Oracle8*i*

XML Reference Guide

Release 3 (8.1.7)

September 2000

Part No. A83730-01

Oracle*8i* XML Reference, Release 3 (8.1.7)

Part No. A83730-01

Primary Author: Chitra Sharma

Contributing Author: Shelley Higgins

Contributors: Stefan Kiritzov, Visar Nimani, Mark Scardina

Contents

Send Us Your Comments xv

Preface xvii

What This Guide Has to Offer xviii
Your Comments Are Welcome xviii

Part I XDK for Java Packages

1 XML Parser for Java

Description 1-2
AttrDecl 1-4
DefaultXMLDocumentHandler 1-9
DOMParser 1-15
DTD 1-22
ElementDecl 1-30
NodeFactory 1-35
NSName 1-38
NSResolver 1-41
oraxsl 1-42
SAXAttrList 1-44
SAXParser 1-50
XMLAttr 1-54
XMLCDATA 1-62
XMLComment 1-65

XMLDocument 1-68
XMLDocumentFragment 1-82
XMLDocumentHandler 1-85
XMLElement 1-90
XMLEntityReference 1-102
XMLNode 1-104
XMLParseException 1-117
XMLParser 1-121
XMLPI 1-127
XMLText 1-130
XMLToken 1-134
XMLTokenizer 1-140
XSLException 1-146
XSLProcessor 1-147
XSLStylesheet 1-153

2 XML Class Generator for Java

Description 2-1
oracle.xml.classgen 2-1
    CGDocument 2-1
oracle.xml.classgen 2-2
    CGNode 2-2
    Direct Known Subclasses: 2-2
    Description 2-2
ClassGenerator 2-4
InvalidContentException 2-7

3 XSQL Servlet

Res 3-2
XSQLActionHandler 3-12
XSQLActionHandlerImpl 3-15
XSQLCommandLine 3-17
XSQLDiagnostic 3-19
XSQLHttpUtil 3-21
XSQLPageRequest 3-23

XSQLPageRequestImpl 3-29
XSQLParserHelper 3-38
XSQLRequest 3-40
XSQLServlet 3-45
XSQLServletPageRequest 3-47
XSQLStylesheetProcessor 3-51
XSQLUtil 3-53

4 Transviewer Beans

Description 4-2
DOMBuilder 4-3
parseDTD(Reader, String) 4-12
parseDTD(String, String) 4-12
DOMBuilderBeanInfo 4-18
DOMBuilderErrorEvent 4-20
DOMBuilderErrorListener 4-26
DOMBuilderEvent 4-27
DOMBuilderListener 4-29
ResourceManager 4-31
XSLTransformer 4-33
XSLTransformerBeanInfo 4-40
XSLTransformerErrorEvent 4-42
XSLTransformerErrorListener 4-44
XSLTransformerEvent 4-45
XSLTransformerListener 4-47
oracle.xml.srcviewer Package 4-48
  Description 4-48
XMLSourceViewBeanInfo 4-65
oracle.xml.async Package 4-67
  Description 4-67
DOMBuilder 4-68
DOMBuilderBeanInfo 4-80
DOMBuilderErrorEvent 4-82
DOMBuilderErrorListener 4-84
DOMBuilderEvent 4-85

DOMBuilderListener 4-87
ResourceManager 4-89
XSLTransformer 4-91
XSLTransformerBeanInfo 4-96
XSLTransformerErrorEvent 4-98
XSLTransformerErrorListener 4-100
XSLTransformerEvent 4-101
XSLTransformerListener 4-103
oracle.xml.transviewer Package 4-104
    Description 4-104
DBAccess 4-105
XMLTransformPanel 4-111
XMLTransformPanelBeanInfo 4-114
XMLTransViewer 4-116
oracle.xml.treeviewer Package 4-117
    Description 4-117
XMLTreeView 4-118
XMLTreeViewBeanInfo 4-122
oracle.xml.srcviewer Package 4-123
    Description 4-123
XMLSourceViewBeanInfo 4-139
oracle.xml.async Package 4-142
    Description 4-142
DOMBuilder 4-143
DOMBuilderBeanInfo 4-155
DOMBuilderErrorEvent 4-157
DOMBuilderErrorListener 4-159
DOMBuilderEvent 4-160
DOMBuilderListener 4-162
ResourceManager 4-164
XSLTransformer 4-166
XSLTransformerBeanInfo 4-171
XSLTransformerErrorEvent 4-173
XSLTransformerErrorListener 4-175
XSLTransformerEvent 4-176

