Coding Guidelines

Introduction

CXF makes use of both PMD and CheckStyle to enforce common coding conventions. However, not everything can be expressed in a rule in one of these systems. This page describes additional conventions and considerations.

CODE FORMATTING

We use an indent of 4 characters with spaces not tabs. The line length is 110 characters.
For Eclipse there is a CXFCodeFormatter.xml you can use.

PROJECTS AND PACKAGES (PROPOSAL)

The package base for all CXF packages is "org.apache.cxf".

CXF is organized in maven projects. The groupId and artifactId of a project should relate to the base package name of the project. So for example
a maven project with groupId="org.apache.cxf" and artifactId="cxf-rt-transports-http-jetty" should only contain packages below "org.apache.cxf.transports.http-jetty.*".

It should always be avoided to have the same package name in more than one maven project as this will cause a lot of trouble in OSGi.

CYCLIC DEPENDENCIES (PROPOSAL)

Cyclic dependencies between packages should be avoided as they make maintenance harder and make it more difficult to understand the architecture. Some more details can be found at: Software erosion and package tangles

Every CXF committer can get a free license of the Architecture tool structure 101 which is very helpful in finding and avoiding cyclic dependencies. Just tell them that you are committer at Apache CXF to get a free license.

COMMONS XML SCHEMA

CXF makes internal use of the Apache Commons XML Schema. Web services are defined in terms of XML Schemas, and that library provides a data model.

The code from the XML Schema project provides almost no consistency checking. For each object in the model, it provides all the fields for all the possible settings. It does not provide or enforce a consistent discipline. It is very easy to write code that produces paradoxical schemas that have unexpected consequences. CXF includes some utilities that address some of these issues.

The XML Schema developers are very fond of 'final', so CXF does not have the option of using subclasses as a way to implement and enforce a model. Instead, CXF defines utilities and wrappers in org.apache.cxf.common.xmlschema.

Element Names and References

In the standard for XML Schema, an element can have a name (a QName) or a refName (another QName). Not both. Commons XML Schema provides the Element object with a QName, a Name, and a refName. The first is a string while the latter two are QNames.

The possibilities for inconsistency here are numerous. You can set the name to be different from the local part of the QName. You can set both a name and a refName.

To avoid this, use the functions in the XmlSchemaTools class. It provides functions for setting these three values.

CXF code should never call XmlSchemaElement.setName. The XMLSchemaTools functions will always set it, as a convenience value, to the local part of the refName or the QName, whichever is active.

CXF code should never call XmlSchemaElement.setQName or setRefName. Call XmlSchemaTools.
setElementQName or
XmlSchemaTools.setElementRefName. These function will throw an exception in the case of an inconsistency, and will also call setName
with the local part.

The Schema Collection

Each CXF service (or, if you prefer, WSDL), has a collection of XML schemas (schemata?). When one schema element refers to another with a qualified name, (e.g. type="bloop:Bleep"), the prefix is as defined for the current schema (or the WSDL as a whole), and the namespace referenced by the schema must be one of the schemas in the collection.

Commons XML Schema provides XmlSchemaCollection for this purpose. It has a number of, well, surprising characteristics. Generally, its authors gave much more thought to the case of reading schemas in from XML files than to creating them in a program. One example: each global type or element has to be presented to the collection API twice for it to be visible to the lookup APIs.

CXF wraps XmlSchemaCollection in the SchemaCollection class to deal with these items.

`FixedExtensionDeserializer` is a class that works around a specific XML Schema bug.