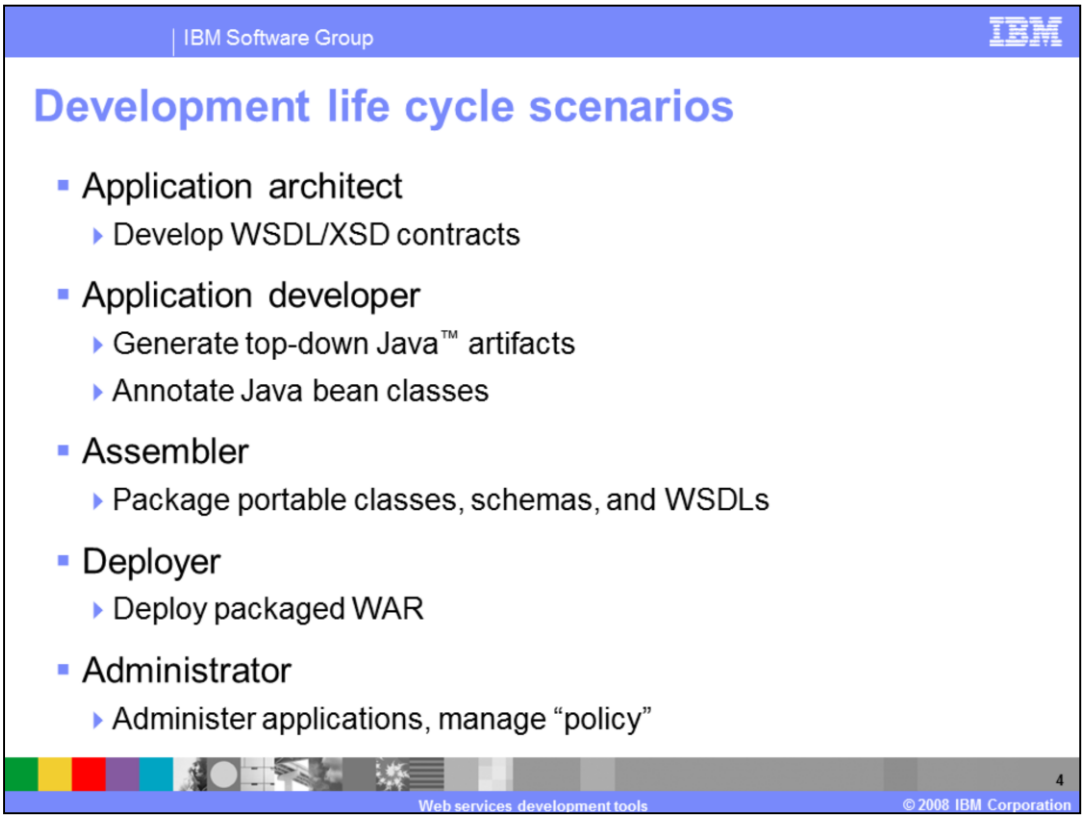
This presentation will cover the Web services development tools for WebSphere® Application Server that are available in Rational® Application Developer V7.5.



This presentation will begin with an overview of the tools options available for developing Web services applications for WebSphere Application Server V7. It will then discuss each of those options in more detail. Starting with command line tools, or scripts that can be used to generate Web services and related artifacts, then it will explain how the IBM Rational Application Developer V7.5 can be used to develop Web services.



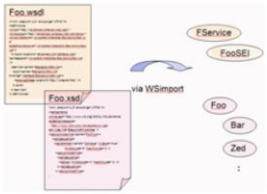
This section will provide an overview of the tools for the feature pack for Web services.

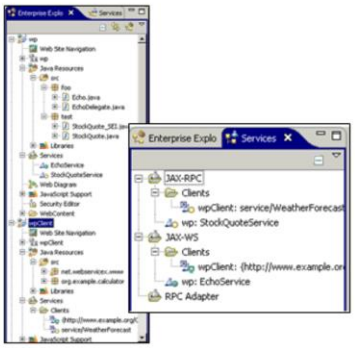


The roles listed above have been identified as key to development and management of Web services in WebSphere Application Server. In many cases, the same people play one or more of these roles. For the architect role, developing WSDL and XSD files, their role actually gets simplified. Since JAXB 2.0 is used as the mapping language, and handles all of XSD schema, it is easier for architects to develop services that use the full breadth of XSD. The application developer also has several new tools available to create JAX-WS based Web services. Administrators have also had their roles improved and simplified through the introduction of policies.

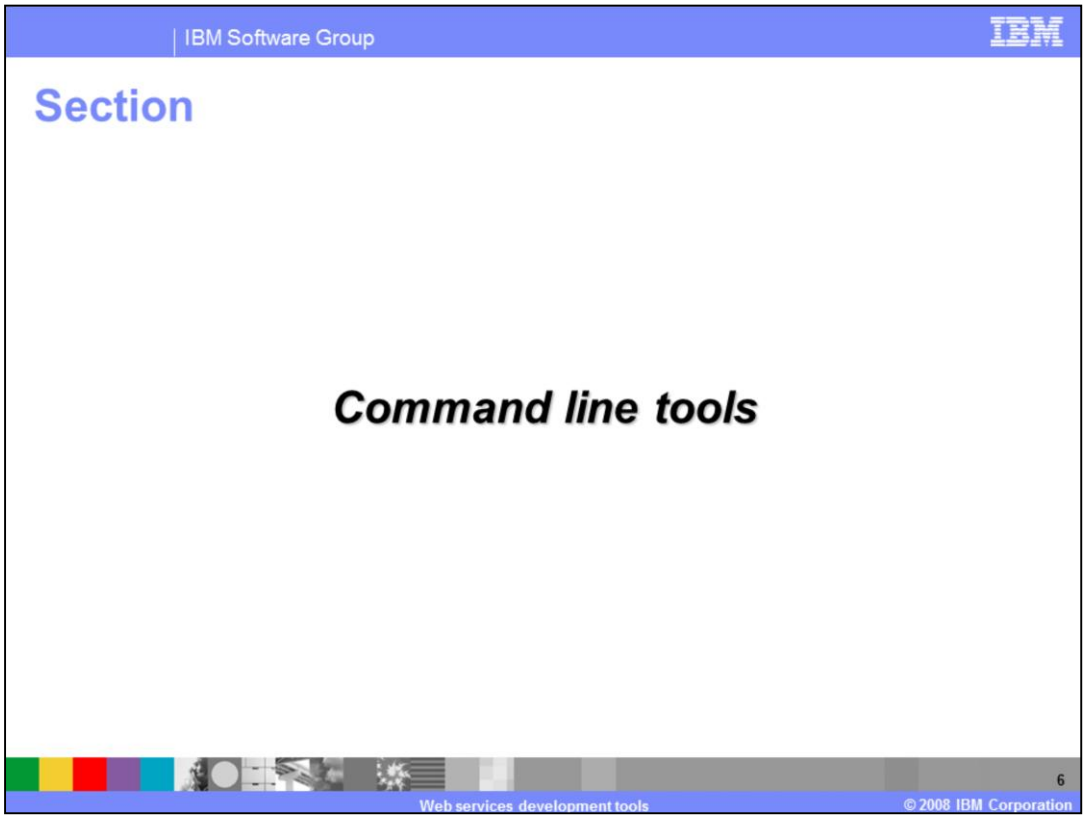
Tool options

- Command line tools
  - JAXB 2.1 XSD: Java generation (xjc)
  - JAXB 2.1: Schema generation (schemagen)
  - JAX-WS 2.1 WSDL: Java (wsimport)
  - JAX-WS 2.1 Java: WSDL (wsgen)
- Rational Application Developer
  - GUI wizards to drive command-line tools
  - Annotation validation
  - Services view and services content folder
  - Policy Set support

