NAME

xjc − Compiles an XML schema file into fully annotated Java classes.

SYNOPSIS

xjc [*options*] **schema** *file/URL/dir/jar ...* [**−b** *bindinfo*] *...*

options The command-line options. See Options.

schema *file/URL/dir/jar ...*
The location of the XML schema file. If **dir** is specified, then all schema files in it are compiled. If **jar** is specified, then the **/META-INF/sun-jaxb.episode** binding file is compiled.

-b *bindinfo*
The location of the bindings files.

DESCRIPTION

Start the binding compiler with the appropriate **xjc** shell script in the bin directory for your platform. There is also an Ant task to run the binding complier. See Using the XJC with Ant at http://jaxb.java.net/nonav/2.1.3/docs/xjcTask.html

OPTIONS

- See also Nonstandard Options
- See also Deprecated and Removed Options

-nv
By default, the XJC binding compiler performs strict validation of the source schema before processing it. Use this option to disable strict schema validation. This does not mean that the binding compiler will not perform any validation, but means that it will perform a less-strict validation.

-extension
By default, the XJC binding compiler strictly enforces the rules outlined in the Compatibility chapter of the JAXB Specification. Appendix E.2 defines a set of W3C XML Schema features that are not completely supported by JAXB v1.0. In some cases, you may be allowed to use them in the **-extension** mode enabled by this switch. In the default (strict) mode, you are also limited to using only the binding customization defined in the specification. By using the **-extension** switch, you will be allowed to use the JAXB Vendor Extensions.

-b *file*
Specifies one or more external binding files to process. Each binding file must have its own **-b** switch. The syntax of the external binding files is flexible. You can have a single binding file that contains customization for multiple schemas or you can break the customization into multiple bindings files: **xjc schema1.xsd schema2.xsd schema3.xsd -b bindings123.xjbxjc schema1.xsd schema2.xsd schema3.xsd -b bindings1.xjb -b bindings2.xjb -b bindings3.xjb**. In addition, the ordering of the schema files and binding files on the command line does not matter.

-d *dir*
By default, the XJC binding compiler generates the Java content classes in the current directory. Use this option to specify an alternate output directory. The output directory must already exist. The XJC binding compiler does not create it for you.

-p *pkg*
When you specify a target package with this command-line option, it overrides any binding customization for the package name and the default package name algorithm defined in the specification.

-httpproxy *proxy*
Specifies the HTTP or HTTPS proxy in the format *[user[:password]@]proxyHost[:proxyPort]*. The old **-host** and **-port** options are still supported by the RI for backward compatibility, but they were deprecated. The password specified with this option is an argument that is visible to other users who use the top command. For greater security, use the **-httpproxyfile** option.

-httpproxyfile file
: Specifies the HTTP or HTTPS proxy with a file. The same format as the **-httpproxy** option, but the password specified in the file is not visible to other users.

-classpath arg
: Specifies where to find client application class files used by the *jxb:javaType* and xjc:*superClass* customization.

-catalog file
: Specifies catalog files to resolve external entity references. Supports the TR9401, XCatalog, and OASIS XML Catalog formats. See XML Entity and URI Resolvers at http://xerces.apache.org/xml-commons/components/resolver/resolver-article.html

-readOnly
: By default, the XJC binding compiler does not write-protect the Java source files it generates. Use this option to force the XJC binding compiler to mark the generated Java sources as read-only.

-npa
: Suppresses the generation of package level annotations into ****/package-info.java**. Using this switch causes the generated code to internalize those annotations into the other generated classes.

-no-header
: Suppresses the generation of a file header comment that includes some note and time stamp. Using this makes the generated code more compatible with the **diff** command.

-target 2.0
: Avoids generating code that relies on any JAXB 2.1 features. This will allow the generated code to run with JAXB 2.0 runtime environment (such as Java SE 6).

-xmlschema
: Treats input schemas as W3C XML Schema (default). If you do not specify this switch, then your input schemas are treated as though they are W3C XML Schemas.

-relaxing
: Treats input schemas as RELAX NG (experimental and unsupported). Support for RELAX NG schemas is provided as a JAXB Vendor Extension.

-relaxing-compact
: Treat input schemas as RELAX NG compact syntax (experimental and unsupported). Support for RELAX NG schemas is provided as a JAXB Vendor Extension.

-dtd
: Treats input schemas as XML DTD (experimental and unsupported). Support for RELAX NG schemas is provided as a JAXB Vendor Extension.

-wsdl
: Treats input as WSDL and compiles schemas inside it (experimental and unsupported).

-quiet
: Suppress compiler output, such as progress information and warnings.

-verbose
: Be extra verbose, such as printing informational messages or displaying stack traces upon some errors.

-help
: Displays a brief summary of the compiler switches.

-version
: Displays the compiler version information.

schema file/URL/dir
: Specifies one or more schema files to compile. If you specify a directory, then the **xjc** command scans it for all schema files and compiles them.

NONSTANDARD OPTIONS

-XLocator
: Causes the generated code to expose SAX Locator information about the source XML in the Java bean instances after unmarshalling.

-Xsync-methods
: Causes all of the generated method signatures to include the **synchronized** keyword.

-mark-generated
: Marks the generated code with the annotation **@javax.annotation.Generated**.

-episode file
: Generates the specified episode file for separate compilation.

DEPRECATED AND REMOVED OPTIONS

-host & -port
: These options are replaced with the **-httpproxy** option. For backward compatibility, these options are supported, but will not be documented and might be removed from future releases.

-use-runtime
: Because the JAXB 2.0 specification has defined a portable runtime environment, it is no longer necessary for the JAXB RI to generate ****/impl/runtime**packages. Therefore, this switch is obsolete and was removed.

-source
: The **-source** compatibility switch was introduced in the first JAXB 2.0 Early Access release. This switch is removed from future releases of JAXB 2.0. If you need to generate 1.0.x code, then use an installation of the 1.0.x code base.

COMPILER RESTRICTIONS

In general, it is safest to compile all related schemas as a single unit with the same binding compiler switches. Keep the following list of restrictions in mind when running the **xjc** command. Most of these issues only apply when you compile multiple schemas with multiple invocations of the **xjc** command.

To compile multiple schemas at the same time, keep the following precedence rules for the target Java package name in mind:

1. The **-p** option has the highest precedence.
2. *jaxb:package* customization.
3. If **targetNamespace** is declared, then apply the **targetNamespace** to the Java package name algorithm defined in the specification.
4. If no **targetNamespace** is declared, then use a hard coded package named **generated**.

You cannot have more than one *jaxb:schemaBindings* per name space, so it is impossible to have two schemas in the same target name space compiled into different Java packages.

All schemas being compiled into the same Java package must be submitted to the XJC binding compiler at the same time. They cannot be compiled independently and work as expected.

Element substitution groups that are spread across multiple schema files must be compiled at the same time.

SEE ALSO

- Binding Compiler (xjc) at http://jaxb.java.net/nonav/2.2.3u1/docs/xjc.html
- Java Architecture for XML Binding (JAXB) at http://www.oracle.com/technetwork/articles/javase/index-140168.html