Xerces-J Documentation

Table of Contents

1
Xerces Java Parser Readme

Xerces Java Parser 1.1.0 Release

The Xerces Java Parser 1.1.0 supports XML 1.0 recommendation and contains advanced parser functionality, such as XML Schema, DOM Level 2 version 1.0, and SAX Version 2, in addition to supporting the industry-standard DOM Level 1 and SAX version 1 APIs.

Note that because some of the standards are still not complete, the stable API will definitely be different from its current form in Xerces-J 1.1.0. This is your chance to give us feedback on the features that are important to you, and let us know whether the APIs that we are providing are the right ones. Please direct your feedback to the Xerces-J mailing list.

The 1.1.0 release benefits from the xml.apache.org collaboration. It includes Assaf Arkin's serialization and HTML DOM contributions. See the release information.

License Information

The Xerces-J 1.1.0 release is available in both source code and precompiled binary (JAR files) form. Both Xerces-J packages are made available under the Apache Software License.

Applications of the Xerces-J Parser

The rich generating and validating capabilities allow the Xerces-J Parser to be used for:

- Building XML-savvy Web servers.
- The next generation of vertical applications which will use XML as their data format.
- On-the-fly validation for creating XML editors.
- Ensuring the integrity of e-business data expressed in XML.
- Building truly internationalized XML applications.

XML Schema (alpha) Support

This release includes preliminary support for the W3C XML Schema Language. The Schema page contains a complete description of the schema capabilities of this release. We intend to track the W3C XML Schema Language in subsequent updates of Xerces-J.

DOM Level 2 (beta) Support

This release includes support for the DOM Level 2. As of this writing, the DOM Level 2 specification is now a Candidate Recommendation.

SAX 2 Support

This release includes support for the SAX Version 2 API's which have been finalized.

Configuration Mechanism

Xerces-J 1.1.0 uses a collection of methods to configure various settings in the parser. This release

includes a new mechanism for setting parser switches. This mechanism uses the SAX2 Configurable interface. We have defined a series of properties and features for the Xerces-J options.

2
Installation

Unpacking the files

Xerces-J is packaged as a ZIP file for all platforms and operating systems. You can run the Java jar command to unpack the distribution.

- jar xf Xerces-J-bin.1.1.0.zip
- jar xf Xerces-J-src.1.1.0.zip
- This command creates a "xerces-1_1_0" sub-directory in the current directory containing all the files.

Files in the binary package release

LICENSE	License for Xerces-J
Readme.html	Web page redirect to docs/html/index.html
xerces.jar	Jar file containing all the parser class files
xercesSamples.jar	Jar file containing all sample class files
data/	Directory containing sample XML data files
docs/html/	Directory containing documentation
docs/apiDocs/	Directory containing Javadoc API for parser framework

Note: *To use Xerces-J you do not need the source files.*

Files in the source package release

LICENSE	License for Xerces-J
Makefile	Top level Makefile -- read README file before building
README	Build instructions
Readme.html	Web page redirect required for building documentation
STATUS	Current source code status information
data/	Directory containing sample XML data files
docs/	Directory containing documentation, in XML form
samples/	Directory containing source code for samples
src/	Directory containing source code for parser and supplemental APIs

3
API Documentation

Javadoc Generated Documentation

Xerces Java Parser comes packaged with API documentation for SAX and DOM, the two most common interfaces for programming XML. In addition, we provide documentation for classes that are not part of the SAX and DOM API's, but are useful for writing Xerces-J programs.

This documentation is generated automatically from the Javadoc-style comments inside the source files. Click on one of the links below to go to the appropriate API documentation.

Xerces-J API Documentation

- Full API documentation
- Hierarchy for all the packages

Package org.apache.xerces.parsers

Classes
- DOMParser
- SAXParser

Interfaces
- Attr
  The Attr interface represents an attribute in an Element object
- CDATASection
  CDATA sections are used to escape blocks of text containing characters that would otherwise be regarded as markup
- CharacterData
  The CharacterData interface extends Node with a set of attributes and methods for accessing character data in the DOM
- Comment
  This represents the content of a comment
- Document
  The Document interface represents the entire HTML or XML document
- DocumentFragment
  DocumentFragment is a "lightweight" or "minimal" Document object
- DocumentType
  Each Document has a doctype attribute whose value is either null or a DocumentType object
- DOMImplementation
  The DOMImplementation interface provides methods for performing operations independent of a particular document object model instance
- Element

The majority of objects (apart from text) in a document are Element nodes
- Entity
  This interface represents an entity, either parsed or unparsed, in a XML document
- EntityReference
  EntityReference objects may be inserted into the structure model
- NamedNodeMap
  Objects implementing the NamedNodeMap interface are used to represent collections of nodes that can be accessed by name
- Node
  The Node interface is the primary datatype for the entire Document Object Model
- NodeList
  The NodeList interface provides the abstraction of an ordered collection of nodes
- Notation
  This interface represents a notation declared in the DTD
- ProcessingInstruction
  The ProcessingInstruction is a way to keep processor-specific information in the text of the document
- Text
  The Text interface represents the textual content (termed character data in XML) of an Element or Attr

Exceptions
- DOMException
  Encapsulate an "exceptional" DOM error

Package org.xml.sax

Classes
- HandlerBase
  Default base class for handlers
- InputSource
  A single input source for an XML entity

Interfaces
- AttributeList
  Interface for an element's attribute specifications
- DocumentHandler
  Receive notification of general document events
- DTDHandler
  Receive notification of basic DTD-related events
- EntityResolver
  Basic interface for resolving entities
- ErrorHandler
  Basic interface for SAX error handlers
- Locator
  Interface for associating a SAX event with a document location
- Parser
  Basic interface for SAX (Simple API for XML) parsers

Exceptions
- SAXException
  Encapsulate a general SAX error or warning
- SAXParseException

Encapsulate an XML parse error or warning

Package org.xml.sax.helpers

Classes

- AttributeListImpl
  Convenience implementation for AttributeList
- LocatorImpl
  Convenience implementation for Locator
- ParserFactory
  Java-specific class for dynamically loading SAX parsers

Package org.apache.xerces.dom.traversal

Classes

- DocumentTraversal
  DocumentTraversal contains methods that creates Iterators to traverse a node and its children
- DocumentTWIF
  DocumentTWIF contains methods that create Iterators to traverse a node and its children
- NodeFilter
  Filters are objects that know how to "filter out" nodes
- NodeIterator
  NodeIterators are used to step through a set of nodes
- TreeWalker
  TreeWalker objects are used to navigate a document tree or subtree using the view of the document

4
Samples

Introduction

There are two new samples to test and demonstrate the new DOM2 Traversal implementation:

- DOMCount
- SAXCount
- DOMWriter
- SAXWriter
- DOMFilter
- IteratorView
- TreeWalker
- TreeViewer

Caveats

Note: *Xerces-J: Running the sample applications requires that you have already loaded the Xerces-J software on your computer.*

Note: *Java: Running the sample applications require that your computer has a correctly installed JDK. If you do not already have a JDK already on your computer download one from Sun's Java website: http://java.sun.com or from IBM's website http://www.ibm.com/developer/java/ where you can find an "Enhanced Windows JDK" that is optimized for the Windows platform. The sample applications described in the following pages support Java 1 - JDK 1.1.6. 1.1.7, 1.1.8 or Java 2 - JDK 1.2.2.*

Note: *UNIX: Command lines in the pages linked below use the Windows path separator ';' (semicolon) and directory separator '\' (backslash).. On UNIX, use the ':' (colon) character to separate the JAR files in the classpath, and replace Windows directory separator '\' (backslash) with '/' (forward slash).*

DOMCount Sample

DOMCount parses your input file, and outputs the total parse time, along with counts of elements, attributes, text characters, and ignorable whitespace characters. DOMCount displays errors and warnings that occur during parsing.

DOMCount uses either the validating or non-validating DOM parser.

SAXCount Sample

SAXCount parses your input file, and outputs the total parse time, along with counts of elements, attributes, text characters, and ignorable whitespace characters. SAXCount displays errors and warnings that occur during parsing.

SAXCount uses either the validating or non-validating SAX parser.

DOMWriter Sample

DOMWriter parses a file, and prints it out in XML format. The command line option, -c, is used to print files in "canonical" XML format, so that two XML documents can be compared. They also display any errors or warnings that occurred during the parse. DOMWriter uses either the validating or non-validating DOM parser. DOMWriter also provides a feature to set the output Java encoding through the -e switch.

SAXWriter Sample

SAXWriter parses a file, and prints it out in XML format. The command line option, -c, is used to print files in "canonical" XML format, so that two XML documents can be compared. They also display any errors or warnings that occurred during the parse. SAXWriter uses either the validating or non-validating SAX parser.

DOMFilter Sample

DOMFilter shows you how to search for specific elements in your XML document. It uses `getElementsByTagName()` to traverse the DOM tree, looking for elements or attributes that match your specification.

IteratorView Sample

IteratorView is an interactive UI sample that displays the DOM tree. It shows the progress of the iteration by moving the selection within the DOM tree. Buttons act as a control panel, allowing the user to interactively iterate through the tree, remove nodes, add nodes, and view the results immediately in the tree.

The `IteratorView` uses an example filter, `NameNodeFilter`, that can be controlled from the UI and a `DOMTreeFull` class that displays the full DOM tree with all the nodes.

TreeWalker Sample

TreeWalkerviewView is an interactive UI sample that displays the DOM tree. It show the progress of the tree traversal by moving the selection within the DOM tree. Buttons act as a control panel, allowing the user to interactively traverse the tree, remove nodes, add nodes, and view the results immediately in the tree.

The `TreeWalkerviewView` uses an example filter, `NameNodeFilter`, that can be controlled from the UI and a `DOMTreeFull` class that displays the full DOM tree with all the nodes.

Treeviewer Sample

TreeViewer displays the input XML file in a graphical tree-style interface. It will also highlight lines have well-formedness or validation errors.

SAX/DOMCount Samples

Running SAXCount and DOMCount

SAXCount and DOMCount invoke the parser on an XML document, and print out information about the document. By default, SAXCount creates a non-validating SAX parser and DOMCount creates a validating DOM parser. They both count the number of elements, attributes, text characters, and ignorable whitespace characters in the document and display the amount of time it takes to complete the task.

The command lines below expect the current directory to be the directory containing the JAR file.

Requirements:

- Xerces-J is loaded on your computer.
- JDK is loaded on your computer.

Source code:

- SAXCount.java
- DOMCount.java

SAXCount

To run SAXCount:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `SAXCount` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then the `SAXCount` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: drive.

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%
cd c:\xerces-1_1_0
java sax.SAXCount data\personal.xml
```

Switches:

SAXCount also allows you to change the default behavior using the following command line flags:

- -p Specify the parser class to be used.
  The available parsers are:
  - org.apache.xerces.parsers.SAXParser [default parser]
- -h Print SAXCount help information. [default is no help]
- -v Turn on validation

Running SAXCount with the default settings is equivalent to running SAXCount like this (type this in as one long command line):

```
java sax.SAXCount -p org.apache.xerces.parsers.SAXParser
    data\personal.xml
```

Bringing up the help information:

```
java sax.SAXCount -h
```

Note: *Parse your own XML file instead of data\personal.xml*

DOMCount

To run DOMCount:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `DOMCount` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then the `DOMCount` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: dirve.

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%
cd c:\xerces-1_1_0
java dom.DOMCount data\personal.xml
```

Switches:

DOMCount also allows you to change the default behavior via the following command line flags (type this in as one long command line):

- -p Specify the parser class to be used.
  The available parsers are:
  - dom.wrappers.NonValidatingDOMParser
  - dom.wrappers.DOMParser [default parser]
- -h Print DOMCount help information. [default is no help]

Running DOMCount with the default settings is equivalent to running `DOMCount` like this:

```
java dom.DOMCount -p dom.wrappers.DOMParser
    data\personal.xml
```

Bringing up the help information:

```
java dom.DOMCount -h
```

Note: *Parse your own XML file instead of data\personal.xml*

SAX/DOMWriter Samples

Running SAXWriter and DOMWriter

`SAXWriter` and `DOMWriter` parse your input file, and print it out in XML format. A command line option can be used to print in a "canonical" XML format so the output can be used to compare XML documents. SAXWriter and DOMWriter also display any errors or warnings that occurred during the parse.