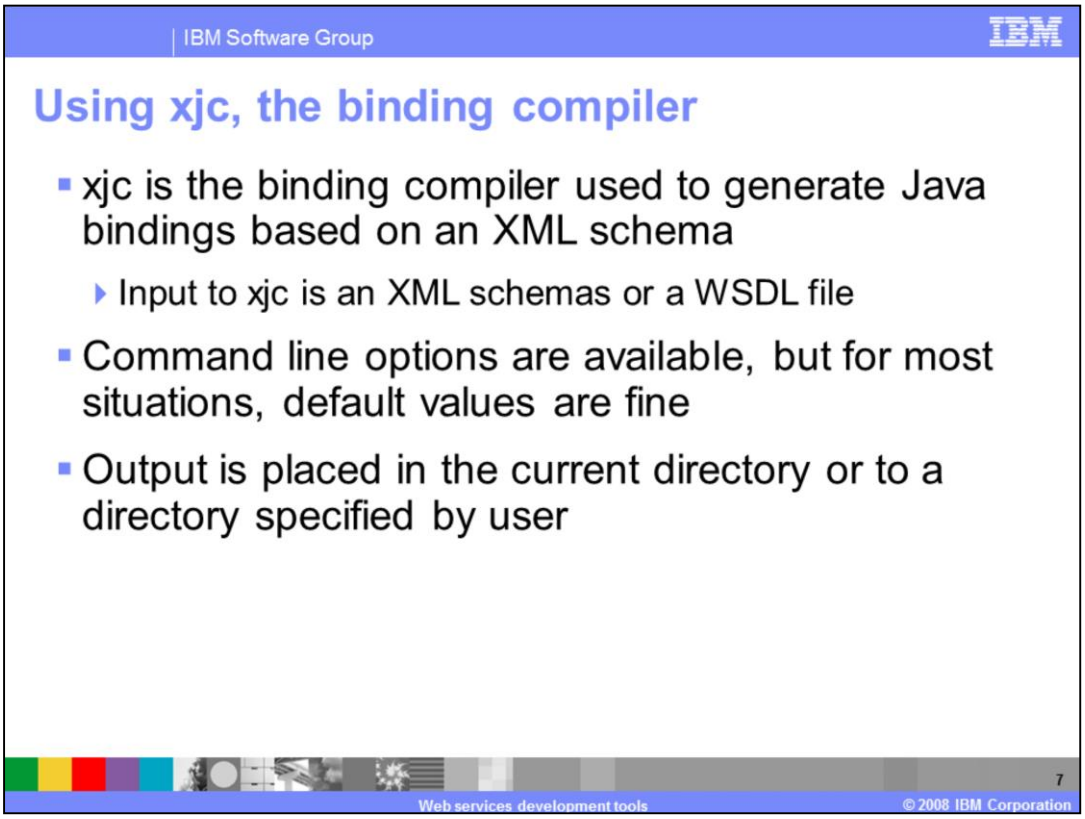




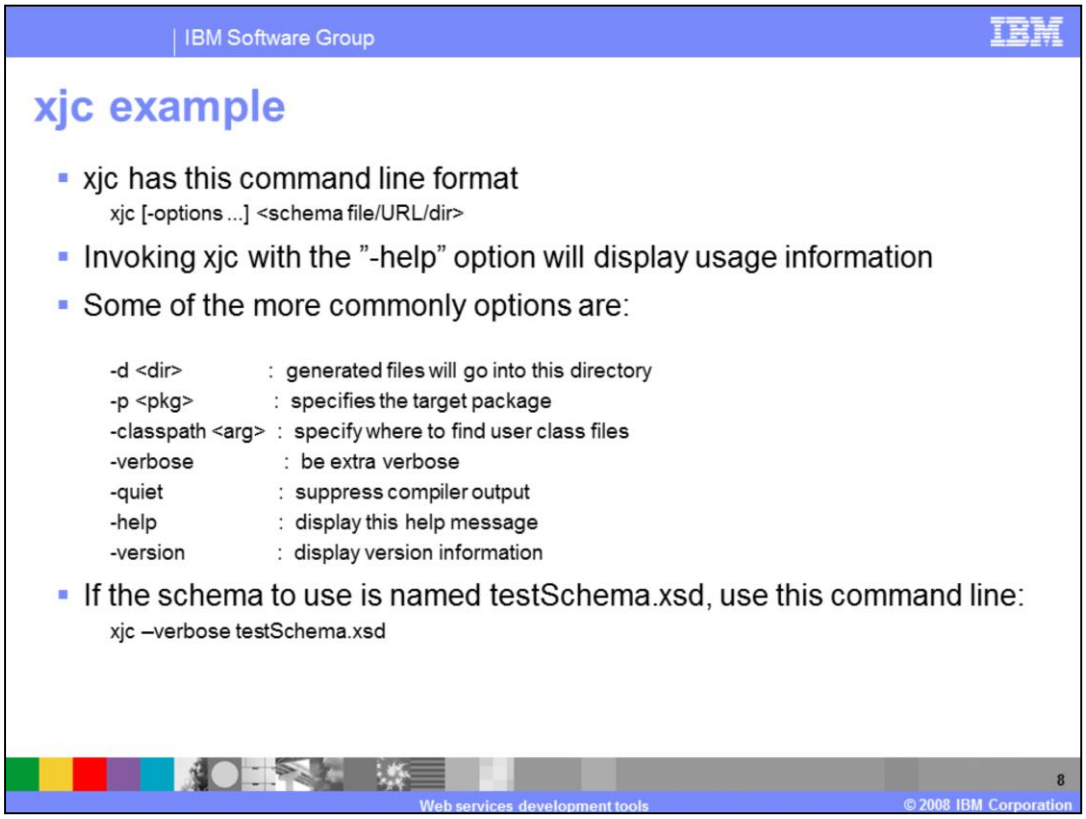
For the application developer, WebSphere Application Server V7 provides a set of tools to help make development easier. At the most basic level there is command line tools to generate various artifacts. An XJC command can be used to generate Java artifacts based on a JAXB 2.1 XSD definition or also from a WSDL file. There are also WSIMPORT and WSGEN commands for top down and bottom up Web services development. There are also updates to the Rational application developer, with additions to wizards for the creation of Web services. These have been extended to support JAX-WS and JAXB based Web services with; annotation validation, graphical wizards, publishing tools for developed Web services and a Jython debugger for scripting.



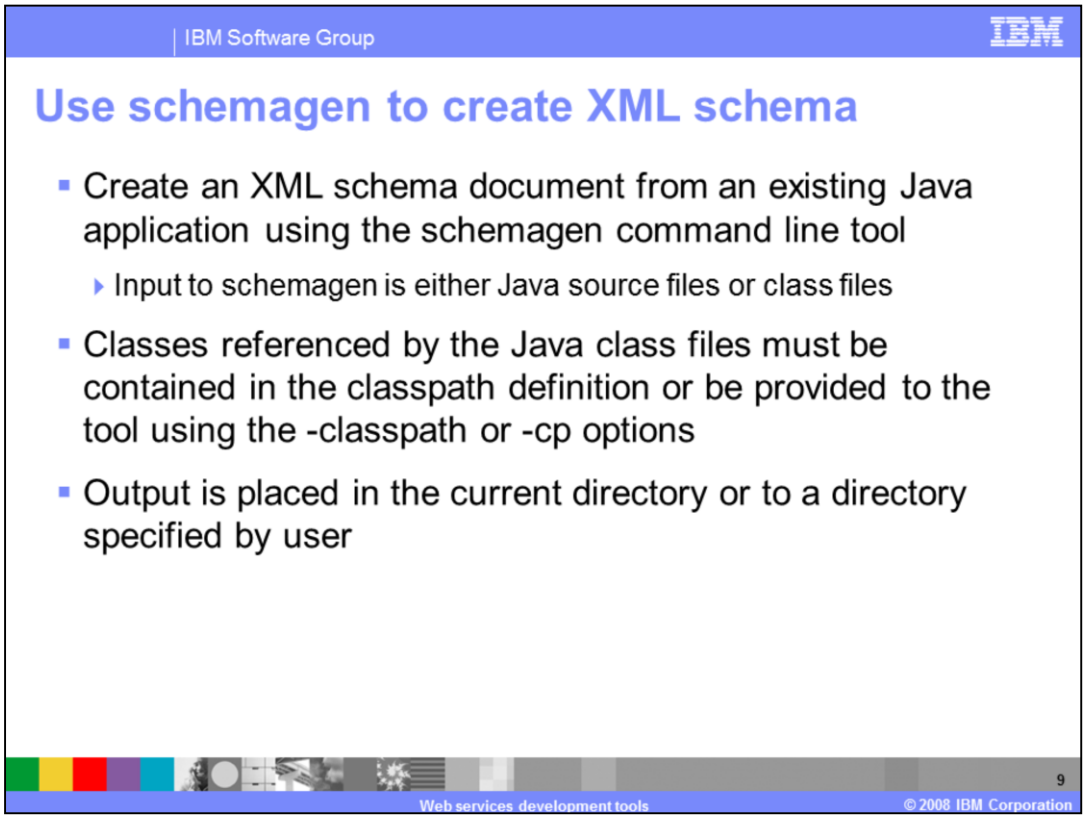
The next section discusses the Web services command line tools for WebSphere Application Server V7.



