

Building Server-Side XML Schema Validation

August 2001

With the advent of XML Schema there is the requirement to validate XML documents against their associated schemas. This article gives an introduction to the XML schema validation process with the XDK for Java and discusses how to build an Oracle Java Stored Procedure to perform the schema validation on the server-side of the Oracle Database. The included sample also demonstrates the deployment procedure for Java Stored Procedures.

XML Schema Validation can provide a flexible and portable form of data validation for use in your applications. You may implement the XML validation process in your client-side or mid-tier applications but if you want either:

- Absolute control of data validation whenever the data is updated/inserted
- Make use of the data management capability of the Oracle database

Then putting your data validation process inside a trigger or your PL/SQL procedures on the server-side is a good solution. Since there is not a built-in PL/SQL API to do XML Schema validation, we can create one using Java Stored Procedures.

The first step in building a Java Stored Procedure for XML Schema validation is to select the components and decide the environment requirements. The components you'll need are:

- XML Schema Processor for Java [xschema.jar]
- XML Parser for Java [xmlparserv2.jar]

Both of these are part of Oracle XML Developer's Kits for Java. See the [Resource](#) section to download the latest version of XDK for Java. The Oracle Database (8.1.6 version and above) is also needed as these versions fully support Java Stored Procedures.

If you download the XDK for Java and have an Oracle 8.1.6 Database or above, you can follow the following steps to build up the Java Stored Procedure and take advantage of XML Schema for data validation.

- Create the Java Class with the Schema Validation Function
- Load and resolve the Java Class into Oracle Database
- Publish the Java by defining the specification

Create the Java Classes for XML Schema Validation

To build the Java Class for XML Schema Validation, two XDK packages, XML Schema Processor and XML Parser are needed:

```
import oracle.xml.parser.schema.*;
import oracle.xml.parser.v2.*;
```

To be able to accept the inputs from PL/SQL, we need another package

```
import oracle.sql.CHAR;
```

You need to set xmlparserv2.jar, xschema.jar and classes12.zip to the CLASSPATH. The JDBC library classes12.zip is for JDK 1.2.x. If you are using JDK 1.1.x, classes111.zip is required to be used.

The SchemaUtil Class is shown below:

```
public class SchemaUtil
```

content

[Create the Java Classes for XML Schema Validation](#)

[Loading and Resolving the Java Class](#)

[Publishing the Java Class by Defining the Specification](#)

[Example Using the Stored Procedures](#)

[Resources](#)

```

{
    public static String validation(CHAR xml, CHAR xsd)
    throws Exception
    {
        //Build Schema Object
        XSDBuilder builder = new XSDBuilder();
        byte [] docbytes = xsd.getBytes();
        ByteArrayInputStream in = new ByteArrayInputStream(docbytes);
        XMLSchema schemadoc = (XMLSchema)builder.build(in,null);
        //Parse the input XML document with Schema Validation
        docbytes = xml.getBytes();
        in = new ByteArrayInputStream(docbytes);
        DOMParser dp = new DOMParser();
        // Set Schema Object for Validation
        dp.setXMLSchema(schemadoc);
        dp.setValidationMode(XMLParser.SCHEMA_VALIDATION);
        dp.setPreserveWhitespace (true);
        StringWriter sw = new StringWriter();
        dp.setErrorStream (new PrintWriter(sw));
        try
        {
            dp.parse (in);
            sw.write("The input XML parsed without errors.\n");
        }
        catch (XMLParseException pe)
        {
            sw.write("Parser Exception: " + pe.getMessage());
        }
        catch (Exception e)
        {
            sw.write("NonParserException: " + e.getMessage());
        }
        return sw.toString();
    }
}

```

This class defines a single method, **validation**, which does the XML Schema validation for the input XML document and returns the error messages.

To compile the class, use following command line:

```
javac SchemaUtil.java
```

This produces the compiled Java class, SchemaUtil.class.

Loading and Resolving the Java Class

With the utility `loadjava`, you can upload the Java source, class, and resource files into an Oracle database, where they are stored as Java schema objects. You can run `loadjava` from the command line or from an application, and you can specify several options including a resolver. Make sure you have `$ORACLE_HOME/bin` in your System Path to be able to run `loadjava`.

Before loading the `SchemUtil.class` into the Database, we need to check if the correct version of the two dependent XDK Packages are loaded into the logon database schema (in this case `xdkdemo/xdkdemo`).

```
connect xdkdemo/xdkdemo
```

To check the status of the `oracle.xml.parser.v2.DOMParser` class, you can use the following SQL statement:

```
SELECT SUBSTR(dbms_java.longname(object_name),1,35) AS class, status
FROM all_objects
WHERE object_type = 'JAVA CLASS'
AND object_name = dbms_java.shortname('oracle/xml/parser/v2/DOMParser');
```

If you see the result:

```
CLASS                                STATUS
-----
oracle/xml/parser/v2/DOMParser        VALID
```

then the Oracle XML Parser for Java is already installed and ready to be used.