`SAXWriter` uses either the validating or non-validating SAX parser. `DOMWriter` uses either the validating or non-validating DOM parser.

`DOMWriter` provides a -e switch to set the output Java encoding.

The command lines below expect the current directory to be the directory containing the JAR file.

Requirements:

- Xerces-J is loaded on your computer.
- JDK is loaded on your computer.

Source code:

- SAXWriter.java
- DOMWriter.java

SAXWriter

To run SAXWriter:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `SAXWriter` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then the `SAXWriter` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: drive.

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%
cd c:\xerces-1_1_0
java sax.SAXWriter data\personal.xml
```

Switches:

SAXCount also allows you to change the default behavior using the following command line flags:

- -p Specify the parser class to be used.
  The available parsers are:
  - org.apache.xerces.parsers.SAXParser [default parser]
- -h Print SAXWriter help information. [default is no help]
- -c Output in canonical format. [default is normal format]

Running `SAXWriter` with the default settings is equivalent to running `SAXWriter` like this (type this in as one long command line):

```
java sax.SAXWriter -p org.apache.xerces.parsers.SAXParser
    data\personal.xml
```

Bringing up the help information:

```
java sax.SAXWriter -h
```

Print in canonical format:

```
java sax.SAXWriter -c  data\personal.xml
```

Note: *Parse your own XML file instead of data\personal.xml*

DOMWriter

To run DOMWriter:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `DOMWriter` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then the `DOMWriter` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: dirve.

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%
cd c:\xerces-1_1_0
java dom.DOMWriter data\personal.xml
```

Switches:

DOMCount also allows you to change the default behavior via the following command line flags (type this in as one long command line):

- -p Specify the parser class to be used.
  The available parsers are:
  - dom.wrappers.NonValidatingDOMParser
  - dom.wrappers.DOMParser [default parser]
- -h Print DOMWriter help information. [default is no help]
- -c Output in canonical format. [default is normal format]
- -e encodingName Output using the specified encoding. [default is UTF8]

Running `DOMWriter` with the default settings is equivalent to running `DOMWriter` like this:

```
java dom.DOMWriter -p dom.wrappers.DOMParser
    data\personal.xml
```

Bringing up the help information:

```
java dom.DOMWriter -h
```

Searching for elements:

```
java dom.DOMWriter -c data\personal.xml
```

Running DOMWriter with the -e switch and no encoding specified will print out a list of valid encoding names:

```
java dom.DOMWriter -e
```

Prints the following:

```
Java Encoding one of (case sensitive):
  Default
  8859_1 8859_2 8859_3 8859_4 8859_5 8859_6 8859_7
  8859_8 8859_9 Cp037 Cp273 Cp277 Cp278 Cp280
  Cp284 Cp285 Cp297 Cp420 Cp424 Cp437 Cp500
  Cp737 Cp775 Cp838 Cp850 Cp852 Cp855 Cp856
  Cp857 Cp860 Cp861 Cp862 Cp863 Cp864 Cp865
  Cp866 Cp868 Cp869 Cp870 Cp871 Cp874 Cp875
```

```
    Cp918 Cp921 Cp922 Cp930 Cp933 Cp935 Cp937
    Cp939 Cp942 Cp948 Cp949 Cp950 Cp964 Cp970
    Cp1006 Cp1025 Cp1026 Cp1046 Cp1097 Cp1098 Cp1112
    Cp1122 Cp1123 Cp1124 Cp1250 Cp1251 Cp1252 Cp1253
    Cp1254 Cp1255 Cp1256 Cp1257 Cp1258 Cp1381 Cp1383
    Cp33722 MS874 DBCS_ASCII DBCS_EBCDIC EUC EUCJIS GB2312
    GBK ISO2022CN_CNS ISO2022CN_GB JIS JIS0208 KOI8_R KSC5601
    MS874 SJIS SingleByte Big5 CNS11643 MacArabic
    MacCentralEurope MacCroatian MacCyrillic MacDingbat
    MacGreek MacHebrew MacIceland MacRoman MacRomania
    MacSymbol MacThai MacTurkish MacUkraine SJIS Unicode
    UnicodeBig UnicodeLittle UTF8
```

Note: *Parse your own XML file instead of data\personal.xml*

DOMFilter Sample

Running DOMFilter

`DOMFilter` parses an XML document, searching for specific elements by name, or elements with specific attributes.

Requirements:

- Xerces-J is loaded on your computer.
- JDK is loaded on your computer.

Source code:

- DOMFilter.java

DOMFilter

To run DOMFilter:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `DOMFilter` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then the `DOMFilter` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: dirve.

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%
cd c:\xerces-1_1_0
java dom.DOMFilter data\personal.xml
```

Switches:

`DOMFilter` also allows you to change the default behavior using the following command line flags (type this in as one long command line):

- -p Specify the parser class to be used.
  The available parsers are:
  - dom.wrappers.NonValidatingDOMParser
  - dom.wrappers.DOMParser [default parser]
- -h Print DOMCount help information. [default is no help]
- -e Specify the name of the element to search for. [defaults to matching all elements]
- -a Specify the name of the attribute to search for. [defaults to matching all attributes]

Running `DOMFilter` with the default settings is equivalent to running `DOMFilter` like this:

```
java dom.DOMFilter -p dom.wrappers.DOMParser
    data\personal.xml
```

Bringing up the help information:

```
java dom.DOMFilter -h
```

Searching for elements:

```
java dom.DOMFilter -e family data\personal.xml
```

Search for attributes:

```
java dom.DOMFilter -a subordinates data\personal.xml
```

Note: *Parse your own XML file instead of data\personal.xml*

IteratorView Sample

IteratorView

The `IteratorView` is an interactive UI sample that displays the DOM tree. It shows the progress of the iteration by moving the selection within the DOM tree. Buttons act as a control panel, allowing the user to interactively iterate through the tree, remove nodes, add nodes, and view the results immediately in the tree.

The `IteratorView` uses an example filter, `NameNodeFilter`, that can be controlled from the UI and a `DOMTreeFull` class that displays the full DOM tree with all the nodes.

The controls are called through to to the corresponding iterator function. If you are familiar with the DOM Level 2 Traversal specification, these controls are fairly easy to understand

Iterator Group

- Next - calls the `next()` functions and selects the next node in the tree.
- Previous - calls the `previous()` function and selects the previous node in the tree.

Selected Node Group

- `remove` - remove the selected Node and update the DOM tree. You must press an iterator button to see next or previous node selection.
- `add` - add a text node, to see the results of adding a node on the iterator. position. Again you must first press next or previous

Iterator Settings Group

- `createNodeIterator` - calls the factory method to create a new iterator with the corresponding `whatToShow` and `NameNodeFilter` settings.
  - The root is set to be the root of the document, so it starts at the top level each time.
- `whatToShow` - you can singly or multiply select values and the iterator is constrained to showing these types.
- `NodeNameFilter` - An empty string is converted to null and given to the `NodeNameFilter` class.
  - An empty string matches all nodes.
  - A non-empty string is forced to match node names.

Running IteratorView

```
    java dom.traversal.IteratorView <fileName>
```

TreeWalker Sample

TreeWalker

The `TreeWalkerviewView` is an interactive UI sample that displays the DOM tree. It show the progress of the tree traversal by moving the selection within the DOM tree. Buttons act as a control panel, allowing the user to interactively traverse the tree, remove nodes, add nodes, and view the results immediately in the tree.

The `TreeWalkerviewView` uses an example filter, `NameNodeFilter`, that can be controlled from the UI and a `DOMTreeFull` class that displays the full DOM tree with all the nodes.

The controls are called through to to the corresponding `TreeWalker` function. If you are familiar with the DOM Level 2 Traversal specification, these controls are fairly easy to understand.

Document Order Traversal Group

- Next - calls the `next()` functions and selects the next in the tree.
- Previous - calls the `previous()` function a and selects the previous node in the DOM tree.

Walk Group

Parent, Previous Sibling, Next Sibling, First Child, Last Child - call the corresponding function in TreeWalker and show the result as a selected Node.

Selected Node Group

- current - set the current node to the selected node.
- remove - remove the selected node and update the tree. You must press a button to see next or previous node selection.
- add - add a text node. You must press a button to see next or previous node selection.

Filter Settings Group

- `createNodeTreeWalker` - calls the factory method to create a new `TreeWalker` with the corresponding `whatToShow` and `NodenameFilter` settings. The selected node becomes the TreeWalker root.
- `whatToShow` - you can singly or multiply select these values by pressing the control key and the `TreeWalker` is constrained to these types.
- `NodeNameFilter` - an empty string is converted to null and given to the `NodeNameFilter` example filter class provided.
  - An empty string (null) matches ALL nodes.
  - A non-empty string is forced to match node names.

Running TreeWalker

```
    java dom.traversal.TreeWalkerView <fileName>
```

TreeViewer Sample

Running TreeViewer

`TreeViewer` displays the input file in a graphical tree based interface. This sample highlights the error handling capabilities of the parser, demonstrating how the parser can recover from many types of common errors.

Requirements:

- Xerces-J is loaded on your computer.
- Either:
  - JDK 1.1.8 and Swing1.1.1 are is loaded on your computer.
- Or:
  - Java 2 (jdk1.2.2) is loaded on your computer.

Source code:

- TreeViewer.java
- TreeView.java
- DOMTree.java
- DefaultImages.java

TreeViewer

To run TreeViewer:

1. open up a MS-DOS command line window
2. set the path to the jdk\bin directory
3. change directory to the latest xerces-1_1_0 directory
4. invoke the `TreeViewer` parser

On Windows:

The easiest way to do this is to create a .bat file using the Notepad editor. Then `TreeViewer` can be invoked by double clicking on the file name or icon. The following command lines assume that both the jdk and the xerces-1_1_0 directories are located directly below the c: dirve.

With jdk1.1.8:

```
set path=c:\jdk1.1.8\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%;
set classpath=c:\Swing-1.1.1\swingall.jar;%CLASSPATH%;
cd c:\xerces-1_1_0
java dom.TreeViewer data\personal.xml
```

With jdk1.2.2:

Swing is included in the Java 2 release and it doesn't required a separate reference.