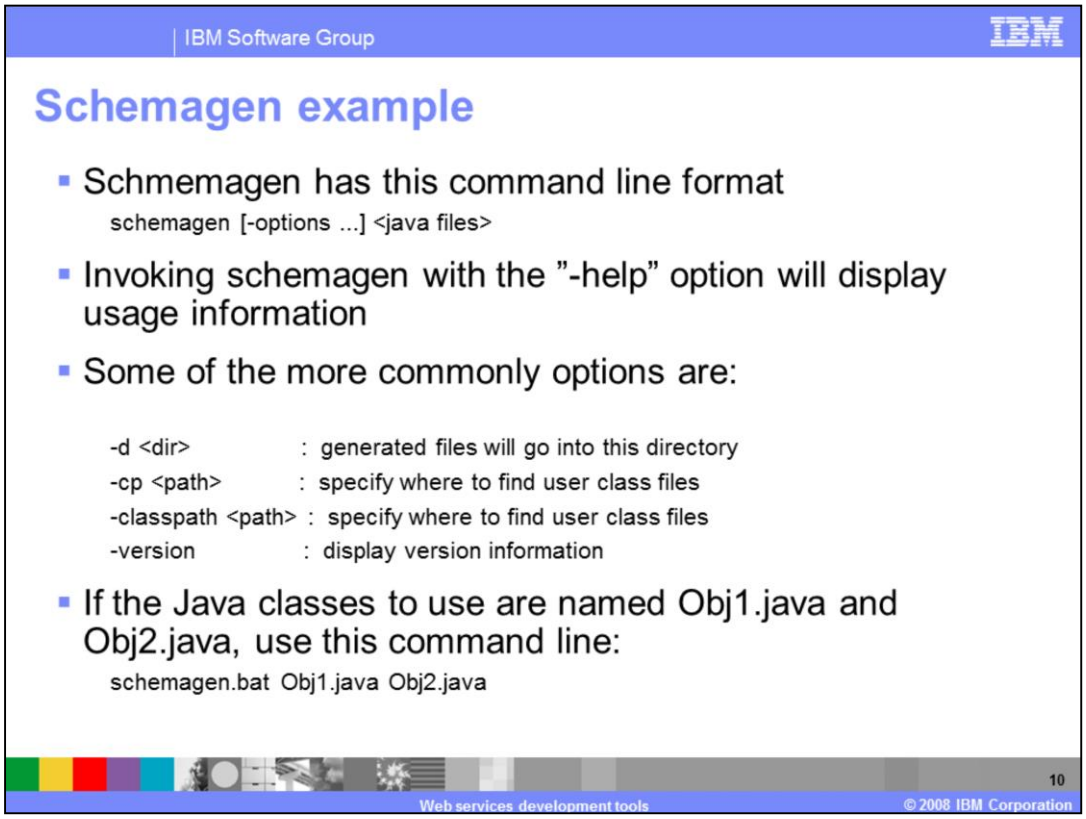
The XJC command can be used to compile the Java bindings based on an XML schema. XML schemas describe the data elements and relationships in an XML document. After a data mapping or binding exists, XML documents can be converted to and from Java objects. Generate fully annotated Java classes from an XML schema file by using the XJC command-line tool. The schema compiler is located in the bin directory under the application server root. The output will be placed in the current directory or one specified by the user.



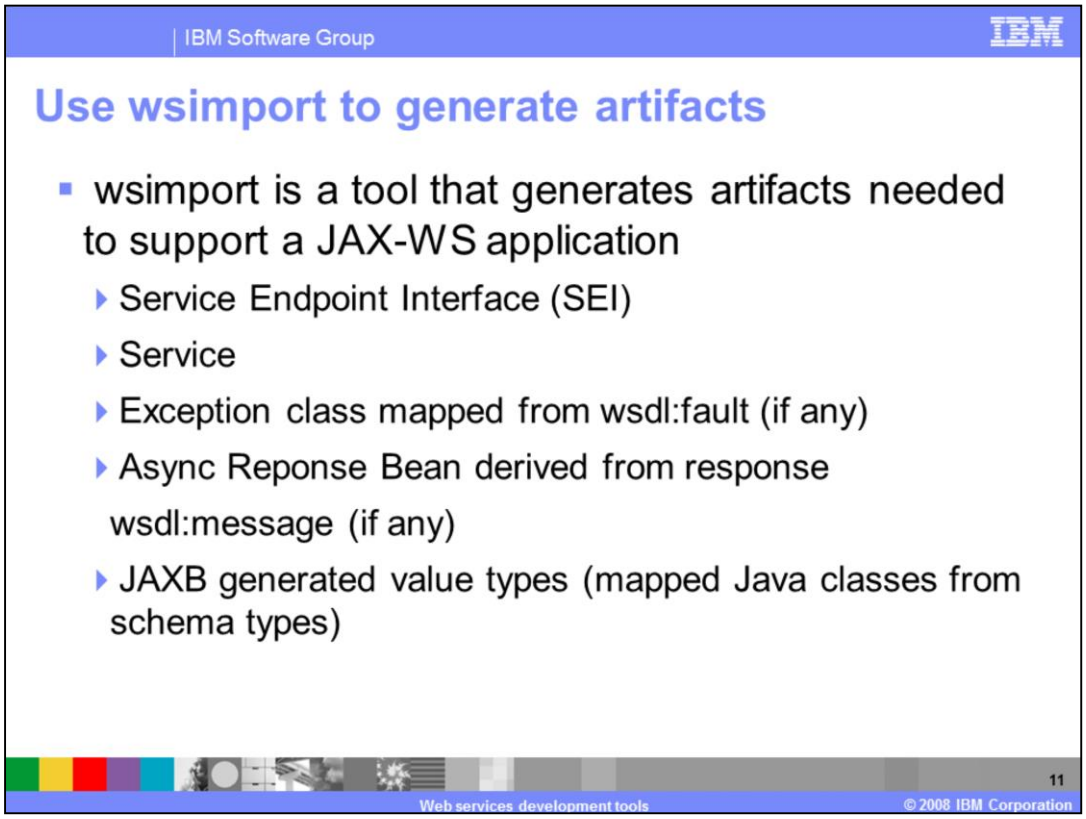
The above information shows how to use the XJC command. Use the –help option to display usage information on the command line. An example is given with a sample test schema.



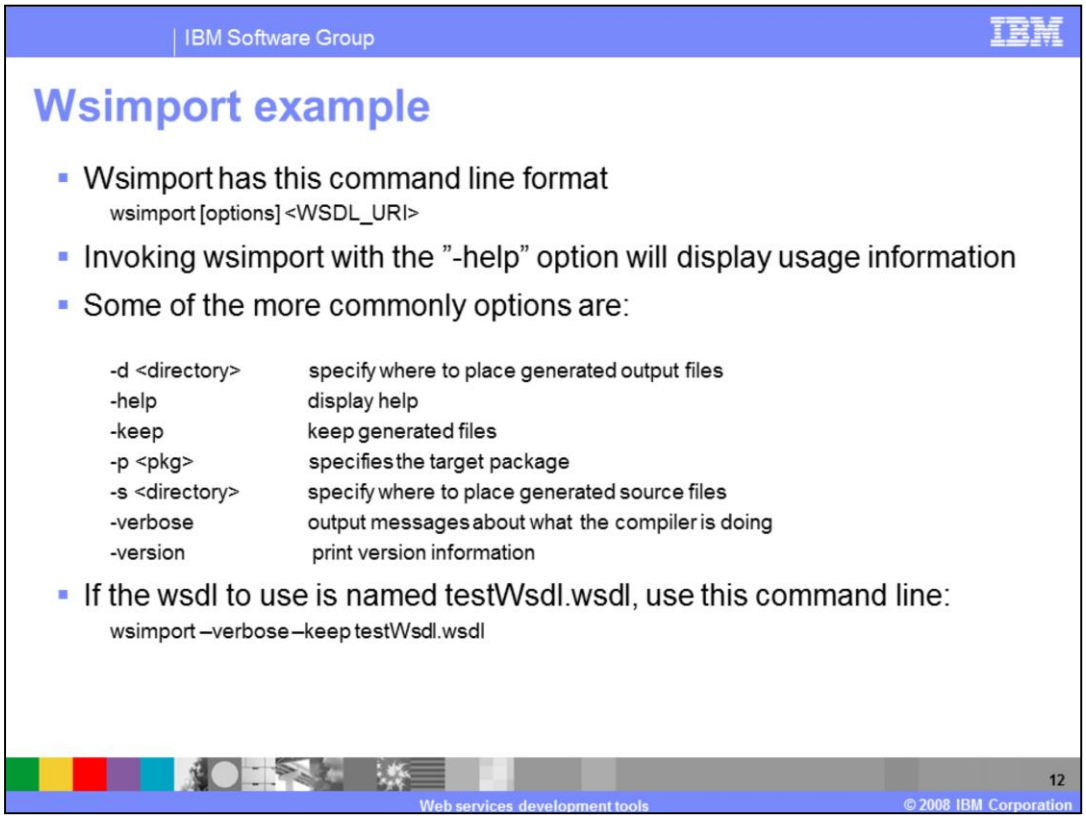
An XML schema can be documented from existing Java classes, which represent the data elements of an application, using the JAXB schema generator or schemagen command-line tool. The JAXB schema generator processes either Java source files or class files. Annotations within the Java classes provide the capability to customize the default mappings from existing Java classes to the generated schema components. The XML schema file and the annotated Java class files contain all the necessary information that JAXB requires to parse the XML documents for serialization and deserialization.