XSLTransformerListener 4-178
oracle.xml.transviewer Package 4-179
Description 4-179
DBAccess 4-180
XMLTransformPanel 4-186
XMLTransformPanelBeanInfo 4-189
XMLTransViewer 4-191
oracle.xml.treeviewer Package 4-192
Description 4-192
XMLTreeView 4-193
XMLTreeViewBeanInfo 4-197
Description 4-199
XMLTreeView 4-200
XMLTreeViewBeanInfo 4-204

Part I XDK for PL/SQL Packages

5 XML Parser for PL/SQL

PL/SQL Parser APIs 5-2
Types and Functions 5-3
Parser Interface Type Description 5-5
Function Prototypes 5-5
Extensible Stylesheet Language (XSL) Package Processor APIs 5-13
Functions 5-14
W3C Document Object Model (DOM) APIs 5-21
Types 5-23
DOM Node Types 5-23
DOM Exception Types 5-23
DOM Interface Types 5-24
Methods 5-25
Node Methods 5-25
Named Node Map Methods 5-31
Node List Methods 5-33
Attr Methods 5-34
C Data Section Methods 5-36

Character Data Methods 5-37
Comment Methods 5-38
DOM Implementation Methods 5-38
Document Fragment Methods 5-39
Document Type Methods 5-39
Element Methods 5-42
Entity Methods 5-46
Entity Reference Methods 5-47
Notation Methods 5-48
Processing Instruction Methods 5-49
Text Methods 5-49
Document Methods 5-51
Method Prototypes 5-56
Node Methods 5-56
Named Node Map Methods 5-65
Node List Methods 5-66
Attr Methods 5-67
C Data Section Methods 5-69
Character Data Methods 5-70
Comment Methods 5-71
DOM Implementation Methods 5-72
Document Fragment Methods 5-73
Document Type Methods 5-74
Element Methods 5-80
Entity Methods 5-84
Entity Reference Methods 5-85
Notation Methods 5-86
Processing Instruction Methods 5-87
Text Methods 5-88
Document Methods 5-89
Interface org.w3c.dom.Attr 5-98
Abstracts 5-99
Interface org.w3c.dom.CDATASection 5-100
Interface org.w3c.dom.CharacterData 5-101
Abstracts 5-102

Interface org.w3c.dom.Comment 5-105
Interface org.w3c.dom.Document 5-105
Abstracts 5-106
Abstracts 5-108
Interface org.w3c.dom.DocumentFragment 5-110
Interface org.w3c.dom.DocumentType 5-111
Abstracts 5-112
Class org.w3c.dom.DOMException 5-113
Interface org.w3c.dom.DOMImplementation 5-114
Abstracts 5-115
Interface org.w3c.dom.Element 5-115
Abstracts 5-117
Abstracts 5-119
Interface org.w3c.dom.Entity 5-120
Abstracts 5-121
Interface org.w3c.dom.EntityReference 5-122
Interface org.w3c.dom.NamedNodeMap 5-122
Abstracts 5-123
Interface org.w3c.dom.Node 5-125
Interface org.w3c.dom.NodeList 5-133
Abstracts 5-133
Interface org.w3c.dom.Notation 5-134
Abstracts 5-135
Interface org.w3c.dom.ProcessingInstruction 5-135
Abstracts 5-135
Interface org.w3c.dom.Text 5-136
Abstracts 5-136
.... 5-137

Part II XDK for C Packages

6 XML Parser for C

Parser APIs 6-2
Calling Sequence 6-3
Memory 6-3

Thread Safety 6-3
Data Types Index 6-4
Function Index 6-4
Data Structures and Types 6-5
Functions 6-7
XSLT API 6-11
Data Structures and Types 6-13
Functions 6-14
Data Structure and Type Description 6-15
Function Prototypes 6-16
W3C SAX APIs 6-16
Data Structures and Types 6-18
Non-SAX Callback Functions 6-19
Data Structure and Type Description 6-20
Function Prototypes 6-21
W3C DOM APIs 6-26
Data Structures and Types 6-30
Function Prototypes 6-32
Namespace APIs 6-77
Data Structures and Types 6-79
Functions 6-80
Data Structure and Type Description 6-81
Function Prototypes 6-82