```
set path=c:\jdk1.2.2\bin;%PATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0.jar;%CLASSPATH%
set classpath=c:\xerces-1_1_0\xerces-1_1_0Samples;%CLASSPATH%;
cd c:\xerces-1_1_0
java dom.TreeViewer data\personal.xml
```

Note: *Parse your own XML file instead of data\personal.xml*

5 Schema

Disclaimer

This package contains an implementation of the W3C XML Schema language. This implementation is experimental. The XML Schema language is still in working draft stage: you should not consider this implementation complete or correct. The limitations of this implementation are detailed below. Please read this document before using this package.

Introduction

This package contains an implementation of a subset of the W3C XML Schema Language as specified in the 7 April 2000 Working Drafts for Structures and Datatypes. The parsers contained in this package are able to read and validate XML documents with the grammar specified in either DTD or XML Schema format. There is no functionality for accessing typed data.

We are making this package available in order to get feedback on the features in the XML Schema language design and on representing an XML document's grammar as part of the document's DOM tree. We intend to update this package until it implements all of the functionality of the then current XML Schema Working Draft. If you are interested in a particular unimplemented feature, we welcome your feedback on the Xerces-J mailing list.

Limitations

The XML Schema implementation in this package is a subset of the features defined in the 7 April 2000 XML Schema Working Drafts.

Components Supported

- Element declarations
- Model group definitions: group
- Model groups: all, choice, sequence
- Attribute declarations
- Attribute group definitions
- Simple type definitions
- Complex type definitions

Components NOT Supported

- Wildcards
- Identity constraints: unique, key, keyref
- Notation declaration
- Annotation (ignored)

Features Supported

- Type derivation

- Anonymous types
- Nested element declaration
- Separate symbol spaces for elements, types, groups, and attribute groups
- Equivalency classes (partial)
- Target namespace
- "xsi:schemaLocation" and "xsl:noNamespacesSchemaLocation"

Features NOT Supported

- Constraints in Chapter 5
- Nullable
- Block
- Abstract
- "xsi:type"
- Include
- Import

Datatypes Supported

- Built-in simple types (primitive, derived)
- Simple type derivation (restriction, list)
- Regular expressions
- Binary (base64, hex)
- Constraining facets

Datatypes NOT Supported

- Date/time -- ISO8601 (work in progress)

Other Limitations

The schema is specified by the xsi:schemaLocation attribute on the root element of the document. The xsi prefix must be bound to the Schema document instance namespace, as specified by the working draft specification. See the sample provided in the Usage section.

Usage

In this release, schema validation has been integrated with the regular SAXParser and DOMParser classes. No special classes are required to parse documents that use a schema.

Documents that use XML Schema grammars specify the location of the grammar using an xsi:schemaLocation attribute attached to the root / top-level element in the document. Here is an example:

```
    <document xmlns:xsi='http://www.w3.org/1999/XMLSchema-instance'
   xsi:schemaLocation='document.xsd'>
    ...
    </document>
```

Review the sample file, 'data/personal.xsd' for an example of an XML Schema grammar.

6
Properties

Setting Features

To set a property on either org.apache.xerces.parsers.SAXParser or org.apache.xerces.parsers.DOMParser, you should use the SAX2 method `setProperty(String,Object)`. To query a property, use the SAX2 method `getProperty(String)`.

For example, to set the document factory by name:

```
DOMParser p=new DOMParser();
try {
  p.setProperty("http://apache.org/xml/properties/dom/document-class-name",
                "org.apache.xerces.dom.DocumentImpl");
} catch (SAXException e) {
  System.out.println("error in setting up parser property");
}
```

General Properties

http://xml.org/sax/properties/xml-string

Type:	java.lang.String
Access:	read-only
Desc:	Get the string of characters associated with the current event. If the parser recognizes and supports this property but is not currently parsing text, it should return null.
Note:	This property is currently not supported because the contents of the XML string returned by this property is not well defined.

DOM Parser Properties

http://apache.org/xml/properties/dom/current-element-node

Type:	org.w3c.dom.Node
Access:	read-only
Desc:	The current DOM element node while parsing.
Note:	This property is useful for determining the location with a DOM document when an error occurs.
See:	http://xml.org/sax/properties/dom-node

http://apache.org/xml/properties/dom/document-class-name

Type:	java.lang.String

Access:	read-write
Desc:	The fully qualified class name of the DOM implementation. The implementation used must have a zero argument constructor.
Default:	"org.apache.xerces.dom.DocumentImpl"
Note:	When the document class name is set to a value other than the name of the default document factory, the deferred node expansion feature does not work.
See:	http://apache.org/xml/features/dom/defer-node-expansion

SAX Parser Properties

http://xml.org/sax/properties/declaration-handler

Type:	org.xml.sax.ext.DeclHandler
Access:	read-write
Desc:	Set the handler for DTD declarations.

http://xml.org/sax/properties/lexical-handler

Type:	org.xml.sax.ext.LexicalHandler
Access:	read-write
Desc:	Set the handler for lexical parsing events.

http://xml.org/sax/properties/dom-node

Type:	org.w3c.dom.Node
Access:	(parsing) read-only; (not parsing) read/write
Desc:	The DOM node currently being visited, if SAX is being used as a DOM iterator. If the parser recognizes and supports this property but is not currently visiting a DOM node, it should return null.

7
Features

Setting Features

To set a feature on either org.apache.xerces.parsers.SAXParser or org.apache.xerces.parsers.DOMParser, you should use the SAX2 method `setFeature(String,boolean)`. To query a feature, use the SAX2 method `getFeature(String)`.

For example, to turn on validation:

```
SAXParser p=new SAXParser();
try {
  p.setFeature("http://xml.org/sax/features/validation", true);
} catch (SAXException e) {
  System.out.println("error in setting up parser feature");
}
```

General Features

http://xml.org/sax/features/validation

True:	Validate the document.
False:	Do not validate the document.
Default:	false
Access:	(parsing) read-only; (not parsing) read/write
Note:	If this feature is set to true, the document must specify a grammar. If this feature is set to false, the document may specify a grammar and that grammar will be parsed but no validation of the document contents will be performed.
See:	http://apache.org/xml/features/validation/dynamic http://xml.org/sax/features/namespaces

http://xml.org/sax/features/external-general-entities

True:	Include external general (text) entities.
False:	Do not include external general entities.
Default:	true
Access:	(parsing) read-only; (not parsing) read/write
See:	http://xml.org/sax/features/external-parameter-entities

http://xml.org/sax/features/external-parameter-entities

True:	Include external parameter entities and the external DTD subset.

False:	Do not include external parameter entities or the external DTD subset.
Default:	true
Access:	(parsing) read-only; (not parsing) read/write
See:	http://xml.org/sax/features/external-parameter-entities

http://xml.org/sax/features/namespaces

True:	Perform namespace processing: prefixes will be stripped off element and attribute names and replaced with the corresponding namespace URIs. By default, the two will simply be concatenated, but the namespace-sep core property allows the application to specify a delimiter string for separating the URI part and the local part.
False:	Do not perform namespace processing.
Default:	false
Access:	(parsing) read-only; (not parsing) read/write
Note:	If the validation feature is set to true, then the document must contain a grammar that supports the use of namespaces.
See:	http://xml.org/sax/features/validation http://xml.org/sax/properties/namespace-sep

http://apache.org/xml/features/validation/dynamic

True:	The parser will validate the document only if a grammar is specified.
False:	Validation is determined by the state of the http://xml.org/sax/features/validation feature.
Default:	false
See:	http://xml.org/sax/features/validation

http://apache.org/xml/features/validation/warn-on-duplicate-attdef

True:	Warn on duplicate attribute declaration.
False:	Do not warn on duplicate attribute declaration.
Default:	true

http://apache.org/xml/features/validation/warn-on-undeclared-elemdef

True:	Warn if element referenced in content model is not declared.
False:	Do not warn if element referenced in content model is not declared.
Default:	true

http://apache.org/xml/features/allow-java-encodings

True:	Allow Java encoding names in XMLDecl and TextDecl line.
False:	Do not allow Java encoding names in XMLDecl and TextDecl line.
Default:	false
Note:	

to be specified as a Java encoding name as well as the standard ISO encoding name. Be aware that other parsers may not be able to use Java encoding names. If this feature is set to false, an error will be generated if Java encoding names are used.

http://apache.org/xml/features/continue-after-fatal-error

True:	Continue after fatal error.
False:	Stops parse on first fatal error.
Default:	false

DOM Features

http://apache.org/xml/features/dom/defer-node-expansion

True:	Lazy DOM node expansion.
False:	Full DOM node expansion.
Default:	true
Note:	This feature only applies when the http://apache.org/xml/properties/dom/document-class-name property is set to a value other than the name of the default document factory. If this feature is set to true, the DOM nodes in the returned document are expanded as the tree is traversed. This feature allows the parser to return a document faster than if the tree is fully expanded during parsing and improves memory usage when the whole tree is not traversed.
See	http://apache.org/xml/properties/dom/document-class-name

http://apache.org/xml/features/dom/create-entity-ref-nodes

True:	Create EntityReference nodes in the DOM tree.
False:	Do not create EntityReference nodes in the DOM tree.
Default:	true
Note:	This feature only affects the appearance of EntityReference nodes in the DOM tree. The document will always contain the entity reference child nodes.

http://apache.org/xml/features/dom/include-ignorable-whitespace

True:	Includes text nodes that can be considered "ignorable whitespace" in the DOM tree.
False:	Does not include ignorable whitespace in the DOM tree.
Default:	true
Note:	The only way that the parser can determine if text is ignorable is by reading the associated grammar and having a content model for the document. When ignorable whitespace text nodes are included in the DOM tree, they will be flagged as ignorable. The ignorable flag can be queried by calling the TextImpl#isIgnorableWhitespace():boolean method.

http://apache.org/xml/features/domx/grammar-access

True:	Creates nodes that describe the grammar in the DOM tree.
False:	Does not create create grammar access nodes in the DOM tree. This setting makes the DOM behave as a standard DOM Level 1 implementation.
Default:	false
Note:	The grammar access nodes are appended as children of the DocumentType node. The grammar is specified as an XML Schema document tree, whether it was read from a document with an associated DTD or XML Schema grammar. This is currently a violation of the DOM Level 1 specification. ****** This is an experimental feature that is not guaranteed** ****** to be supported in future versions of the parser.**

SAX Features

http://xml.org/sax/features/namespace-prefixes

True:	Report the original prefixed names and attributes used for Namespace declarations.
False:	Do not report attributes used for Namespace declarations, and optionally do not report original prefixed names.
Default:	true
Access:	(parsing) read-only; (not parsing) read/write

http://xml.org/sax/features/string-interning

True:	All element names, prefixes, attribute names, Namespace URIs, and local names are internalized using java.lang.String.intern.
False:	Names are not necessarily internalized.
Default:	false
Access:	(parsing) read-only; (not parsing) read/write
Note:	Xerces-J does not support interning all strings using the String.intern() method because Xerces-J does its own intern optimizations for String objects.

8
Frequently Asked Questions

General FAQs

What are the new features?

Here are some of the new features in Xerces-J:

- Support for SAX2 alpha.
- Support for DOM Level 2 Core, Events, Traversal, and Ranges.
- Preliminary support for W3C XML Schema Language (validation only).
- Access to DTD information as a DOM Tree.
- Improved Conformance.
- Improved Performance.
- Parser option control based on SAX2.
- New "serialization" classes allow you output XML, HTML, and XHTML.
- New classes implement the HTML portion of the DOM Level 1 specification.

How do I turn on validation?

You can turn validation on and off via the SAX2 Configurable interface. This works for both the `SAXParser` and `DOMParser` classes.

The code snippet below shows how to turn validation on -- assume that parser is an instance of either `org.apache.xerces.parsers.SAXParser` or `org.apache.xerces.parsers.DOMParser`.