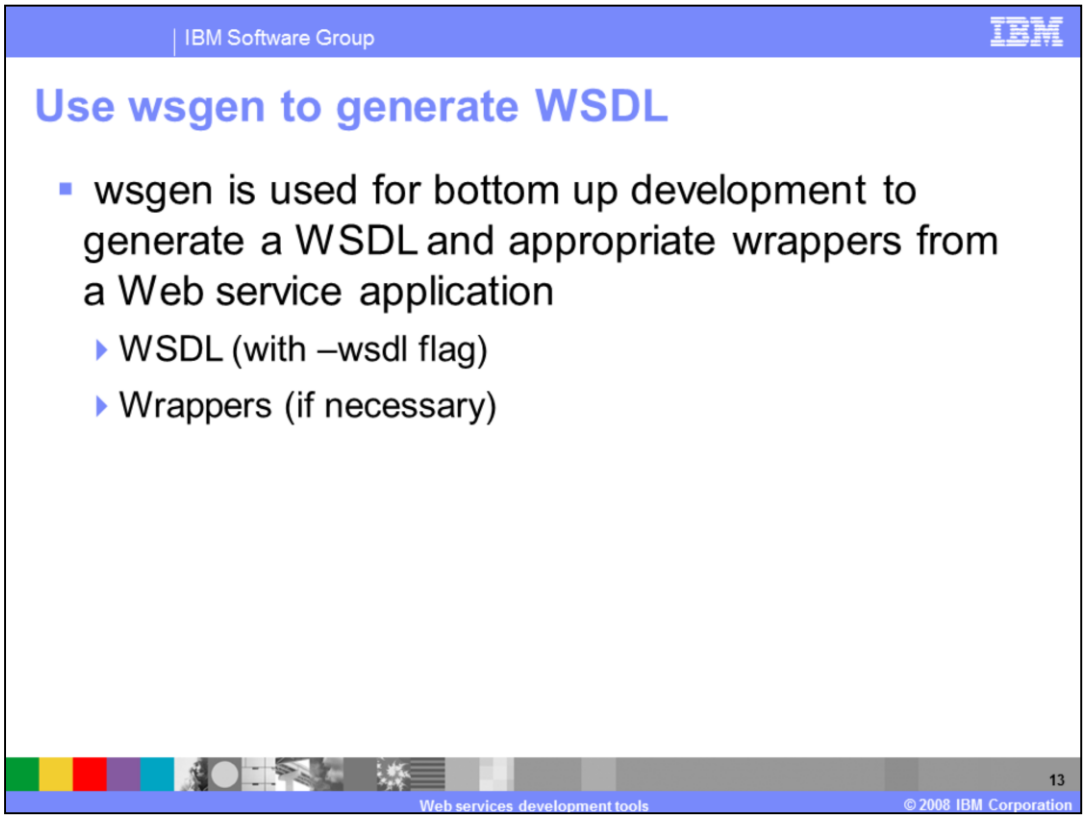
The above information shows how to use the schemagen command. Use the –help option to display usage information on the command line. An example is given with sample test files.



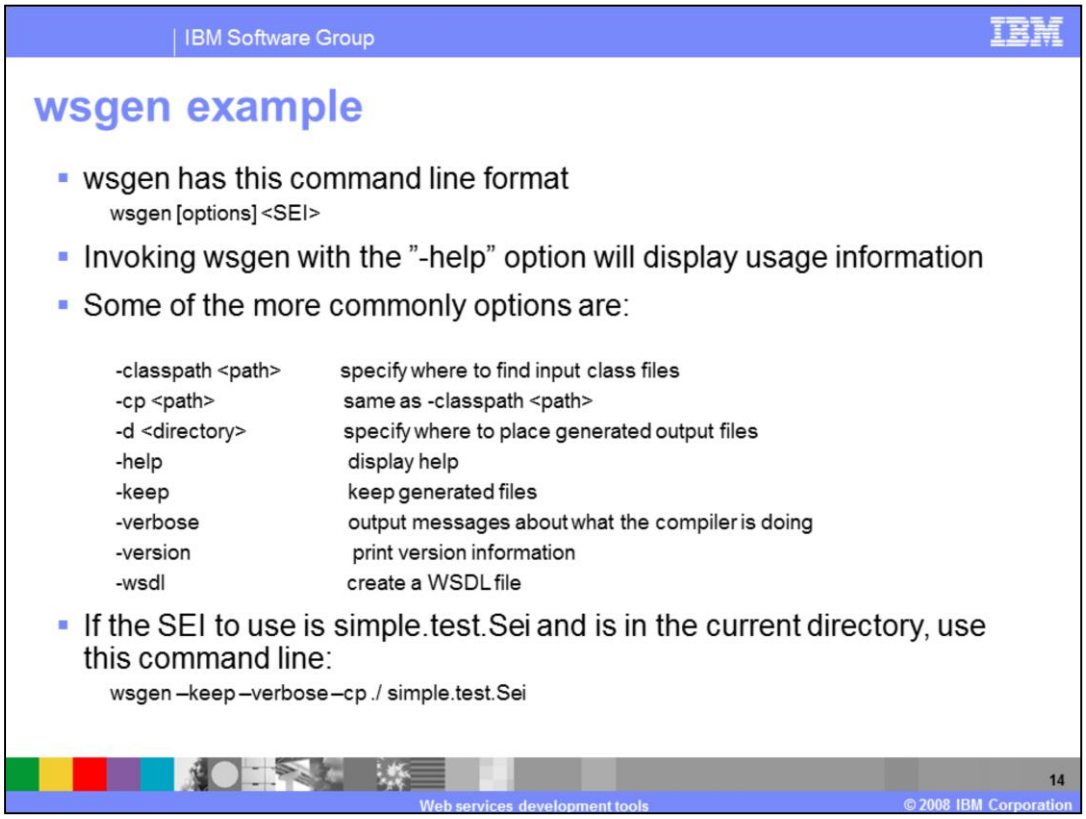
The wsimport command-line tool processes an existing Web Services Description Language (WSDL) file and creates the required portable artifacts for developing JAX-WS based Web service applications. The wsimport command-line tool supports the top-down approach to developing JAX-WS Web services, when a WSDL is used to generate the various artifacts, including the service endpoint interface, the service class, an exception class defined by the WSDL fault element, an asynchronous response bean based on the WSDL message element, and the JAXB generated types.



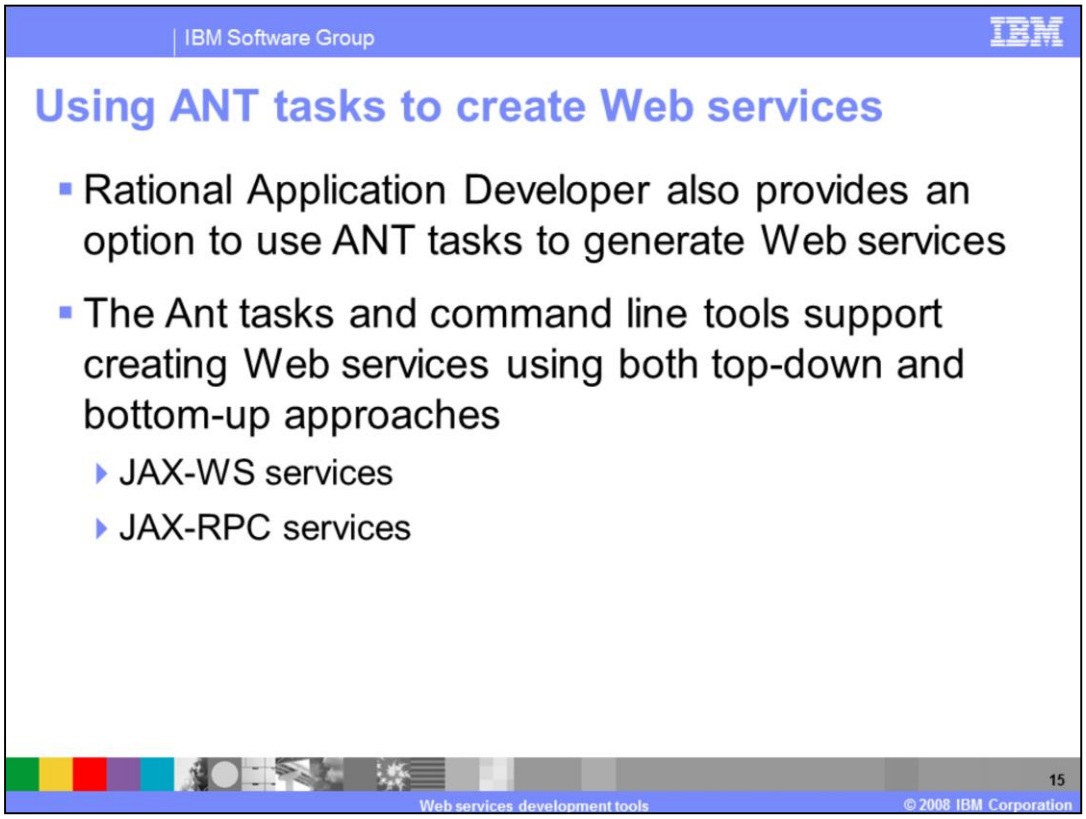
The above information shows how to use the wsimport command. Use the –help option to display usage information on the command line. An example is given with a sample WSDL, notice the use of the –keep option to retain the generated files.