Part III XDK for C++ Packages

7 XML Parser for C++

Class: Attr 7-2
Class: CDATASection 7-5
Class: Comment 7-6
Class: Document 7-7
Class: DocumentType 7-12
Class: DOMImplementation 7-14
Class: Element 7-15
Class: Entity 7-20

Class: EntityReference 7-22
Class: NamedNodeMap 7-23
Class: Node 7-26
Class: NodeList 7-38
Class: Notation 7-39
Class: ProcessingInstruction 7-41
Class: Text 7-43
Class: XMLParser 7-44
C++ Parser APIs 7-49
C++ SAX API 7-49
SAX callback structure 7-49
C++ DOM API's 7-56

8 Oracle XML Class Generator 1.0.0.0.0 (C++)

.... 8-1
Overview of the XML Class Generator for C++ 8-2
Input 8-2
Output 8-3
Relevant XML Standards 8-3
sample/Example usage 8-3
METHODS 8-5
generate 8-5
METHOD INDEX 8-7
METHOD 8-8
Constructors 8-8

Part IV XML-SQL Utility (XSU) Packages

9 Oracle XSU Java API

oracle.xml.sql.dml.OracleXMLSave 9-2

10 XSU’s PL/SQL API

DBMS_XMLQuery 10-2
Function and Procedure Index: 10-4

Functions and Procedures: 10-8
DBMS_XMLSave 10-18
Types.......... 10-18
Functions and Procedures: 10-22

Index

Send Us Your Comments

Oracle*8i* Application Developer's Guide - XML, Release 3 (8.1.7)

Part No. A83730-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: infodev@us.oracle.com
- FAX: (650) 506- 7228 Attn: Server Technologies, Information Development
- Postal service:
  Oracle Corporation
  Server Technologies - Information development
  500 Oracle Parkway MS4op12
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This preface discusses the following topics:

- What This Guide Has to Offer
- Your Comments Are Welcome

What This Guide Has to Offer

This reference book presents XML supplied with Oracle8*i*. XDK for Java, PL/SQL , C and C++ Packages with examples in separate sections. The final section presents XSU for Java and PL/SQL.

For more information, XML users should see:

- *Oracle8i Application Developer's Guide - Fundamentals*
- Oracle8i Application Developer's Guide - XML

Your Comments Are Welcome

The Oracle Corporation technical staff values your comments. As we write and revise, your opinions are the most important feedback we receive. Please use the Reader's Comment Form to tell us what you like and dislike about this Oracle publication. You can send comments to us in the following ways:

- E-mail: infodev@us.oracle.com
- FAX: (650) 506-7228 Attn: Information Development Department
- Letter:

  Oracle Corporation
  Information Development Department
  500 Oracle Parkway MS 4OP12
  Redwood Shores, CA 94065
  USA

Part I

XDK for Java Packages

This section contains the following chapters:

- Chapter 1, "XML Parser for Java"
- Chapter 2, "XML Class Generator for Java"
- Chapter 3, "XSQL Servlet"
- Chapter 4, "Transviewer Beans"

1 XML Parser for Java

Package Oracle.xml.parser.v2 Description

Class Summary

Interfaces	
NSName	This interface provides Namespace support for Element and Attr names
NSResolver	This interface provides support for resolving Namespaces
XMLDocumentHandler	This interface extends the `org.xml.sax.DocumentHandler` interface.
XMLToken	Basic interface for XMLToken
Classes	
AttrDecl	This class hold information about each attribute declared in an attribute list in the Document Type Definition.
Package Oracle.xml.parser.v2	This class implements the default behaviour for the `XMLDocumentHandler` interface.
DOMParser	This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.
DTD	Implements the DOM DocumentType interface and holds the Document Type Definition information for an XML document.
ElementDecl	This class represents an element declaration in a DTD.
NodeFactory	This class specifies methods to create various nodes of the DOM tree built during parsing.
oraxsl	The oraxsl class provides a command-line interface to applying stylesheets on multiple XML documents.
SAXAttrList	This class implements the SAX `AttributeList` interface and also provides Namespace support.
SAXParser	This class implements an eXtensible Markup Language (XML) 1.0 SAX parser according to the World Wide Web Consortium (W3C) recommendation.
XMLAttr	This class implements the DOM Attr interface and holds information on each attribute of an element.
XMLCDATA	This class implements the DOM CDATASection interface.
XMLComment	This class implements the DOM Comment interface.