```
((Configurable)parser).setFeature("http://xml.org/sax/features/validation",
true);
```

What international encodings are supported by Xerces-J?

- UTF-8
- UTF-16 Big Endian, UTF-16 Little Endian
- IBM-1208
- ISO Latin-1 (ISO-8859-1)
- ISO Latin-2 (ISO-8859-2) [Bosnian, Croatian, Czech, Hungarian, Polish, Romanian, Serbian (in Latin transcription), Serbocroatian, Slovak, Slovenian, Upper and Lower Sorbian]
- ISO Latin-3 (ISO-8859-3) [Maltese, Esperanto]
- ISO Latin-4 (ISO-8859-4)
- ISO Latin Cyrillic (ISO-8859-5)
- ISO Latin Arabic (ISO-8859-6)
- ISO Latin Greek (ISO-8859-7)
- ISO Latin Hebrew (ISO-8859-8)
- ISO Latin-5 (ISO-8859-9) [Turkish]
- Extended Unix Code, packed for Japanese (euc-jp, eucjis)
- Japanese Shift JIS (shift-jis)
- Chinese (big5)
- Chinese for PRC (mixed 1/2 byte) (gb2312)

- Japanese ISO-2022-JP (iso-2022-jp)
- Cyrllic (koi8-r)
- Extended Unix Code, packed for Korean (euc-kr)
- Russian Unix, Cyrillic (koi8-r)
- Windows Thai (cp874)
- Latin 1 Windows (cp1252)
- cp858
- EBCDIC encodings:
  - EBCDIC US (ebcdic-cp-us)
  - EBCDIC Canada (ebcdic-cp-ca)
  - EBCDIC Netherland (ebcdic-cp-nl)
  - EBCDIC Denmark (ebcdic-cp-dk)
  - EBCDIC Norway (ebcdic-cp-no)
  - EBCDIC Finland (ebcdic-cp-fi)
  - EBCDIC Sweden (ebcdic-cp-se)
  - EBCDIC Italy (ebcdic-cp-it)
  - EBCDIC Spain, Latin America (ebcdic-cp-es)
  - EBCDIC Great Britain (ebcdic-cp-gb)
  - EBCDIC France (ebcdic-cp-fr)
  - EBCDIC Hebrew (ebcdic-cp-he)
  - EBCDIC Switzerland (ebcdic-cp-ch)
  - EBCDIC Roece (ebcdic-cp-roece)
  - EBCDIC Yugoslavia (ebcdic-cp-yu)
  - EBCDIC Iceland (ebcdic-cp-is)
  - EBCDIC Urdu (ebcdic-cp-ar2)
  - Latin 0 EBCDIC
  - EBCDIC Arabic (ebcdic-cp-ar1)

Building and Running FAQs

Which version of Swing is required?

This release uses Swing 1.1 (JFC 1.1). Swing is only used by the sample programs and is not required by the parser itself.

How do I recompile the source files?

To build Xerces-J on Windows, you need a copy of Cygnus's Cygwin. See http://sourceware.cygnus.com/cygwin. Once Cygwin is installed, you need to set two environment variables. Edit the batch file BuildAll.bat to set these variables. Execute BuildAll.bat, and then set your classpath to point to the src and samples directories in the Xerces-J distribution. You can then go to the top of the Xerces-J tree and type 'make'.

To build Xerces-J Java on UNIX, you need to set an environment variable. Edit the shell script BuildAll to set this variables. Execute BuildAll, and then set your classpath to point to the src and samples directories in the Xerces-J distribution. You can then go to the top of the Xerces-J tree and type 'make'.

How do I regenerate the api documentation?

To regenerate the api documentation, you need to set up your environment to build Xerces-J. Instead of typing 'make', you type 'make apidocs'.

How do I import Xerces-J into Visual Age for Java

- Why does VisualAge for Java 2.0 report problems when I import the Xerces-J parser?

  The current version of the Xerces-J parser uses Swing 1.1, while VisualAge for Java 2.0 comes with Swing 1.0.2. The free update for the Professional version of VisualAge for Java 2.0 installs Swing 1.0.3. The most important difference between Swing 1.0.2 - 1.0.3 and 1.1 is the Java package was changed from com.sun.java.swing.* to javax.swing.*.

  To fix the errors, you must download the Java Foundation Classes 1.1 with Swing 1.1 from Sun's Java

home page and import the "swingall.jar" file into VisualAge for Java 2.0. The Swing 1.1 package can be found at the following URL:

http://java.sun.com/products/jfc/index.html

Refer to the VisualAge for Java 2.0 documentation for information about how to import a JAR file into the repository and add that code to your workspace.

- Are there any other tips for importing the Xerces-J parser into VisualAge for Java 2.0?

  The most useful tip applies to **any** updated code that you import into the VisualAge for Java 2.0 product. Before updating code, do the following:

  1. version the old code
  2. delete it from your workspace
  3. import the new code

  Deleting code from your workspace does not actually delete the code permanently -- the versioned code is moved to the repository where it can be retrieved later. Be aware, though, that removing code from your workspace will cause problems with all of the other classes that use that code. VisualAge for Java 2.0 will flag them as errors but this situation is temporary. When you import the new code, the errors found when deleting the old code will be fixed.

  If you are unsure as to how to perform any of these steps, refer to the VisualAge for Java 2.0 documentation.

Is this Xerces-J version 100% pure Java compliant?

Running JavaPureCheck on the Xerces-J parser code indicated 339 pure Java classes, 9 warnings, and 0 errors. The nine warnings are enumerated below with explanations. To see the entire report, click here.

There are many common cases where JavaPureCheck issues warnings even though the code is pure Java. These are the most common reasons:

1. Warning: method reference: `java.lang.Class.forName(java.lang.String)`

   This warning is issued in the following two cases:

   1. Program code calls the `Class.forName(String)` method to dynamically load a class file. In this situation, the specified class may contain impure Java code. In the cases where this method is called directly in the parser code, an explanation is provided detailing why this warning can be ignored.
   2. Program code makes direct reference to an object's class. For example: `Class stringClass = String.class;`. In this situation the Java compiler converts `String.class` to the method call `Class.forName("java.lang.String")`. As long as the object whose class is being referenced is pure Java, the code making the reference remains pure.

2. Warning: possible hard-coded path: ...

   When a String literal contains a common path separator character (e.g. '/' or '\'), JavaPureCheck assumes that it is a hard-coded path and that the class may not contain portable code. While a human tester can verify that the string is not a path, JavaPureCheck must be conservative and issue a warning.

Explanations:

- Class: org.xml.sax.helpers.ParserFactory
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: The ParserFactory class is part of the standard SAX 1.0 distribution. The warning given that this class "may load impure class" is correct -- the ParserFactory class may load impure Java class files. However, the purpose of this utility class is to load parser classes by name and therefore cannot dynamically check the pureness of parser classes loaded in this fashion. Since the Xerces-J parser does not use this method directly, it is not a problem that concerns the Xerces-J parser.
- Class: org.apache.xml.serialize.SerializerFactory
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: The SerializerFactory supports querying a system property to dynamically instantiate a serializer by class name. However, all of the serializer classes provided in this distribution are pure.

The only way to load an impure serializer is if the user provided an impure serializer implementation.

- Class: org.apache.xml.serialize.OutputFormat
  - Warning: possible hard-coded path: text/xml
  - Warning: possible hard-coded path: text/plain
  - Warning: possible hard-coded path: application/pdf
  - Warning: possible hard-coded path: text/html
  - Warning: possible hard-coded path: -//W3C//DTD XHTML 1.0 Strict//EN
  - Explanation: These strings are not filenames.
- Class: org.apache.xml.serialize.HTMLdtd
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: Referencing class object that is pure Java.
- Class: org.apache.html.dom.HTMLDocumentImpl
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: Referencing class object that is pure Java.
- Class: org.apache.xerces.readers.StringReader
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: Referencing class object that is pure Java.
- Class: org.apache.xerces.parsers.DOMParser
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: The DOMParser class allows the user to set the DOM implemenation to use, by name. However, the default DOM implementation is pure Java.
  - Explanation: Referencing class object that is pure Java.
- Class: org.apache.xerces.utils.CharDataChunk
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: Referencing class object that is pure Java.
- Class: org.apache.xerces.utils.UTF8DataChunk
  - Warning: method reference: java.lang.Class.forName(java.lang.String)
  - Explanation: Referencing class object that is pure Java.

The results file of the JavaPureCheck can be viewed by clicking here.

> ***Note:*** *The samples were not checked with JavaPureCheck and are not guaranteed to be pure Java. We reserve the right to write samples in the future that are platform specific and therefore may not pass as pure Java. The parser, however, will remain pure Java.*

How do I get Xerces-J to run on the Mac under MRJ?

Prerequisites (available from http://developer.apple.com/java/):

- MRJ 2.1 (this is the most recent version of the JVM)
- MRJ SDK 2.1 (this is the most recent version of the Java developer tools)

Instructions (other variations would work also):

1. Download the .tar.gz file containing Xerces-J.
2. Use Stuffit Expander(tm), Suntar, or some other Macintosh tool that supports the .tar.gz format to expand the downloaded file.
3. JBindery, part of MRJ SDK 2.1, is used to create a double-clickable Java application with the necessary configuration information built in. It is analogous to writing a .bat or .sh script.
4. **To run the dom.DOMWriter example:**
   1. Double click on JBindery to start it up.
   2. Click on the Classpath panel.
   3. Click on the "Add .zip File" button and add both the "Xerces-J.jar" and "Xerces-JSamples.jar" files.
   4. Click on the Command panel.
   5. Enter "dom.DOMWriter" as the Class name. Enter "data/personal.xml" in the Optional parameters box.
   6. Click on Save Settings button, pick a name such as "Run `dom.DOMWriter`" for the file, and **be sure** that "Save as Application" is selected (this is the default) and save the file.

7. Quit JBindery.
8. You can now double click on the file you created in step f to run the XJParse example.

Why do I get ArrayIndexOutOfBoundsException in the Symantec Visual Cafe debugger?

The Visual Cafe debugger is set to trap `ArrayIndexOutOfBoundsException` exceptions by default. Xerces-J uses `ArrayIndexOutOfBoundsException` internally to signal exceptional, but not erroneous conditions. In order to run Xerces-J2 inside Visual Cafe's debugger, you need to turn off the trapping of these exceptions.

To do this:

1. Select the "Options" item in the "Project" menu.
2. Select the "Debugger" tab in the dialog which appears.
3. Select "Exceptions" from the popup menu.
4. Remove the check from the checkbox for `java.lang.ArrayIndexOutOfBoundsException`.

Writing Application FAQs

How do I create a DOM parser?

```
import org.apache.xerces.parsers.DOMParser;
import org.w3c.dom.Document;
import org.xml.sax.SAXException;
import java.io.IOException;

  ...

String xmlFile = "file:///xerces-1_1_0/data/personal.xml";

DOMParser parser = new DOMParser();

try {
    parser.parse(xmlFile);

} catch (SAXException se) {
    se.printStackTrace();
} catch (IOException ioe) {
    ioe.printStackTrace();
}

Document document = parser.getDocument();
```

How do I create a SAX parser?

```
import org.apache.xerces.parsers.SAXParser;
import org.xml.sax.Parser;
import org.xml.sax.ParserFactory;
import org.xml.sax.SAXException;
import java.io.IOException;

  ...