The wsgen command-line tool generates the necessary portable artifacts required for JAX-WS applications when starting from Java code. This tool will generate a WSDL file only when specified. When using a bottom-up approach to develop JAX-WS Web services, creating a Web service from a service endpoint implementation, use the **wsgen** tool to generate the required artifacts. The wsgen tool accepts a properly annotated service endpoint implementation using the @WebService annotation as input and generates artifacts for the WSDL and the JAXB wrappers that may be necessary.

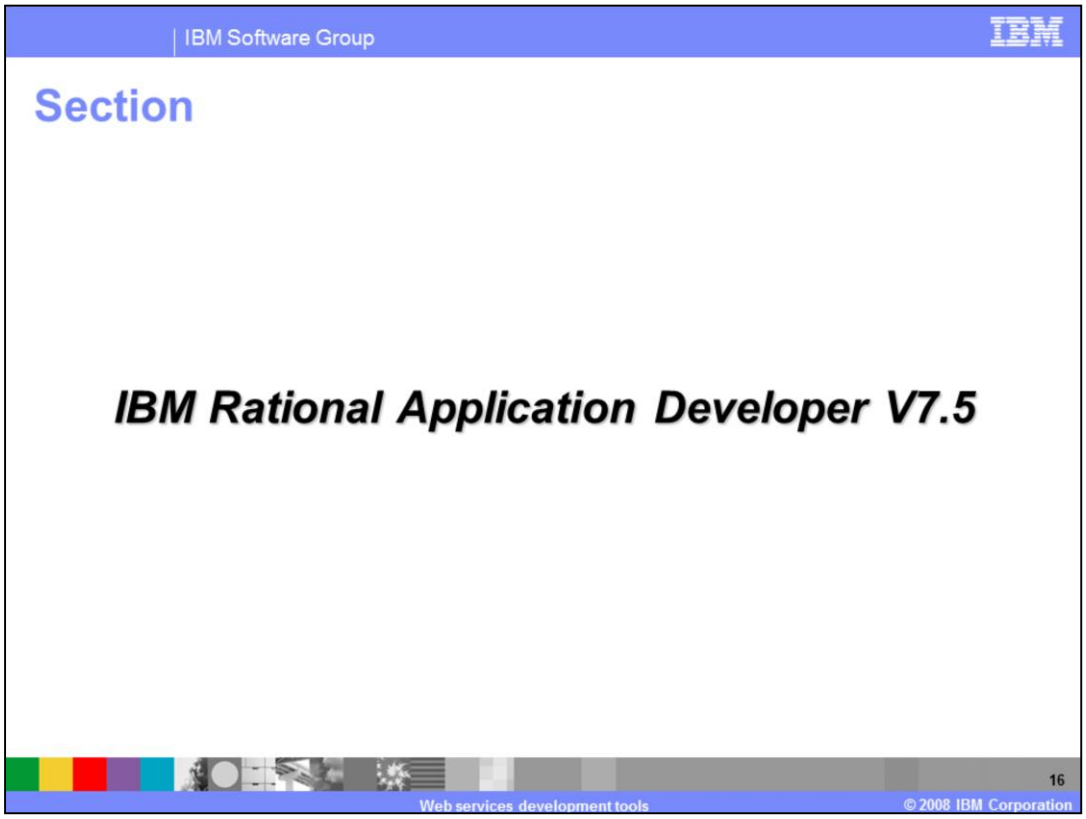


The above information shows how to use the wsgen command. Use the –help option to display usage information on the command line. An example is given with a sample service endpoint interface, notice the use of the –keep option to retain the generated files.

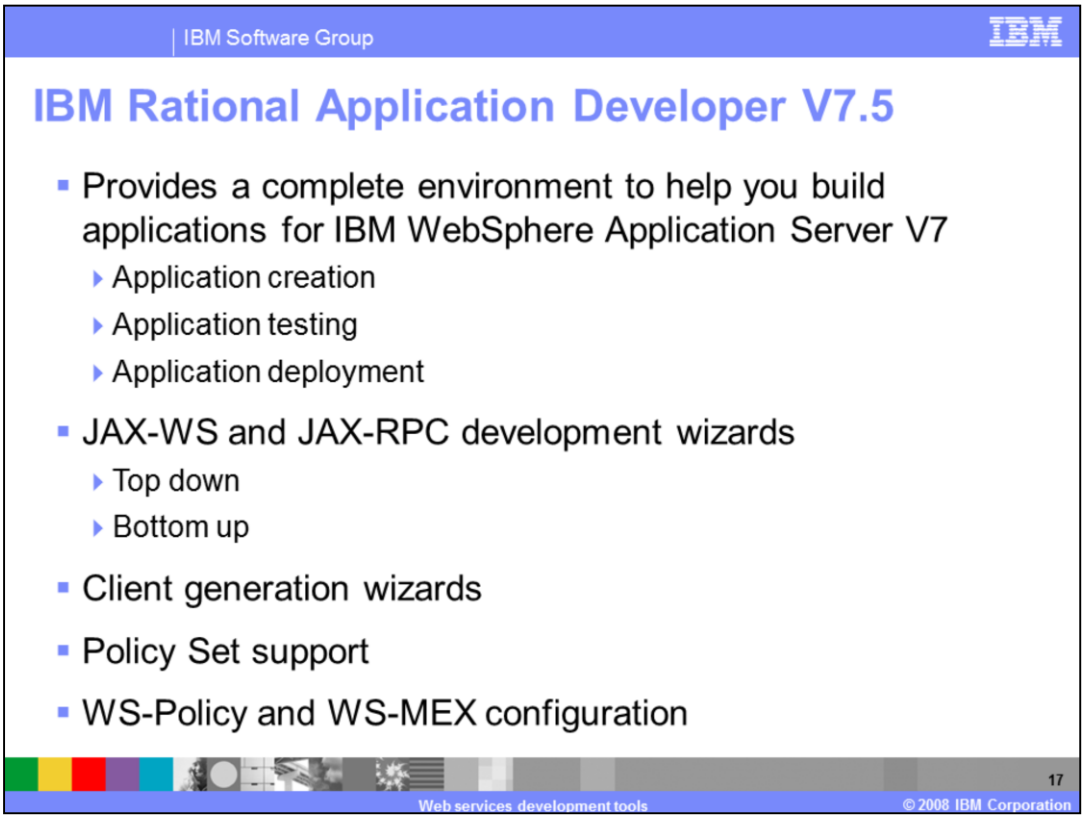


If you prefer not to use the Web service wizards, you can use Ant tasks or command line tools to create Web services using the IBM® WebSphere® runtime environments or Axis runtime environment.

The Ant tasks and command line tools support creating Web services using both top-down and bottom-up approaches. Once you have created your Web service, you can then deploy it to a server, test it, and publish it as a business entity or business service.



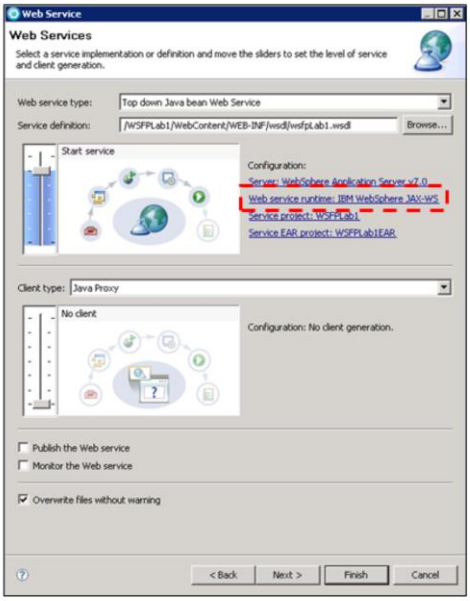
The next section explains the addition in IBM Rational Application Developer V7.5.



IBM Rational Application Developer version 7.5 offers a complete environment to develop and build applications for WebSphere Application Server version 7; including the ability to create, test and deploy applications. Development wizards are provided to develop both JAX-WS and JAX-RPC applications from WSDL documents or from Java implementations. Client generation wizards are also provided, as are tools to work with policy set configurations in Rational Application Developer. Also tools to configure WS-Policy and WS-Metadata Exchange are provided in this release.

