

This FAQ addresses frequently asked questions relating to the XML features of Oracle XML Developer's Kit (XDK) in Oracle Application Server (OracleAS) 10g.

Oracle XML Developer's Kit provides the basic building blocks for reading, manipulating, transforming and viewing XML documents. To provide a broad variety of deployment options, the Oracle XDK consists of the XML components in C, C++ and Java. This FAQ covers the following XDK components:

[XML Parsers](#): create and parse XML using industry standard DOM, SAX and JAXP interfaces.  
[XSLT Processors](#): transform or render XML into other text-based formats such as HTML.  
[XML Schema Processors](#): allow validate XML against the XML schemas defining the simple and complex datatypes.  
[XML SQL Utility](#): generates XML documents, DTDs and XML schemas from SQL queries.  
[XSQL Servlet](#): combines XML, SQL, and XSLT in the server to deliver dynamic web content.

For further information, please visit the OTN XML Center at <http://www.oracle.com/technology/tech/xml/index.html>, which contains additional samples, demos and documents.

## 1.0 XML Parser

### 1.1 Does Oracle XML parser support DTD caching? How do I set it up?

Yes, the XML parser in Oracle XDK Java provides the validating/non-validating DTD caching through the **setDoctype()** function. After you set the DTD object to the XML parser using this function, the XML parser will cache this DTD object for further XML parsing.

For example, if your application requires to validate multiple XML documents with the same DTD. After the first XML document is parsed, you can get the DTD from the XML parser and set it to the XML parser:

```
import oracle.xml.parser.v2.DOMParser;
import oracle.xml.parser.v2.XMLDocument;
import oracle.xml.parser.v2.DTD;
...
// Get an instance of the parser
DOMParser parser = new DOMParser();

// Parse the first document and set the DTD for caching
parser.setValidationMode(DOMParser.DTD_VALIDATION);
parser.setAttribute(DOMParser.USE_DTD_ONLY_FOR_VALIDATION, Boolean.TRUE);
parser.parse("{XML_Document_URL}");
DTD dtd =parser.getDoctype();
```

### **parser.setDoctype(dtd);**

```
// loop of XML parsing
for(...)
{
// XML Parsing with DTD Cache
}
```

The parser will cache this DTD and use it for parsing the XML documents thereafter. You also should set the `DOMParser.USE_DTD_ONLY_FOR_VALIDATION` attribute, if the cached DTD object is used only for validation by:

```
parser.setAttribute(DOMParser.USE_DTD_ONLY_FOR_VALIDATION, Boolean.TRUE);
```

Otherwise, the XML parser will copy the DTD object and add it to the DOM tree.

Additionally, the following example shows how to set the external DTDs:

```
parser.parseDTD("{DTD_URL}", "{Root_Element_Name}");
DTD dtd = parser.getDoctype();
parser.setDoctype(dtd);
```

```
// loop of XML parsing
for(...)
{
// XML Parsing with DTD Cache
}
```

### **1.2 How do I include binary data in an XML documents?**

There is no way to directly include binary data within the XML document. However, there are two work-arounds:

- 1/ The binary data could be referred as an external unparsed entity that resided in a different file.
- 2/ The binary data can be unencoded (meaning converting binary data into ASCII data) and be included in a CDATA section. The limitation on the encoding technique is that it only produces legal characters for the CDATA section.

### **1.3 How can make the XML Parser ignore the parsing of the <!DOCTYPE> tag?**

When loading XML documents with `<!DOCTYPE>` definitions, the parser will give errors if the DTD URLs can't be resolved. However, there are two ways to make the XML parsers ignores the definitions:

- 1/ Put `standalone="yes"` to `<?xml ...?>` declaration.
- 2/ Use the `XMLParser.setAttribute(XMLParser.STANDALONE, Boolean.TRUE)` function to treat the input file as if it had `standalone="yes"` in the XML declaration.

### **1.4 Which version of JAXP does XDK XML parser support?**

XDK XML Parser for Java support JAXP 1.2 in this release.



## 2.0 XSLT Processor

### 2.1 When should I use the XSLT extensions?

Before using the XSLT extensions, make sure you understand the following:

First, currently the extensions are only provided through the Java Binding Mechanisms in XDK XSLT Processor for Java.

Second, you should use extensions only if the built-in XSLT functions can't solve the problems such as using some math functions such as `cos()` and `sin()` functions.

Third, you should know the XSLT stylesheet is not portable after using the XSLT extensions. This is because XSLT extension binds to a specific programming language. In other words, if you define Java XSLT extensions, the XSLT stylesheet can only be used by the certain XSLT processors implemented in Java.

When using Oracle XSLT extensions, make sure you set the namespace of the extension class to be: `http://www.oracle.com/XSL/Transform/java/`

### 2.2 What are the built-in extensions in the Oracle XDK XSLT Processor for Java?

Oracle XDK XSLT Processor for Java provides two built-in extension functions:

`<ora:output>`: where `xmlns:ora="http://www.oracle.com/XSL/Transform/java"`.