String xmlFile = "file:///xerces-1_1_0/data/personal.xml";

String parserClass = "org.apache.xerces.parsers.SAXParser";
Parser parser = ParserFactory.makeParser(parserClass);

try {
    parser.parse(xmlFile);
} catch (SAXException se) {
```

```
        se.printStackTrace();
    } catch (IOException ioe) {
            ioe.printStackTrace();
    }
```

How do I control the various parser options?

For this release, all of the parser control API's have been switched over to the SAX2 Configurable interface. This provide a uniform and extensible mechanism for setting and querying parser options. Here are guides to the set of available features and properties.

How do I use the lazy evaluating DOM implementation?

The DOM parser class `org.apache.xerces.parsers.DOMParser` uses a DOM implementation that can take advantage of lazy evaluation to improve performance. There is also a mode where the parser creates all of the nodes as the document is parsed. By default, the parser uses the lazy evaluation DOM implementation.

Nodes in the DOM tree are only created when they are accessed. The initial call to `getDocument()` will return a DOM tree that consists only of the Document node. All of the immediate children of a Node are created when any of that Node's children are accessed. This shortens the time it takes to parse an XML file and create a DOM tree at the expense of requiring more memory during parsing and traversing the document.

The lazy evaluation feature is set using the SAX2 Configurable interface. To turn off this feature, use the following code:

```
    DOMParser parser = new DOMParser();
    parser.setFeature("http://apache.org/xml/features/dom/defer-node-expansion",
   false);
```

To turn the lazy evaluation feature back on, use the following code:

```
    parser.setFeature("http://apache.org/xml/features/dom/defer-node-expansion",
   true);
```

How do handle errors?

When you create a parser instance, the default error handler does nothing. This means that your program will fail silently when it encounters an error. You should register an error handler with the parser by supplying a class which implements the `org.xml.sax.ErrorHandler` interface. This is true regardless of whether your parser is a DOM based or SAX based parser.

How can I control the way that entities are represented in the DOM?

The feature `http://apache.org/xml/features/dom/create-entity-ref-nodes` controls how entities appear in the DOM tree. When this feature is set to true (the default), an occurance of an entity reference in the XML document will be represented by a subtree with an EntityReference node at the root whose children represent the entity expansion.

If the property is false, an entity reference in the XML document is represented by only the nodes that represent the entity expansion.

In either case, the entity expansion will be a DOM tree representing the structure of the entity expansion, not a text node containing the entity expansion as text.

Why does "non-validating" not mean "well-formedness checking only"?

Using a "non-validating" parser does not mean that only well-formedness checking is done! There are still many things that the XML specification requires of the parser, including entity substitution, defaulting of attribute values, and attribute normalization.

This table describes what "non-validating" really means for Xerces-J parsers. In this table, "no DTD" means no internal or external DTD subset is present.

	non-validating parsers		validating parsers	

	DTD present	no DTD	DTD present	no DTD
DTD is read	Yes	No	Yes	Error
entity substitution	Yes	No	Yest	Error
defaulting of attributes	Yes	No	Yes	Error
attribute normalization	Yes	No	Yes	Error
checking against model	No	No	Yes	Error

How do I associate my own data with a node in the DOM tree?

The class `org.apache.xerces.dom.NodeImpl` provides a `void setUserData(Object o)` and an `Object getUserData()` method that you can use to attach any object to a node in the DOM tree.

Beware that you should try and remove references to your data on nodes you no longer use (by calling `setUserData(null)`, or these nodes will not be garbage collected until the whole document is.

How do I more efficiently parse several documents sharing a common DTD?

DTDs are not currently cached by the parser. The common DTD, since it is specified in each XML document, will be re-parsed once for each document.

However, there are things that you can do now, to make the process of reading DTD's more efficient:

- keep your DTD and DTD references local
- use internal DTD subsets, if possible
- load files from server to local client before parsing
- Cache document files into a local client cache. You should do an HTTP header request to check whether the document has changed, before accessing it over the network.
- Do not reference an external DTD or internal DTD subset at all. In this case, no DTD will be read.
- Use a custom `EntityResolver` and keep common DTDs in a memory buffer.

How do I access the DOM Level 2 functionality?

The DOM Level 2 specification is at the stage of "Candidate Recommendation" (CR), which allows feedback from implementors before it becomes a "Recommedation". It is comprised of "core" functionality, which is mainly the DOM Namespaces implementation, and a number of optional modules (called Chapters in the spec).

Please refer to:

http://www.w3.org/TR/DOM-Level-2/ for the latest DOM Level 2 specification.

The following DOM Level 2 modules are fully implemented in Xerces-J:

- Chapter 1: Core - most of these enhancements are for Namespaces, and can be acessed through additional functions which have been added directly to the org.w3c.dom.* classes.
- Chapter 6: Events - The org.w3c.dom.events.EventTarget interface is implemented by all `Nodes` of the DOM. The Xerces-J DOM implementation handles all of the event triggering, capture and flow.
- Chapter 7: Traversal - The Traversal module interfaces are located in org.w3c.dom.traversal. The `NodeIterator` and `TreeWalker`, and `NodeFilter` interfaces have been supplied to allow traversal of the DOM at a higher-level. Our DOM Document implementation class, `DocumentImpl` class now implements `DocumentTraversal`, which supplies the factory methods to create the iterators and treewalkers.
- Chapter 8. Range - The Range module interfaces are located in org.w3c.dom.range. The Range interface allows you to specify ranges or selections using boundary points in the DOM, along with functions (like delete, clone, extract..) that can be performed on these ranges. Our DOM Document implementation class, `DocumentImpl` class now implements `DocumentRange`, that supplies the factory method to create a `Range`.

  Note: *Since the DOM Level 2 is still in the CR phase, some changes to these specs are still possible. The purpose of this phase is to provide feedback to the W3C, so that the specs can be clarified and implementation concerns can be addressed.*

How do I read data from a stream as it arrives?

For performance reasons, all the standard Xerces processing uses readers which buffer the input. In order to read data from a stream as it arrives, you need to instruct Xerces to use the `StreamingCharReader` class as its reader. To do this, create a subclass of `org.apache.xerces.readers.DefaultReaderFactory` and override `createCharReader` and `createUTF8Reader` as shown below.

```
 public class StreamingCharFactory extends
org.apache.xerces.readers.DefaultReaderFactory {
   public XMLEntityHandler.EntityReader createCharReader(XMLEntityHandler
entityHandler,
                                                         XMLErrorReporter
errorReporter,
                                                         boolean
sendCharDataAsCharArray,
                                                         Reader reader,
                                                         StringPool stringPool)
throws Exception
     {
       return new org.apache.xerces.readers.StreamingCharReader(entityHandler,
                  errorReporter, sendCharDataAsCharArray, reader, stringPool);
     }

   public XMLEntityHandler.EntityReader createUTF8Reader(XMLEntityHandler
entityHandler,
                                                         XMLErrorReporter
errorReporter,
                                                         boolean
sendCharDataAsCharArray,
                                                         InputStream data,
                                                         StringPool stringPool)
throws Exception
     {
       XMLEntityHandler.EntityReader reader;
       reader = new org.apache.xerces.readers.StreamingCharReader(entityHandler,
                    errorReporter, sendCharDataAsCharArray,
                    new InputStreamReader(data, "UTF8"), stringPool);
       return reader;
     }

 }
```

In your program, after you instantiate a parser class, replace the `DefaultReaderFactory` with `StreamingCharFactory`, and be sure to wrap the `InputStream` that you are reading from with an `InputStreamReader`.

```
 InputStream in = ... ;
 SAXParser p = new SAXParser();
 DocumentHandler h = ... ;
 // set the correct reader factory
 p.setReaderFactory(((StreamingSAXClient)h).new StreamingCharFactory());
 p.setDocumentHandler(h);

 // be sure to wrap the input stream in an InputStreamReader.
 p.parse(new InputSource(new InputStreamReader(in)));
```

Performance FAQs

General Performance

Don't use XML where it doesn't make sense. XML is not a panacea. You will not get good performance by transferring and parsing a lot of XML files.

Using XML is memory, CPU, and network intensive.

Parser Performance

Avoid creating a new parser each time you parse; reuse parser instances. A pool of reusable parser instances might be a good idea if you have multiple threads parsing at the same time.

Parsing Documents Performance

- Convert the document to US ASCII ("US-ASCII") or Unicode ("UTF-8" or "UTF-16") before parsing. Documents written using ASCII are the fastest to parse because each character is guaranteed to be a single byte and map directly to their equivalent Unicode value. For documents that contain Unicode characters beyond the ASCII range, multiple byte sequences must be read and converted for each character. There is a performance penalty for this conversion. The UTF-16 encoding alleviates some of this penalty because each character is specified using two bytes, assuming no surrogate characters. However, using UTF-16 can roughly double the size of the original document which takes longer to parse.
- Explicitly specify "US-ASCII" encoding if your document is in ASCII format. If no encoding is specified, the XML specification requires the parser to assume UTF-8 which is slower to process.
- Avoid external entities and external DTDs. The extra file opens and transcoding setup is expensive.
- Reduce character count; smaller documents are parsed quicker. Replace elements with attributes where it makes sense. Avoid gratuitous use of whitespace because the parser must scan past it.
- Avoid using too many default attributes. Defaulting attribute values slows down processing.

XML Application Performance

- Turn validation off if you don't need it. Validation is expensive. Also, avoid using a DOCTYPE line in your XML document. The current version of the parser will always read the DTD if the DOCTYPE line is specified even when not validating.
- For large documents, avoid using DOM which uses a lot of memory. Instead, use SAX if appropriate. The DOM parser requires that the entire document be read into memory before the application processes the document. The SAX parser uses very little memory and notifies the application as parts of the document are parsed.

Migrating to Xerces Java Parser

What should I be aware of when using various DOM parsers?

There are a couple of points to note when using the various DOM parsers. This FAQ discusses some of the differences between the Xerces, Oracle and Sun XML parsers:

1. **Parsing methods:**

   The Xerces-J and Oracle parsers have a parser object that parses XML files and constructs a DOM tree which is queried after parsing.

   The Sun parser calls a static method on the `XmlDocument` class to parse and construct a DOM tree.
2. **Specifying the source file:**

   All three parsers allow specifying the source of the XML document using the SAX `InputSource` object as well as parsing from `java.io.InputStream` and `java.io.Reader` object.
3. **Error handling:**

   The Xerces-J parser uses the SAX `ErrorHandler` mechanism on all parser types, including DOM.

   The Oracle parser only allows you to specify which output stream or writer you want the error to be written.

   The Sun parser has no way to request error notification when parsing XML files into DOM trees. An exceptions will be thrown if an error occurs during parsing.
4. **Validation:**

The Xerces-J and Oracle DOM parsers use a method to set validation.

Because of the way that DOM documents are constructed from XML files in the Sun parser, validation is set via a parameter to the static `createXmlDocument` method.

5. **Standard versus proprietary features:**

   If the user has written their programs using the W3C DOM API, then migrating to Xerces-J is easy. If however, the user takes advantage of non-standard, proprietary features of the Oracle and Sun parsers and DOM implementations, migration will be harder. This document does not go into any detail regarding migration of features specific to each parser's implementation that are non-standard.

Samples:

Xerces 1.0.x:

```
// instantiate parser
org.apache.xerces.parsers.DOMParser parser;
parser = new org.apache.xerces.parsers.DOMParser();

// specifying validation
boolean validate = /* true or false */;
parser.setFeature("http://xml.org/sax/features/validation", validate);

// installing error handler
org.xml.sax.ErrorHandler handler = /* SAX error handler */;
parser.setErrorHandler(handler);

// parsing document from URI string
String uri = /* uri */;
parser.parse(uri);

// parsing document from input stream
java.io.InputStream stream = /* input stream */;
parser.parse(new org.xml.sax.InputSource(stream));

// parsing document from reader
java.io.Reader reader = /* reader */;
parser.parse(new org.xml.sax.InputSource(reader));

// querying document after parse
org.w3c.dom.Document document;
document = parser.getDocument();
```

Oracle 2.0.2.x:

```
// instantiate parser
oracle.xml.parser.v2.DOMParser parser;
parser = oracle.xml.parser.v2.DOMParser();