Web services creation wizard

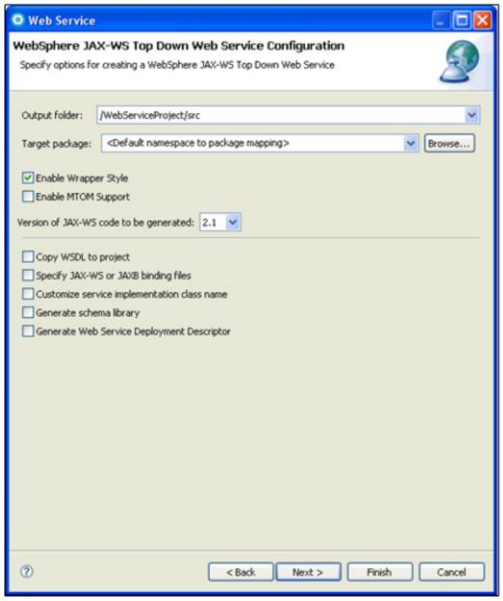
- Select the Web service runtime
  - IBM WebSphere JAX-RPC
  - IBM WebSphere JAX-WS
- Configure the client that should be created



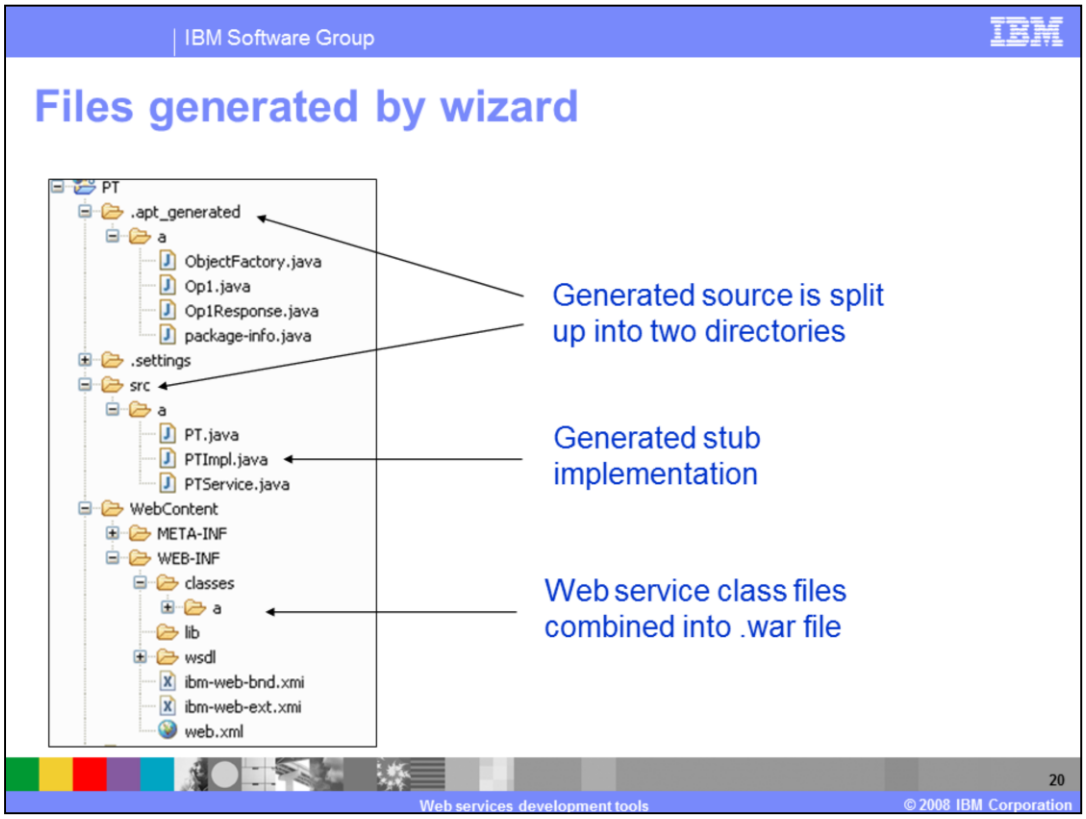
When creating a new Web service using the Web services creation wizard, new options have been added. One of these is for the Web Service runtime to use, this can be changed between IBM WebSphere JAX-RPC and IBM WebSphere JAX-WS. The type of client that should be created can also be chosen.

Web service creation wizard

- Select options related to the Web service
- Option to generate the Web service deployment descriptor



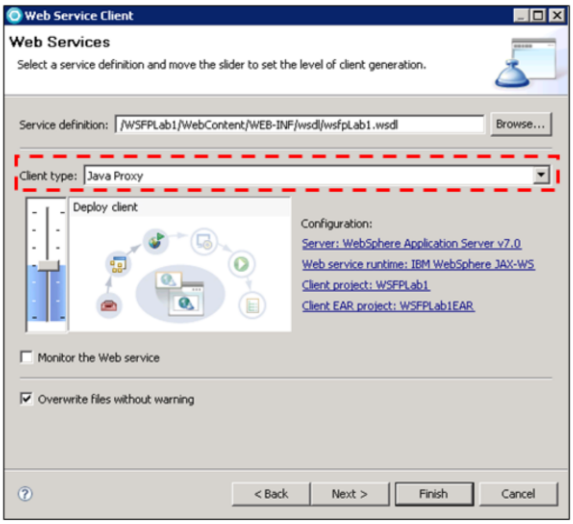
Further options can be configured for a JAX-WS Web service. These include enabling the wrapper style for the Web Service, copying the WSDL to the project, and specifying JAX-WS or JAX-B binding files. The option to generate the schema library or the Web service deployment descriptor have also been added in Rational Application Developer V7.5.



This picture illustrates the location of various files when generated by the tools for a bottom up Web service. The source files are split between the SRC folder and a folder for portable Java artifacts for the Web service. The stub implementation is placed in the SRC folder. The class files are combined into a single .WAR file that can be exported and deployed.

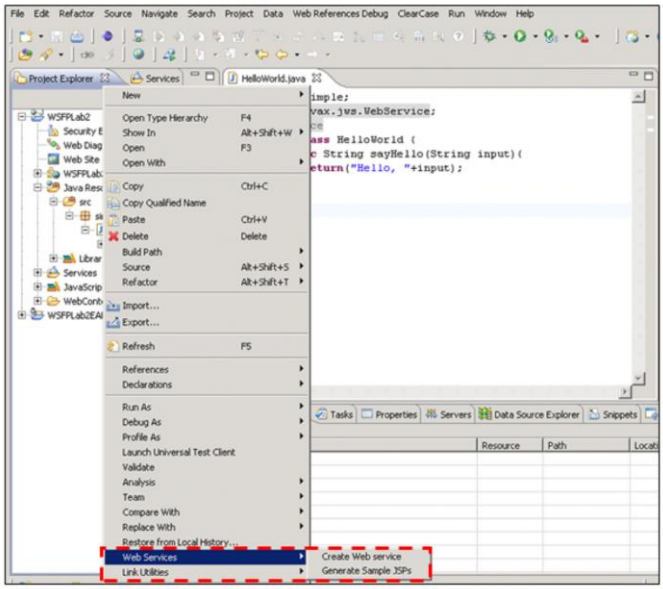
Web service client wizard

- Choose from the available client types
- Select the Web service runtime
  - IBM WebSphere JAX-RPC
  - IBM WebSphere JAX-WS