This element can be used as a top-level element or within the XSL templates. When it is used as a top-level element, it is similar to the `<xsl:output>` and can have all the attributes allowed by the `<xsl:output>`. Additionally, it has an "name" attribute used as an identifier.

When `<ora:output>` is used in an XSL template, it can only have two attributes, the "use" and the "href" attribute. The "use" attribute specifies the name of the top-level `<ora:output>` to be used, and the "href" gives the output URL for the sub-tree of the XSLT result.

`ora:node-set`: where `xmlns:ora="http://www.oracle.com/XSL/Transform/java"`.

The `ora:node-set` function converts a result tree fragment into a node-set.

### 2.3 How can I check whether the XML element contains child nodes or child attributes?

You can use the following syntax to check whether the XML element contains child nodes:

```
<xsl:if test="not(node())">
<xsl:attribute name="null">yes</xsl:attribute>
</xsl:if>
```

You can use the following syntax to check whether the XML element contains attribute nodes:

```
<xsl:if test="not(attribute::node())">
<xsl:attribute name="attrnull">yes</xsl:attribute>
</xsl:if>
```

With these, you can specify the \*null indicators\* as shown in the following XSLT stylesheet:

```

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="/mytaglist/mytag">
<xsl:element name="mytag">
<xsl:if test="not(node())">
<xsl:attribute name="null">yes</xsl:attribute>
</xsl:if>
<xsl:if test="not(attribute::node())">
<xsl:attribute name="attnull">yes</xsl:attribute>
</xsl:if>
<xsl:for-each select="@*">
<xsl:attribute name="{name()}"><xsl:value-of select="."/></xsl:attribute>
</xsl:for-each>
<xsl:apply-templates/>
</xsl:element>
</xsl:template>
<xsl:template match="node()|@*|comment()|processing-instruction()">
<xsl:copy>
<xsl:apply-templates select="@*|node()"/>
</xsl:copy>
</xsl:template>
</xsl:stylesheet>

```



### 3.0 XML Schema Processor

#### 3.1 Does XDK XML schema processor fully support W3C XML Schema recommendation 1.0?

Yes.

#### 3.2 What is the example calling sequence of XML schema processor in C++?

Here is the example calling sequence:

- 1/ XMLSchema.initialize(): initializes the process
- 2/ Parse XML documents as an input to XML schema processor
- 3/ XMLSchema.validate(): validate the parsed XML documents
- 4/ XMLSchema.terminate(): end the process



### 4.0 XML SQL Utility

#### 4.1 Can XML SQL Utility insert XML data across tables?

XML-SQL Utility (XSU) can only store to a single table. It maps a canonical representation of the XML documents into any table/view. However, there are ways to store XML across tables with the XSU. First approach is that you use XSLT to transform the XML document into multiple XML documents and insert them separately. Second approach is that you define database views over multiple tables (object views if needed) and then do the inserts into the view. If the view is inherently non-updatable (because of complex joins,...), then you can create INSTEAD-OF triggers over the views for managing the insertion.

#### 4.2 How can I load data stored in XML attributes into database tables with XSU?

XML SQL Utility assumes the canonical mapping of the XML document and the database

schema when loading data. This requires all data stored in XML elements. You have to use XSLT to transform the XML attributes into XML elements before inserting the data into database when using XSU.

#### **4.3 Does XML SQL Utility commit after it's done with the inserting/deleting/updating?**

By default, the XML SQL Utility does not explicit commit. If the autocommit is on (default for the JDBC connection) then after each batch of statement (using the "setBatchSize" feature) the executions of a commit happen. The user can override this by turning autocommit off and then specifying after how many statement executions should a commit occur which can be done using the "setCommitBatch" feature (when batchSize=1).

Finally, what happens if an error occurs? Well, the XSU rollbacks either to the state the target table was before the particular call to the XSU, or the state right after the last commit made during the current call to the XSU.



## **5.0 XSQL Servlet**

### **5.1 What is Oracle XSQL Servlet?**

Oracle XSQL Servlet is part of XDK Java components and provides servlet engine, Java APIs and a command-line utility. To run the XSQL command-line utility, you need to set the following libraries and the XSQLConfig.xml file in the JAVA CLASSPATH:

xmlparserv2.jar: Oracle XDK XML Parser for Java  
classes12.jar: JDBC  
xsu12.jar: XML SQL Utility  
xml.jar: XSQL Servlet

The command is:

```
java oracle.xml.xsql.XSQLCommandLine xsql xsqlFileURI [outFileName] [param1=value1 ...  
paramN=valueN]
```

To run XSQL Servlet in Web servers, you need to do specific setups for XSQL servlet in the Web servers. You can find the information in the Release Notes of XSQL.

### **5.2 Can we perform DML operations using XSQL servlet?**

Yes. This is supported using the <xsql:dml> tag or using the <xsql:insert-request> tag.



## **ORACLE FUSION MIDDLEWARE**

Oracle Application Server 10g: Oracle XML Developer's Kit FAQ

September, 2005

Author: Jinyu Wang

Oracle Corporation  
World Headquarters  
500 Oracle Parkway  
Redwood Shores, CA 94065  
U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

oracle.com

Copyright © 2005, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.