// specifying validation
boolean validate = /* true or false */;
parser.setValidationMode(validate);

// installing error stream to output stream (with and without encoding)
java.io.OutputStream output = /* output stream */;
String encoding = /* Java encoding name */;
parser.setErrorStream(output);
parser.setErrorStream(output, encoding);

// installing error stream to print writer
java.io.PrintWriter printer = /* print writer */;
parser.setErrorStream(printer);
```

```
// parsing document from URI string
String uri = /* uri */;
parser.parse(uri);

// parsing document from input stream
java.io.InputStream stream = /* input stream */;
parser.parse(stream);

// parsing document from reader
java.io.Reader reader = /* reader */;
parser.parse(reader);

// querying document after parse
org.w3c.dom.Document document;
document = parser.getDocument();
```

Sun TR2:

```
// parsing document from URI string
String uri = /* uri */;
Document document;
document = com.sun.xml.tree.XmlDocument.createXmlDocument(uri);

// parsing document from URI string (with validation)
boolean validate = /* true or false */;
document = com.sun.xml.tree.XmlDocument.createXmlDocument(uri, validate);

// parsing document from input stream
java.io.InputStream stream = /* input stream */;
document = com.sun.xml.tree.XmlDocument.createXmlDocument(stream, validate);

// parsing document from reader
java.io.Reader reader = /* reader */;
document = com.sun.xml.tree.XmlDocument.createXmlDocument
           (new org.xml.sax.InputSource(reader), validate);
```

What should I be aware of when using various SAX parsers?

There are a couple of points to note when using the various SAX parsers:

The SAX API has detailed specifications on how documents are parsed and entities are resolved, so little migration effort required. The only change is the construction of the SAX parser. See the following examples for construction details.

> ***Note:*** *Regarding validation:*
>
> *If a parser is SAX2 compliant, there is a standard way of turning on validation. The Xerces parser implements the appropriate methods today, even though they haven't been finalized, yet. The parsers downloaded from Oracle and Sun do not yet implement these methods. The Oracle parser has a method to turn validation on and off. The Sun parser requires you to instantiate a separate parser object to perform validation.*

Samples:

Xerces 1.0.x:

```
// instantiate parser
org.apache.xerces.parsers.SAXParser parser;
parser = new org.apache.xerces.parsers.SAXParser();

// specifying validation
boolean validate = /* true or false */;
```

```
    parser.setFeature("http://xml.org/sax/features/validation", validate);

    // installing error handler
    org.xml.sax.ErrorHandler errorHandler = /* SAX error handler */;
    parser.setErrorHandler(errorHandler);

    // installing document handler
    org.xml.sax.DocumentHandler documentHandler = /* SAX document handler */;
    parser.setDocumentHandler(documentHandler);

    // parsing document from URI string
    String uri = /* uri */;
    parser.parse(uri);

    // parsing document from input stream
    java.io.InputStream stream = /* input stream */;
    parser.parse(new org.xml.sax.InputSource(stream));

    // parsing document from reader
    java.io.Reader reader = /* reader */;
    parser.parse(new org.xml.sax.InputSource(reader));
```

Oracle 2.0.2.x:

```
    // instantiate parser
    oracle.xml.parser.v2.SAXParser parser;
    parser = oracle.xml.parser.v2.SAXParser();

    // specifying validation
    boolean validate = /* true or false */;
    parser.setValidationMode(validate);

    // ... the rest is the same as Xerces ...
```

Sun TR2:

```
    // instantiate parser
    boolean validate = /* true or false */;
    com.sun.xml.parser.Parser parser;
    if (validate) {
        parser = new com.sun.xml.parser.ValidatingParser();
    }
    else {
        parser = new com.sun.xml.parser.Parser();
    }

    // ... the rest is the same as Xerces ...
```

How do I migrate my code from XML4J Version 2.0.x?

Migrating from the version 2.0.x native SAX and DOM parser classes should be straight forward.

change this XML4J class	to this Xerces-J class
com.ibm.xml.parsers.SAXParser	org.apache.xerces.parsers.SAXParser
com.ibm.xml.parsers.ValidatingSAXParser	org.apache.xerces.parsers.SAXParser + switch
com.ibm.xml.parsers.NonValidatingDOMParser	org.apache.xerces.parsers.DOMParser
com.ibm.xml.parsers.DOMParser	org.apache.xerces.parsers.DOMParser + switch

Table entries that say " + switch" mean that you should use the Configurable API to turn validation on. See the answer in Validation.

Common Problems

I tried to use Xerces-J to parse an HTML file and it generated an error. What did I do wrong?

Unfortunately, HTML does not, in general, follow the XML grammar rules. Most HTML files do not meet the XML style quidelines. Therefore, the XML parser generates XML well-formedness errors.

Typical errors include:

- Missing end tags, e.g. <P> with no </P> (end tags are not required in HTML)
- Missing closing slash on <IMG HREF="foo" /> (not required in HTML)
- Missing quotes on attribute values, e.g. <IMG width="600"> (not generally required in HTML)

HTML must match the XHTML standard for well-formedness before it can be parsed by Xerces-J or any other XML parser. You can find the XHTML standard on the W3C web site.

I get an "invalid UTF-8 character" error.

There are many Unicode characters that are not allowed in an XML document, according to the XML spec. Typical disallowed characters are control characters, even if you escape them using the Character Reference form: &#xxxx; . See the XML spec, sections 2.2 and 4.1 for details. If the parser is generating this error, it is very likely that there is a character in the file that you can not see. You can generally use a UNIX command like "od -hc" to find it.

I get an error when I access EBCDIC XML files, what is happening?

If an XML document/file is not UTF-8, then you MUST specify the encoding. When transcoding a UTF8 document to EBCDIC, remember to change this:

- <?xml version="1.0" encoding="UTF-8"?>
  to something like this:
  <?xml version="1.0" encoding="ebcdic-cp-us"?>

I get an error on the EOF character (0x1A) -- what is happening?

You are probably using the **LPEX** editor, which automatically inserts an End-of-file character (0x1A) at the end of your XML document (other editors might do this as well). Unfortunately, the EOF character (0x1A) is an illegal character according to the XML specification, and Xerces-J correctly generates an error.

9
JavaPureCheck Output

JavaPureCheck Results

```
    ######################## JavaPureCheck Report ############################
    #
    #      Generated on          : January 26, 2000 10:34:32 AM PST
    #      System Model Version  : jdk11
    #      JavaPureCheck Version : 3.15
    #      Rule Base Version     : 1.92
    #
    #      Summary:
    #
    #      PURE: 339         WARNING: 9         ERROR: 0
    #
    #      Final Result   :  WARNING
    #
    ##########################################################################

    Class: org.xml.sax.InputSource
    Status: PURE

    Class: org.xml.sax.misc.LexicalHandler
    Status: PURE

    Class: org.xml.sax.misc.NamespaceHandler
    Status: PURE

    Class: org.xml.sax.misc.DeclHandler
    Status: PURE

    Class: org.xml.sax.SAXNotSupportedException
    Status: PURE

    Class: org.xml.sax.AttributeList
    Status: PURE

    Class: org.xml.sax.SAXNotRecognizedException
    Status: PURE

    Class: org.xml.sax.DTDHandler
    Status: PURE

    Class: org.xml.sax.ErrorHandler
    Status: PURE
```

```
Class: org.xml.sax.DocumentHandler
Status: PURE

Class: org.xml.sax.SAXException
Status: PURE

Class: org.xml.sax.helpers.LocatorImpl
Status: PURE

Class: org.xml.sax.helpers.ConfigurableParserAdapter
Status: PURE

Class: org.xml.sax.helpers.ParserFactory
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.xml.sax.helpers.AttributeListImpl
Status: PURE

Class: org.xml.sax.SAXParseException
Status: PURE

Class: org.xml.sax.Parser
Status: PURE

Class: org.xml.sax.EntityResolver
Status: PURE

Class: org.xml.sax.HandlerBase
Status: PURE

Class: org.xml.sax.Locator
Status: PURE

Class: org.xml.sax.Configurable
Status: PURE

Class: org.apache.xml.serialize.HTMLSerializer
Status: PURE

Class: org.apache.xml.serialize.BaseMarkupSerializer
Status: PURE

Class: org.apache.xml.serialize.SerializerFactory
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xml.serialize.Serializer
Status: PURE

Class: org.apache.xml.serialize.OutputFormat$DTD
Status: PURE
```

```
Class: org.apache.xml.serialize.OutputFormat$Defaults
Status: PURE

Class: org.apache.xml.serialize.OutputFormat
        Warning: possible hard-coded path: text/xml
        Note: Defines a bad path
        Explanation:         <Explanation required>
        Warning: possible hard-coded path: text/plain
        Note: Defines a bad path
        Explanation:         <Explanation required>
        Warning: possible hard-coded path: application/pdf
        Note: Defines a bad path
        Explanation:         <Explanation required>
        Warning: possible hard-coded path: text/html
        Note: Defines a bad path
        Explanation:         <Explanation required>
        Warning: possible hard-coded path: -//W3C//DTD XHTML 1.0 Strict//EN
        Note: Defines a bad path
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xml.serialize.Method
Status: PURE

Class: org.apache.xml.serialize.SerializerFactoryImpl
Status: PURE

Class: org.apache.xml.serialize.HTMLdtd
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xml.serialize.XMLSerializer
Status: PURE

Class: org.apache.xml.serialize.XHTMLSerializer
Status: PURE

Class: org.apache.xml.serialize.ElementState
Status: PURE

Class: org.apache.xml.serialize.TextSerializer
Status: PURE

Class: org.apache.xml.serialize.LineSeparator
Status: PURE

Class: org.apache.xml.serialize.DOMSerializer
Status: PURE

Class: org.apache.html.dom.HTMLDirectoryElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTableRowElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTableColElementImpl
```

```
Status: PURE

Class: org.apache.html.dom.HTMLFieldSetElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLModElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTableSectionElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLDocumentImpl
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.html.dom.NameNodeListImpl
Status: PURE

Class: org.apache.html.dom.HTMLStyleElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLBuilder
Status: PURE

Class: org.apache.html.dom.HTMLFontElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTableElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLMetaElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTextAreaElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLPreElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLUListElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLOptionElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLHRElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLLabelElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLTableCaptionElementImpl
Status: PURE
```

```
Class: org.apache.html.dom.HTMLBaseFontElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLMenuElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLQuoteElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLObjectElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLSelectElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLFormElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLScriptElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLBRElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLOListElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLImageElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLBaseElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLFrameElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLHtmlElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLParagraphElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLDivElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLLIElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLHeadElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLOptGroupElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLButtonElementImpl
Status: PURE
```

```
Class: org.apache.html.dom.HTMLTableCellElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLInputElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLBodyElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLAreaElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLHeadingElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLLegendElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLCollectionImpl
Status: PURE

Class: org.apache.html.dom.HTMLFormControl
Status: PURE

Class: org.apache.html.dom.HTMLParamElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLDOMImplementationImpl
Status: PURE

Class: org.apache.html.dom.HTMLTitleElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLMapElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLDListElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLAppletElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLIFrameElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLFrameSetElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLAnchorElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLIsIndexElementImpl
Status: PURE

Class: org.apache.html.dom.HTMLLinkElementImpl
Status: PURE

Class: org.apache.html.dom.CollectionIndex
```

```
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$XMLDeclDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner
Status: PURE

Class: org.apache.xerces.framework.Version
Status: PURE

Class: org.apache.xerces.framework.XMLContentSpecNode
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$TrailingMiscDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLValidator$ContentSpec
Status: PURE