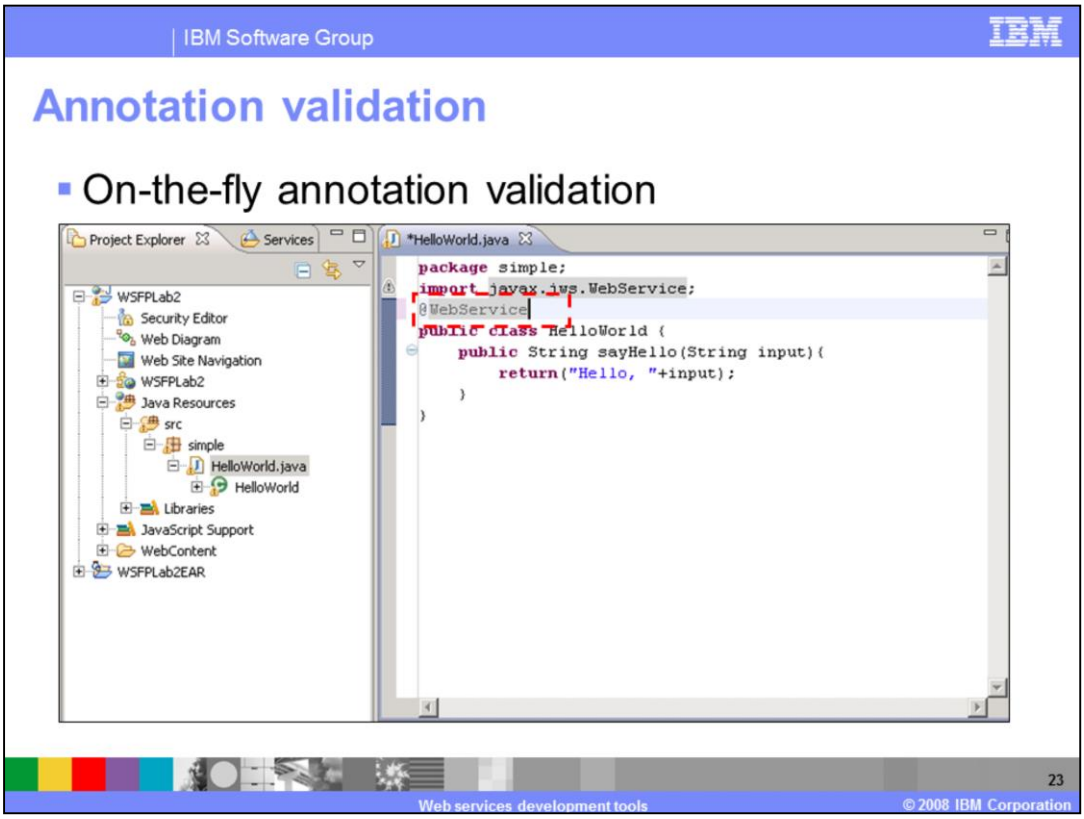


There is also a client generation wizard for IBM WebSphere JAX-WS and IBM WebSphere JAX-RPC clients. The additional options allow a user to enable asynchronous invocation of the client, to specify JAX-WS or JAX-B binding files, and to customize the client proxy class name.

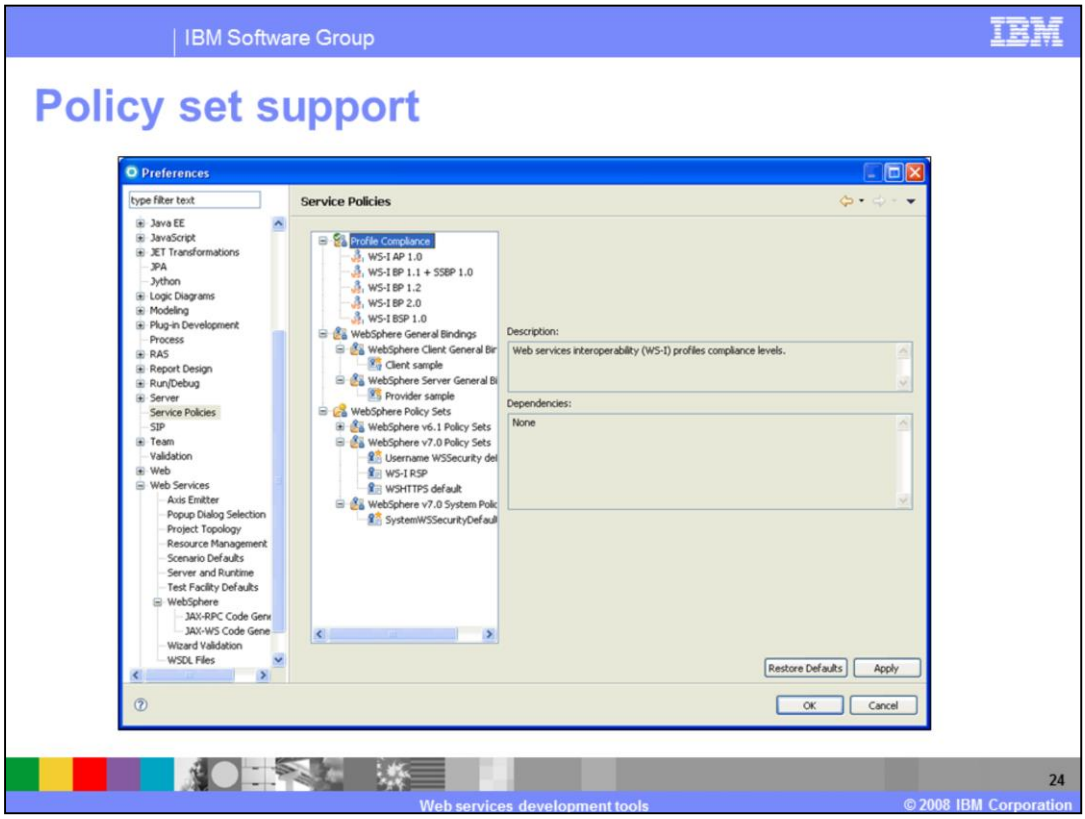
Bottom up mapping and JSP wizards



There are also tools provided for bottom up development as well. Right click a class to be able to create a Web service from that class, or to generate sample JSPs to test that class.

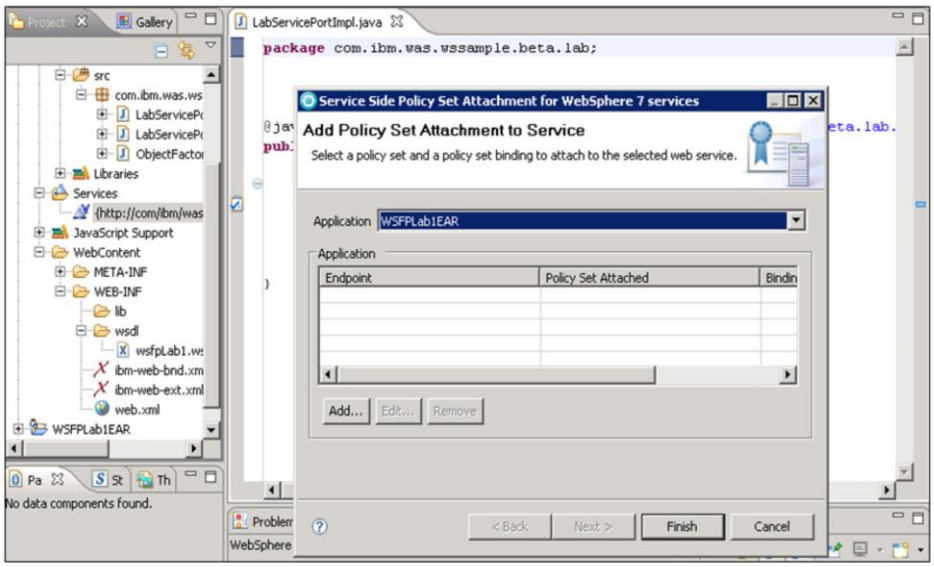


As annotations are added to the source, the AST will validate those annotations as they are typed. The file doesn't have to be saved to be validated. Tips will be provided as the annotations are typed as well.

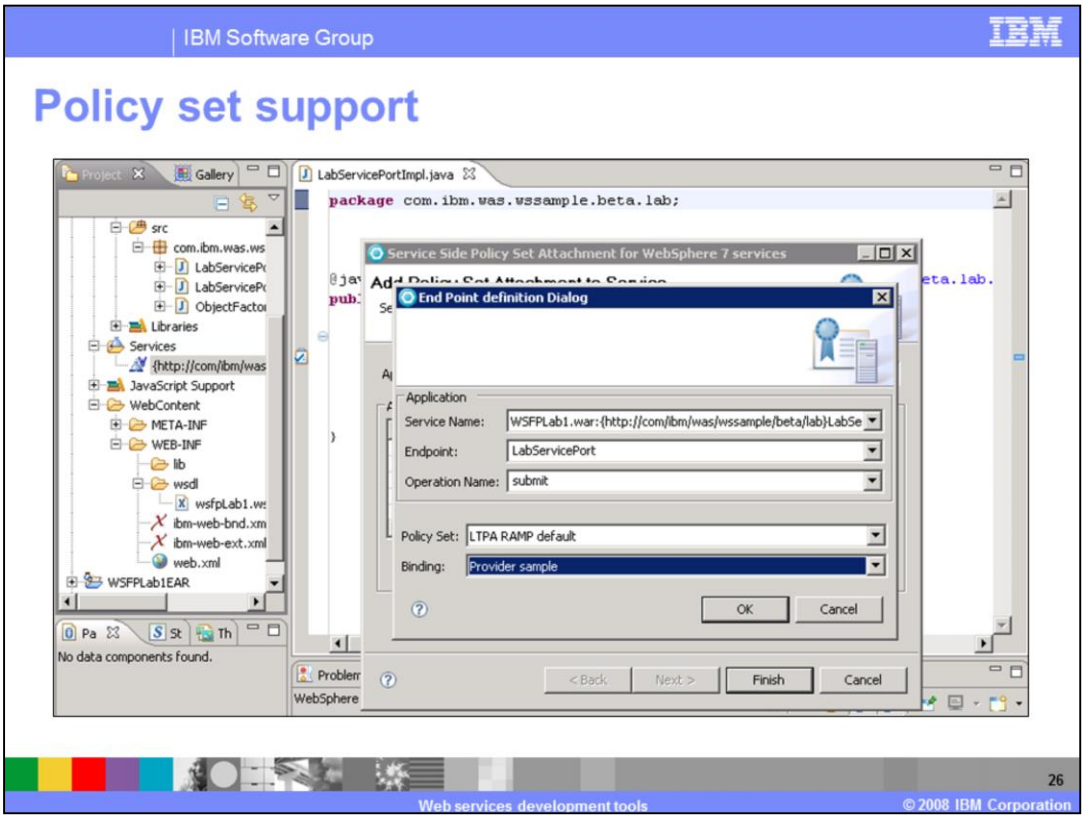


Rational Application Developer also has the abilities to work with policy sets. Policy sets can be imported in the preferences view under quality of services. This allows a developer to work with the same policy sets as are used in the production environment.