If you see the above result, but the status is `INVALID`, try the command:

```
ALTER JAVA CLASS _oracle/xml/parser/v2/DOMParser Resolve
```

If the verification procedure produces the SQL*Plus message “no rows selected”, you need to load the XML Parser into Database by:

```
loadjava -resolve -verbose -user xdktemp/xdktemp xmlparserv2.jar
```

If the parser is installed, then you don't need to complete any further installation steps. The SQL command for status checking will be:

```
SELECT SUBSTR(dbms_java.longname(object_name),1,35) AS class, status
FROM all_objects
WHERE object_type = 'JAVA CLASS'
      AND object_name = dbms_java.shortname('oracle/xml/parser/schema/XMLSchema');
```

Before loading the `SchemaUtil.class`, make sure that the loaded XML Parser has the same version with which you compiled the `SchemaUtil.class`. The following code can be used to check the current version of the loaded Oracle XML Parser:

```
CREATE OR REPLACE FUNCTION XMLVersion RETURN VARCHAR2
IS LANGUAGE JAVA NAME
'oracle.xml.parser.v2.XMLParser.getReleaseVersion() returns java.lang.String';
/
CREATE OR REPLACE Procedure getXMLVersion AS
begin
  dbms_output.put_line(XMLVersion());
end;
/
```

Then by issuing the command:

```
SQL> set serveroutput on
SQL> exec getXMLVersion;
```

You should receive the following result:

```
Oracle XDK Java      9.0.2.0.0A      Beta
```

If the version doesn't match, you need to drop the package and reload it. To drop the package, you can issue following command line:

```
dropjava -verbose -user xdktemp/xdktemp xmlparserv2.jar xschema.jar
```

Once all of the versions are synced, we can finally load the SchemaUtil.class by:

```
loadjava -resolve -verbose -user xdktemp/xdktemp SchemaUtil.class
```

Publishing the Java Class by Defining the Specification

For each Java method callable from SQL, you must write a call specification in Java, which exposes the method's top-level entry point to the Oracle server.

```
CREATE OR REPLACE FUNCTION SchemaValidation(xml IN VARCHAR2,xsd IN VARCHAR2)
return varchar2
IS LANGUAGE JAVA NAME
'SchemaUtil.validation(oracle.sql.CHAR,oracle.sql.CHAR) returns java.lang.String';
```

Now the Java stored procedure specification is created, both SQL and PL/SQL can call it as if it were PL/SQL function.

Example Using the Stored Procedures

You can call Java stored procedures from SQL DML statements, PL/SQL blocks, and PL/SQL subprograms. Using the SQL CALL statement, you can also call them from the top level (from SQL*Plus, for example) and from database triggers. The following example shows how to do XML Schema Validation using the created Java stored procedure.

1. Creating a Database Schema to store XML and XML Schema Documents.

```
create table schema_tab(id number, xsd VARCHAR2(4000));
create table xml_tab(id number, xml VARCHAR2(4000));
```

2. Loading the XML Schema Document into the Database

You can just use the SQL commands to insert the data show in DBData.sql, like:

```
Insert into schema_tab(1, '[XML schema]');
```

3. Calling the Java Stored Procedure to Validate the input XML Document inside the trigger of the xml_tab table

```
-- Write XML Buffer to Output
```

```
CREATE OR REPLACE PROCEDURE printBufferOut(xmlstr IN OUT NOCOPY VARCHAR2) AS
BEGIN
    line    VARCHAR2(20000);
    nlpos   INTEGER;
    LOOP
```

```

EXIT WHEN xmlstr is null;
nlpos := instr(xmlstr,chr(10));
line := substr(xmlstr,1,nlpos-1);
-- print line
IF(length(line) <250) THEN
  dbms_output.put_line(' | '||line);
ELSE
  dbms_output.put(' | ');
  LOOP
    EXIT WHEN line is null;
    dbms_output.put_line(substr(line,1,250));
    line := substr(line,250+1);
  END loop;
END if;
xmlstr := substr(xmlstr,nlpos+1);
IF (nlpos = 0) THEN
  dbms_output.put_line(' | '||xmlstr);
  EXIT;
END if;
END LOOP;
END printBufferOut;
/
show errors;
CREATE OR REPLACE PROCEDURE dbvalid(xmlid IN NUMBER, xsdid IN NUMBER) IS
  p_xml varchar2(4000);
  p_xsd varchar2(4000);
  p_out varchar2(4000);
begin
  select xml into p_xml from xml_tab where id=xmlid;
  select xsd into p_xsd from schema_tab where id=xsdid;
  p_out := SchemaValidation(p_xml,p_xsd);
  printBufferOut(p_out);
end;
/

```

For the date with the [xdksample_093001.zip](#) you can execute the command and get following result:

```

SQL> exec dbvalid(1,1);
| The input XML parsed without errors.
PL/SQL procedure successfully completed.
SQL> exec dbvalid(2,1);
| | <Line 5, Column 42>: XSD-2023: (Error) Invalid value of attribute:
'1999-11-31'
| <Line 21, Column 27>: XSD-2105: (Error) Identity constraint validation error:

```

'Key sequence not found in key reference'.

| | Parser Exception: Invalid value of attribute: '1999-11-31'

PL/SQL procedure successfully completed.

You can now use this Java Stored Procedure to Validate the XML document using PL/SQL.

Resources

· Download XML Developers Kit for Java from ORACLE Technology Network:

<http://technet.oracle.com/tech/xml/xdkhome.html>

· Source code for the demo: [xdksample_093001.zip](#)