Class: org.apache.xerces.framework.XMLDTDScanner$EventHandler
Status: PURE

Class: org.apache.xerces.framework.XMLAttrList
Status: PURE

Class: org.apache.xerces.framework.XMLParser$NullReader
Status: PURE

Class: org.apache.xerces.framework.XMLParser$ReaderState
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$EventHandler
Status: PURE

Class: org.apache.xerces.framework.XMLValidator
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$PrologDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLErrorReporter
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$ScannerDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$EndOfInputDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLParser
Status: PURE

Class: org.apache.xerces.framework.XMLDocumentScanner$ContentDispatcher
Status: PURE

Class: org.apache.xerces.framework.XMLDTDScanner
Status: PURE
```

```
Class: org.apache.xerces.domx.DOMException
Status: PURE

Class: org.apache.xerces.domx.XGrammarWriter
Status: PURE

Class: org.apache.xerces.domx.XGrammarWriter$OutputFormat
Status: PURE

Class: org.apache.xerces.readers.XMLEntityHandler$EntityReader
Status: PURE

Class: org.apache.xerces.readers.StreamingCharReader$DeferredError
Status: PURE

Class: org.apache.xerces.readers.StringReader
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xerces.readers.UTF8CharReader
Status: PURE

Class: org.apache.xerces.readers.CharReader
Status: PURE

Class: org.apache.xerces.readers.XMLDeclRecognizer
Status: PURE

Class: org.apache.xerces.readers.XCatalog
Status: PURE

Class: org.apache.xerces.readers.UCSReader
Status: PURE

Class: org.apache.xerces.readers.DefaultReaderFactory
Status: PURE

Class: org.apache.xerces.readers.XMLEntityHandler$CharBuffer
Status: PURE

Class: org.apache.xerces.readers.UTF8Recognizer
Status: PURE

Class: org.apache.xerces.readers.StreamingCharReader
Status: PURE

Class: org.apache.xerces.readers.UCSRecognizer
Status: PURE

Class: org.apache.xerces.readers.XCatalog$Parser
Status: PURE

Class: org.apache.xerces.readers.UTF8Reader
Status: PURE
```

```
Class: org.apache.xerces.readers.XMLEntityReaderFactory
Status: PURE

Class: org.apache.xerces.readers.XMLEntityReader
Status: PURE

Class: org.apache.xerces.readers.XCatalog$Parser$Resolver
Status: PURE

Class: org.apache.xerces.readers.AbstractCharReader$DeferredError
Status: PURE

Class: org.apache.xerces.readers.AbstractCharReader
Status: PURE

Class: org.apache.xerces.readers.MIME2Java
Status: PURE

Class: org.apache.xerces.readers.UTF8Recognizer$XMLDeclReader
Status: PURE

Class: org.apache.xerces.readers.EBCDICRecognizer
Status: PURE

Class: org.apache.xerces.readers.XMLEntityHandler
Status: PURE

Class: org.apache.xerces.readers.XMLCatalogHandler
Status: PURE

Class: org.apache.xerces.parsers.RevalidatingDOMParser
Status: PURE

Class: org.apache.xerces.parsers.SAXParser
Status: PURE

Class: org.apache.xerces.parsers.DOMParser
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xerces.validators.datatype.InvalidDatatypeValueException
Status: PURE

Class: org.apache.xerces.validators.datatype.BooleanValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.DecimalValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.IllegalFacetException
Status: PURE

Class: org.apache.xerces.validators.datatype.DatatypeMessageProvider
Status: PURE

Class: org.apache.xerces.validators.datatype.InternalDatatypeValidator
```

```
Status: PURE

Class: org.apache.xerces.validators.datatype.UnknownFacetException
Status: PURE

Class: org.apache.xerces.validators.datatype.DatatypeValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.IntegerValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.TimeDurationValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.DoubleValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.IllegalFacetValueException
Status: PURE

Class: org.apache.xerces.validators.datatype.StringValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.FloatValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.TimeInstantValidator
Status: PURE

Class: org.apache.xerces.validators.datatype.RealValidator
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator
Status: PURE

Class: org.apache.xerces.validators.dtd.CMStateSet
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$EventHandler
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttributeValidator
Status: PURE

Class: org.apache.xerces.validators.dtd.CMNode
Status: PURE

Class: org.apache.xerces.validators.dtd.SimpleContentModel
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorENTITY
Status: PURE

Class: org.apache.xerces.validators.dtd.CMLeaf
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorID
Status: PURE
```

```
Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorENTITIES
Status: PURE

Class: org.apache.xerces.validators.dtd.CMBinOp
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorNOTATION
Status: PURE

Class: org.apache.xerces.validators.dtd.DFAContentModel
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorIDREF
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorIDREFS
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorNMTOKENS
Status: PURE

Class: org.apache.xerces.validators.dtd.ElementDeclPool
Status: PURE

Class: org.apache.xerces.validators.dtd.CMException
Status: PURE

Class: org.apache.xerces.validators.dtd.CMUniOp
Status: PURE

Class: org.apache.xerces.validators.dtd.EntityPool$RequiredNotation
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorCDATA
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$AttValidatorNMTOKEN
Status: PURE

Class: org.apache.xerces.validators.dtd.XMLContentModel
Status: PURE

Class: org.apache.xerces.validators.dtd.InsertableElementsInfo
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator
                            $AttValidatorENUMERATION
Status: PURE

Class: org.apache.xerces.validators.dtd.DTDValidator$ContentSpecImpl
Status: PURE

Class: org.apache.xerces.validators.dtd.EntityPool
Status: PURE

Class: org.apache.xerces.validators.dtd.MixedContentModel
Status: PURE
```

```
Class: org.apache.xerces.validators.schema.XSchemaValidator
                                  $DatatypeValidatorRegistry
Status: PURE

Class: org.apache.xerces.validators.schema.XSchemaValidator
Status: PURE

Class: org.apache.xerces.validators.schema.DatatypeContentModel
Status: PURE

Class: org.apache.xerces.validators.schema.XUtil
Status: PURE

Class: org.apache.xerces.validators.schema.XSchemaValidator$ErrorHandler
Status: PURE

Class: org.apache.xerces.validators.schema.SchemaMessageProvider
Status: PURE

Class: org.apache.xerces.validators.schema.XSchemaValidator$Resolver
Status: PURE

Class: org.apache.xerces.validators.schema.XSchemaValidator$1
Status: PURE

Class: org.apache.xerces.utils.StringPool
Status: PURE

Class: org.apache.xerces.utils.SymbolCache
Status: PURE

Class: org.apache.xerces.utils.CharDataChunk
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xerces.utils.ChunkyByteArray
Status: PURE

Class: org.apache.xerces.utils.NamespacesScope$NamespacesHandler
Status: PURE

Class: org.apache.xerces.utils.XMLMessages
Status: PURE

Class: org.apache.xerces.utils.StringHasher
Status: PURE

Class: org.apache.xerces.utils.StringPool$StringProducer
Status: PURE

Class: org.apache.xerces.utils.XMLMessageProvider
Status: PURE

Class: org.apache.xerces.utils.ImplementationMessages
Status: PURE
```

```
Class: org.apache.xerces.utils.ChunkyCharArray
Status: PURE

Class: org.apache.xerces.utils.NamespacesScope
Status: PURE

Class: org.apache.xerces.utils.UTF8DataChunk
        Warning: method reference: java.lang.Class.forName(java.lang.String)
        Note: May load impure class
        Explanation:         <Explanation required>
Status: WARNING

Class: org.apache.xerces.utils.StringPool$CharArrayRange
Status: PURE

Class: org.apache.xerces.utils.XMLCharacterProperties
Status: PURE

Class: org.apache.xerces.msg.XMLMessages
Status: PURE

Class: org.apache.xerces.msg.SchemaMessages
Status: PURE

Class: org.apache.xerces.msg.ImplementationMessages
Status: PURE

Class: org.apache.xerces.msg.DatatypeMessages
Status: PURE

Class: org.apache.xerces.dom.DeferredNotationImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredEntityImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredNode
Status: PURE

Class: org.apache.xerces.dom.DeferredProcessingInstructionImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredDocumentTypeImpl
Status: PURE

Class: org.apache.xerces.dom.TreeWalkerImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredEntityReferenceImpl
Status: PURE

Class: org.apache.xerces.dom.NotationImpl
Status: PURE

Class: org.apache.xerces.dom.events.EventImpl
Status: PURE
```

```
Class: org.apache.xerces.dom.events.MutationEventImpl
Status: PURE

Class: org.apache.xerces.dom.DOMImplementationImpl
Status: PURE

Class: org.apache.xerces.dom.NodeImpl$EnclosingAttr
Status: PURE

Class: org.apache.xerces.dom.DeferredCommentImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredElementImpl
Status: PURE

Class: org.apache.xerces.dom.DocumentTypeImpl
Status: PURE

Class: org.apache.xerces.dom.NodeImpl$LEntry
Status: PURE

Class: org.apache.xerces.dom.DeferredElementNSImpl
Status: PURE

Class: org.apache.xerces.dom.ElementDefinitionImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredTextImpl
Status: PURE

Class: org.apache.xerces.dom.DocumentFragmentImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredDocumentImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredDocumentImpl$IntVector
Status: PURE

Class: org.apache.xerces.dom.CharacterDataImpl$1
Status: PURE

Class: org.apache.xerces.dom.CommentImpl
Status: PURE

Class: org.apache.xerces.dom.NodeIteratorImpl
Status: PURE

Class: org.apache.xerces.dom.DOMExceptionImpl
Status: PURE

Class: org.apache.xerces.dom.CharacterDataImpl
Status: PURE

Class: org.apache.xerces.dom.ElementNSImpl
Status: PURE

Class: org.apache.xerces.dom.AttrNSImpl
```

```
Status: PURE

Class: org.apache.xerces.dom.LCount
Status: PURE

Class: org.apache.xerces.dom.DeferredCDATASectionImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredAttrNSImpl
Status: PURE

Class: org.apache.xerces.dom.DeepNodeListImpl
Status: PURE

Class: org.apache.xerces.dom.NodeImpl
Status: PURE

Class: org.apache.xerces.dom.ProcessingInstructionImpl
Status: PURE

Class: org.apache.xerces.dom.AttrImpl
Status: PURE

Class: org.apache.xerces.dom.CDATASectionImpl
Status: PURE

Class: org.apache.xerces.dom.ElementImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredAttrImpl
Status: PURE

Class: org.apache.xerces.dom.NodeContainer
Status: PURE

Class: org.apache.xerces.dom.TextImpl
Status: PURE

Class: org.apache.xerces.dom.EntityReferenceImpl
Status: PURE

Class: org.apache.xerces.dom.DocumentImpl
Status: PURE

Class: org.apache.xerces.dom.DeferredElementDefinitionImpl
Status: PURE

Class: org.apache.xerces.dom.NamedNodeMapImpl
Status: PURE

Class: org.apache.xerces.dom.EntityImpl
Status: PURE

Class: org.w3c.dom.DocumentType
Status: PURE

Class: org.w3c.dom.html.HTMLBodyElement
Status: PURE
```

```
Class: org.w3c.dom.html.HTMLUListElement
Status: PURE

Class: org.w3c.dom.html.HTMLLabelElement
Status: PURE

Class: org.w3c.dom.html.HTMLObjectElement
Status: PURE

Class: org.w3c.dom.html.HTMLOptGroupElement
Status: PURE

Class: org.w3c.dom.html.HTMLDivElement
Status: PURE

Class: org.w3c.dom.html.HTMLBaseElement
Status: PURE

Class: org.w3c.dom.html.HTMLAnchorElement
Status: PURE

Class: org.w3c.dom.html.HTMLLIElement
Status: PURE

Class: org.w3c.dom.html.HTMLBRElement
Status: PURE

Class: org.w3c.dom.html.HTMLTitleElement
Status: PURE

Class: org.w3c.dom.html.HTMLFieldSetElement
Status: PURE

Class: org.w3c.dom.html.HTMLDOMImplementation
Status: PURE

Class: org.w3c.dom.html.HTMLFrameSetElement
Status: PURE

Class: org.w3c.dom.html.HTMLFrameElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableRowElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableColElement
Status: PURE