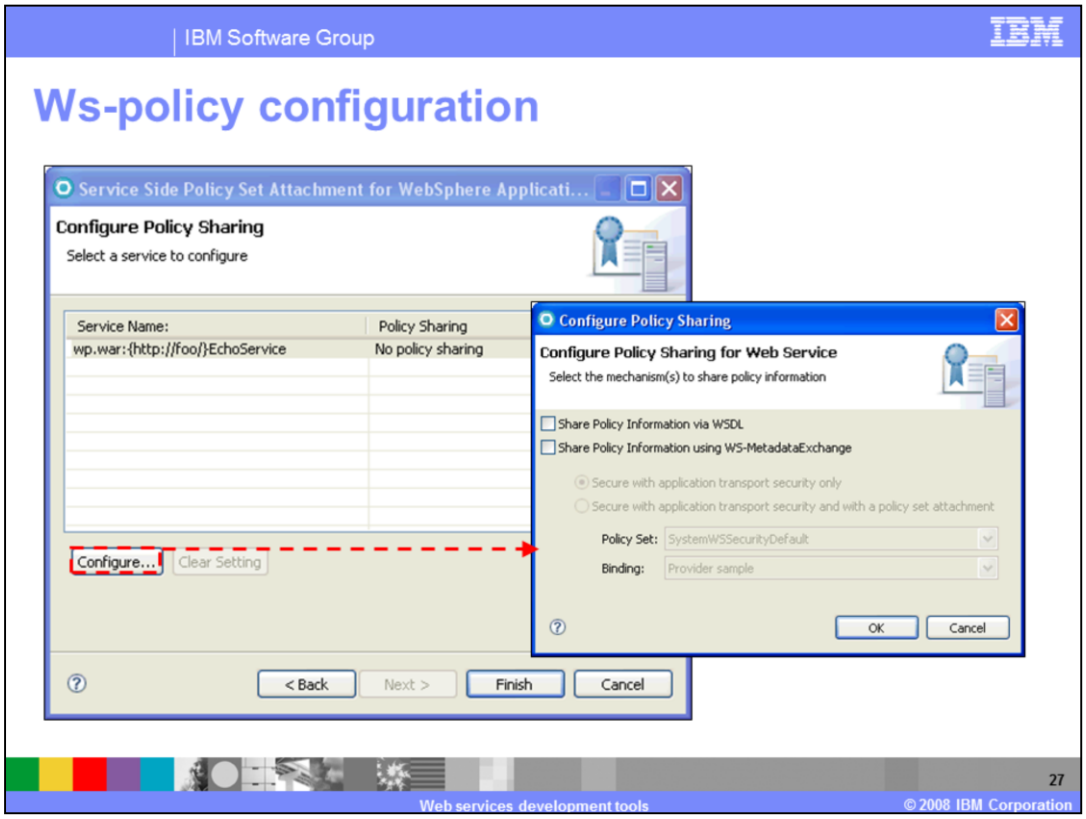
Policy set support



Policy sets can be attached to Web services using a wizard. All policy sets attached to the service will be listed, and more can be added or removed.



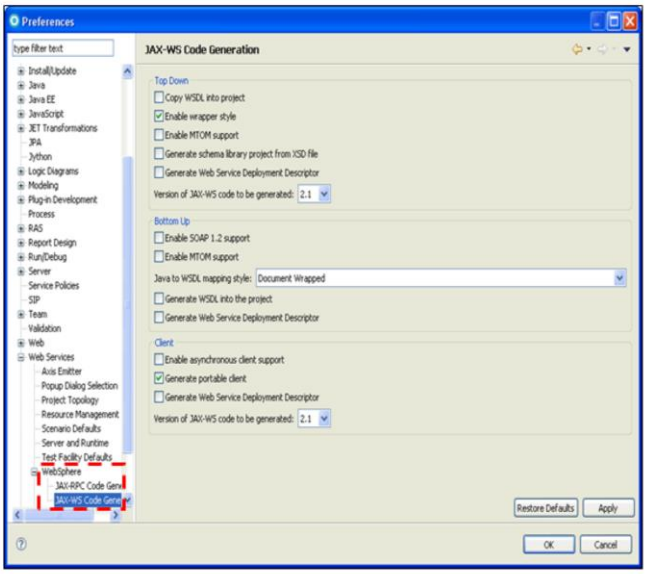
Policy sets can be modified as well, through the wizard shown here. All policy sets attached to the service will be listed and each can be modified. Customized policy sets can then be exported to be imported into a WebSphere Application Server environment.



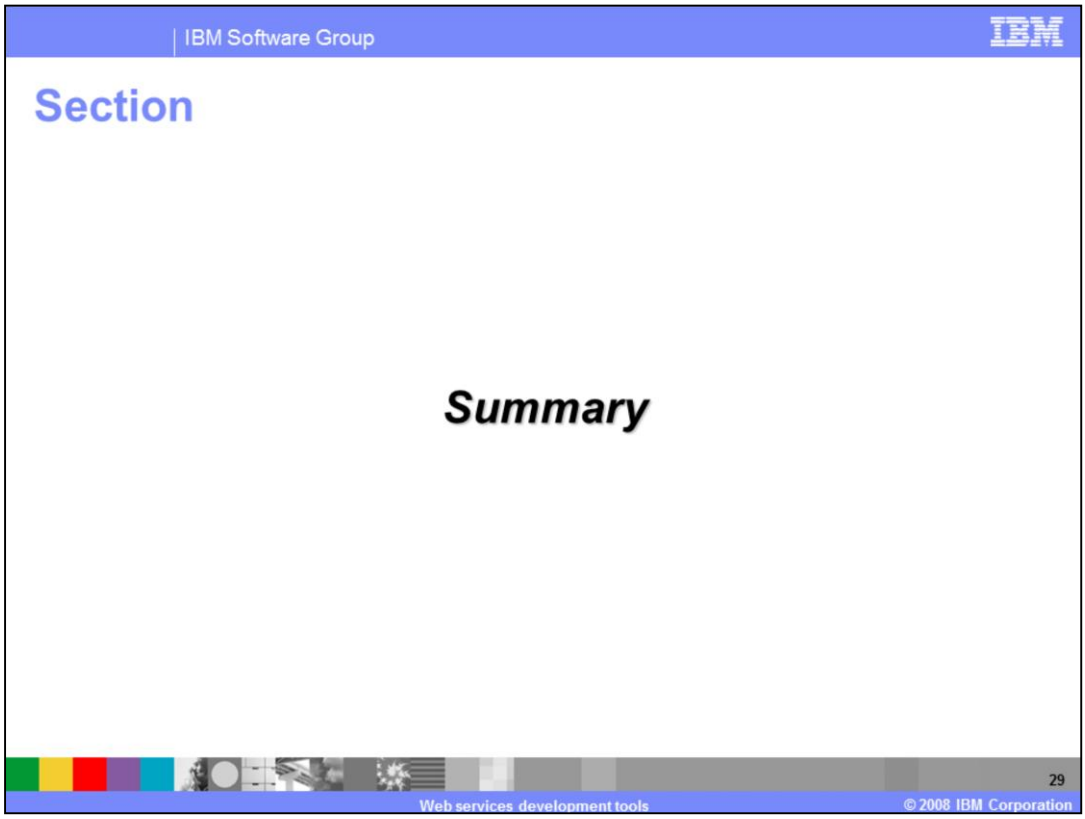
Services can also be configured for WS-Policy and WS-Metadata Exchange support as well. Developers can select an option to share this information in the WSDL based on WS-Policy or using the WS-Metadata Exchange specifications.

Code generation preferences

- Set code defaults for the project
- Separate code generation preferences for JAX-WS and JAX-RPC applications



The defaults used by the tools for JAX-WS and JAX-RPC Web services can be set in the preferences, as shown above. Items such as enabling SOAP 1.2 support, enabling MTOM support and others can enabled by default on all JAX-WS Web services that are created.



The next section will summarize the materials.

Developers have several choices when creating JAX-WS applications with WebSphere Application Server V7. Command line tools and the IBM Rational Application Developer version 7.5 are available.

Trademarks, copyrights, and disclaimers