Class: org.w3c.dom.html.HTMLScriptElement
Status: PURE

Class: org.w3c.dom.html.HTMLHeadingElement
Status: PURE

Class: org.w3c.dom.html.HTMLMapElement
Status: PURE
```

```
Class: org.w3c.dom.html.HTMLTextAreaElement
Status: PURE

Class: org.w3c.dom.html.HTMLIFrameElement
Status: PURE

Class: org.w3c.dom.html.HTMLDocument
Status: PURE

Class: org.w3c.dom.html.HTMLModElement
Status: PURE

Class: org.w3c.dom.html.HTMLDirectoryElement
Status: PURE

Class: org.w3c.dom.html.HTMLMenuElement
Status: PURE

Class: org.w3c.dom.html.HTMLButtonElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableElement
Status: PURE

Class: org.w3c.dom.html.HTMLParagraphElement
Status: PURE

Class: org.w3c.dom.html.HTMLHRElement
Status: PURE

Class: org.w3c.dom.html.HTMLFontElement
Status: PURE

Class: org.w3c.dom.html.HTMLBaseFontElement
Status: PURE

Class: org.w3c.dom.html.HTMLLegendElement
Status: PURE

Class: org.w3c.dom.html.HTMLStyleElement
Status: PURE

Class: org.w3c.dom.html.HTMLOListElement
Status: PURE

Class: org.w3c.dom.html.HTMLOptionElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableCellElement
Status: PURE

Class: org.w3c.dom.html.HTMLFormElement
Status: PURE

Class: org.w3c.dom.html.HTMLLinkElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableCaptionElement
```

```
Status: PURE

Class: org.w3c.dom.html.HTMLInputElement
Status: PURE

Class: org.w3c.dom.html.HTMLMetaElement
Status: PURE

Class: org.w3c.dom.html.HTMLParamElement
Status: PURE

Class: org.w3c.dom.html.HTMLPreElement
Status: PURE

Class: org.w3c.dom.html.HTMLSelectElement
Status: PURE

Class: org.w3c.dom.html.HTMLCollection
Status: PURE

Class: org.w3c.dom.html.HTMLQuoteElement
Status: PURE

Class: org.w3c.dom.html.HTMLHeadElement
Status: PURE

Class: org.w3c.dom.html.HTMLAreaElement
Status: PURE

Class: org.w3c.dom.html.HTMLAppletElement
Status: PURE

Class: org.w3c.dom.html.HTMLHtmlElement
Status: PURE

Class: org.w3c.dom.html.HTMLIsIndexElement
Status: PURE

Class: org.w3c.dom.html.HTMLImageElement
Status: PURE

Class: org.w3c.dom.html.HTMLTableSectionElement
Status: PURE

Class: org.w3c.dom.html.HTMLElement
Status: PURE

Class: org.w3c.dom.html.HTMLDListElement
Status: PURE

Class: org.w3c.dom.ProcessingInstruction
Status: PURE

Class: org.w3c.dom.traversal.NodeIterator
Status: PURE

Class: org.w3c.dom.traversal.NodeFilter
Status: PURE
```

```
Class: org.w3c.dom.traversal.TreeWalker
Status: PURE

Class: org.w3c.dom.traversal.DocumentTraversal
Status: PURE

Class: org.w3c.dom.Attr
Status: PURE

Class: org.w3c.dom.CDATASection
Status: PURE

Class: org.w3c.dom.DocumentFragment
Status: PURE

Class: org.w3c.dom.Comment
Status: PURE

Class: org.w3c.dom.Text
Status: PURE

Class: org.w3c.dom.DOMImplementation
Status: PURE

Class: org.w3c.dom.Element
Status: PURE

Class: org.w3c.dom.DOMException
Status: PURE

Class: org.w3c.dom.EntityReference
Status: PURE

Class: org.w3c.dom.events.MutationEvent
Status: PURE

Class: org.w3c.dom.events.EventListener
Status: PURE

Class: org.w3c.dom.events.EventTarget
Status: PURE

Class: org.w3c.dom.events.EventException
Status: PURE

Class: org.w3c.dom.events.Event
Status: PURE

Class: org.w3c.dom.events.DocumentEvent
Status: PURE

Class: org.w3c.dom.Document
Status: PURE

Class: org.w3c.dom.NodeList
Status: PURE
```

```
    Class: org.w3c.dom.Notation
    Status: PURE

    Class: org.w3c.dom.NamedNodeMap
    Status: PURE

    Class: org.w3c.dom.CharacterData
    Status: PURE

    Class: org.w3c.dom.Node
    Status: PURE

    Class: org.w3c.dom.Entity
    Status: PURE

    Final Result   :  WARNING
```

10 Releases

May 19, 2000

- Reworked internals for XML Schema support [andyc, ericye, jeffreyr]
- Updated XML Schema support to April working draft [ericye, jeffreyr]
- Changed code to notify error handler of content model validation errors before calling endElement [lehors]

May 9, 2000

- Upgraded to SAX 2.0. [lehors]
- Added support for WML DOM. [david]
- Preliminary work for full schema support. [andyc, gmarcy, jefreyr, ericye]
- Reorganized DOM classes to use much less memory. [lehors]
- Entities and entity reference nodes are now readonly as expected. [lehors]
- Entity references now have their replacement value when created with createEntityReference. [lehors]
- Fixed problem in Deferred DOM which made building it N^2 order. [lehors]
- Fixed handling of elements with IDs in Deferred DOM. [lehors]
- Added support for namespaces in parameter entities. [lehors]
- setNodeValue raised an exception when it should simply be a no-op. [lehors]
- Attributes returned by setNamedItem and setNamedItemNS could not be reused. [lehors]
- Implemented new DOM Level 2 methods hasAttribute and hasAttributeNS. [lehors]
- 43: importNode now uses createElementNS and createAttributeNS when necessary. [lehors]
- 59: after a first iteration over the whole list, item(index) returned the wrong value. [lehors]
- 60: Fixed. [andyc]
- 62: Cloned attributes return wrong owner element. [awiner@us.oracle.com]
- Fixed caption element creation pb with HTMLTableElement. [arkin]
- Improved performance of XML serialize. Added support of encodings and reuse of serializer. [arkin]
- Updated XML serializer to not be strictly dependent on DOM Level 2. [andyc]

March 8, 2000

- 26: Schema lookup disabled when validation is off. [lehors]
- 23: With UCS-4 and UTF-16 encodings, 0xD characters are now properly normalized.[gmarcy]
- 27: SAX2 http://xml.org/sax/features/namespaces - default set to true. [pier]
- 28: Fixed error in setNamespaces javadoc [elharo@metalab.unc.edu]
- 33: Relaxed restraint of xml:space attribute for XHTML DTDs [david-b@pacbell.net]
- 34: SAX 2.0beta2 changes accepted. [rpfeiffe]
- 35: Fixed getLength() to always return the right value[lehors]
- 36: Fixed setPrefix() to update nodeName[lehors]

February 8, 2000

- Removed hard-coded strings to enable national language support (NLS) [jeffreyr,lehors,rip]

- Update for SAX2beta [andyc]
- Add new feature `http://apache.org/xml/features/dom/include-ignorable-whitespace` [andyc]
- ChunkyByteArray - needed resize chunk array for large files [andyc]
- UTF8DataChunk - memory perf fix [andyc]
- DOM L1 - cache node children length to accelerate `Node#item()`[andyc]
- DOM L1 - Off by one error in DeferredDocumentImpl [jflight@impresse.com]
- DOM L1 - Slight refactoring of DOM to save memory [lehors]
- Schema datatypes - add binary & URI, update decimal & String [twl]
- Schema datatypes - allow StringValidator to validate maxLength & enumeration facets [gtj@peakin.com]
- Schema datatypes - timeDuration & timeInstant validators [gtj@peakin.com]
- Schema functionality - allow schema lookup via installed entity resolver [gtj@peakin.com]
- Schema bug fixes (reportSchemaError & integer subtype registration) [gtj@peakin.com]
- Serializer bug on CData sections [arkin]
- Serializers now prints PUBLIC/SYSTEM id [arkin]
- HTMLTableElementImpl - Empty cell no longer created when adding a row [arkin]
- DOM L2 - TreeWalkerImpl didn't iterate backwards properly [bmj01@club-internet.fr]
- DOM L2 - Tracking the candidate recommendation [lehors]
- DOM L2 Namespace bug fixes [rip, lehors]
- DOM L2 API moved to org.w3c.dom where it belongs [rip]
- DOMMemTest [lehors]

December 31, 1999

- Include documentation in the repository.
- Switch packaging to jar files from .zip and .tar.gz files.
- StreamingCharReader for stream based applications.
- Assaf Arkin's serialization package.
- Assaf Arkin's HTML DOM Level 1 classes.
- Performance improvements.
- Bug fixes.

November 5, 1999

- Created initial code base from IBM's XML4J.
- Modified documentation to reflect new name (Xerces)

11
Caveats

Caveats and Limitations

This is a list of the limitations in this release of Xerces-J:

- Due to a transcoding bug in the Sun JDK's (1.1.6, 1.1.7, 1.1.8, 1.2) handling of EBCDIC data (specifically, end of line characters), you must use IBM's JDK 1.1.6, if you want to use EBCDIC support. The IBM JDK has the bug fixed. We don't know when the bug fix will get into the Sun JDK's.

12
Feedback Procedures

Questions or Comments

For all questions or comments, write to the Xerces-J mailing list.

If you are submitting a bug (and bug reports are definitely appreciated!), please provide the following information:

- Version number of Xerces-J (1.1.0?)
- Version number of JDK (1.1.8? 1.2?)
- Sample XML file that causes the bug
- Sample Schema file (if required to recreate the bug)
- Sample DTD file (if required to recreate the bug)

13
Y2K Compliance

Apache Xerces Parser Year-2000 Readiness

Q: Are the Xerces parsers Year-2000-compliant?

Yes, Xerces-J and Xerces-C are Year 2000 compliant. They do not currently use any dates at all (at least until the XML Schema date datatypes are fully supported). However, you may still have Y2K problems if the underlying OS or Java implementation has problems with dates past year 2000 (e.g. OS calls which accept or return year numbers).

Most (UNIX) systems store dates internally as signed 32-bit integers which contain the number of seconds since 1st January 1970, so the magic boundary to worry about is the year 2038 and not 2000. But modern operating systems shouldn't cause any trouble at all.

The Apache Xerces project is an open-source software product of the Apache Software Foundation. The project and the Foundation cannot and does not offer legal assurances regarding any suitability of the software for your application. There are several commercial support organizations and derivative products available that may be able to certify the software and provide you with any assurances you may require (IBM's Websphere product is one of them).

The Apache HTTP server software is distributed with the following disclaimer, found in the software license:

```
    THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
    WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
    OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
    DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
    ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
    SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
    LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
    USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
    ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
    OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
    OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
    SUCH DAMAGE.
```