Class Summary	
XMLDocument	This class implements the DOM Document interface, represents an entire XML document and serves the root of the Document Object Model tree.
XMLDocumentFragment	This class implements the DOM DocumentFragment interface.
XMLElement	This class implements the DOM `Element` interface.
XMLEntityReference	
XMLNode	Implements the DOM `Node` interface and serves as the primary datatype for the entire Document Object Model.
XMLParser	This class serves as a base class for the `DOMParser` and `SAXParser` classes.
XMLPI	This class implements the DOM Processing Instruction interface.
XMLText	This class implements the DOM Text interface.
XMLTokenizer	This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.
XSLProcessor	This class provides methods to transform an input XML document using a previously constructed `XSLStylesheet`.
XSLStylesheet	The class holds XSL stylesheet information such as templates, keys, variables, and attribute sets.
Exceptions	
XMLParseException	Indicates that a parsing exception occurred while processing an XML document
XSLException	Indicates that an exception occurred during XSL tranformation

oracle.xml.parser.v2
AttrDecl

Syntax

```
public class AttrDecl extends XMLNode implements
oracle.xml.parser.v2.XMLConstants, java.io.Serializable

java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.AttrDecl
```

All Implemented Interfaces

`java.lang.Cloneable, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants`

Description

This class holds information about each attribute declared in an attribute list in the Document Type Definition.

Member Summary

Fields	
CDATA	AttType - StringType - CDATA
DEFAULT	Attribute presence - Default
ENTITIES	AttType - TokenizedType - Entities
ENTITY	AttType - TokenizedType - Entity
ENUMERATION	AttType - EnumeratedType - Enumeration
FIXED	Attribute presence - Fixed
ID	AttType - TokenizedType - ID
IDREF	AttType - TokenizedType - ID reference
IDREFS	AttType - TokenizedType - ID references
IMPLIED	Attribute presence - Implied

Member Summary

NMTOKEN	AttType - TokenizedType - Name token
NMTOKENS	AttType - TokenizedType - Name tokens
NOTATION	AttType - EnumeratedType - Notation
REQUIRED	Attribute presence - Required
Methods	
getAttrPresence()	Gets attribute presence
getAttrType()	Gets attribute type
getDefaultValue()	Gets attribute default value
getEnumerationValues()	Gets attribute values

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface `Node`

Inherited Member Summary

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Fields

CDATA

`public static final int CDATA`

AttType - StringType - CDATA

DEFAULT

`public static final int DEFAULT`

Attribute presence - Default

ENTITIES

`public static final int ENTITIES`

AttType - TokenizedType - Entities

ENTITY

`public static final int ENTITY`

AttType - TokenizedType - Entity

ENUMERATION

`public static final int ENUMERATION`

AttType - EnumeratedType - Enumeration

FIXED

`public static final int FIXED`

Attribute presence - Fixed

ID

`public static final int ID`

AttType - TokenizedType - ID

IDREF

`public static final int IDREF`

AttType - TokenizedType - ID reference

IDREFS

`public static final int IDREFS`

AttType - TokenizedType - ID references

IMPLIED

`public static final int IMPLIED`

Attribute presence - Implied

NMTOKEN

`public static final int NMTOKEN`

AttType - TokenizedType - Name token

NMTOKENS

`public static final int NMTOKENS`

AttType - TokenizedType - Name tokens

NOTATION

`public static final int NOTATION`

AttType - EnumeratedType - Notation

REQUIRED

`public static final int REQUIRED`

Attribute presence - Required

Methods

getAttrPresence()

`public int getAttrPresence()`

Gets attribute presence

Returns

The presence of the attribute

getAttrType()

`public int getAttrType()`

Gets attribute type

Returns

The type of the attribute

getDefaultValue()

`public java.lang.String getDefaultValue()`

Gets attribute default value

Returns

The default value of the attribute

getEnumerationValues()

`public java.util.Vector getEnumerationValues()`

Gets attribute values

Returns

The values of the attribute as an `Enumeration`

oracle.xml.parser.v2 **DefaultXMLDocumentHandler**

Syntax

`public class DefaultXMLDocumentHandler extends` org.xml.sax.HandlerBase `implements XMLDocumentHandler`

```
java.lang.Object
  |
  +--org.xml.sax.HandlerBase
        |
        +--oracle.xml.parser.v2.DefaultXMLDocumentHandler
```

Direct Known Subclasses:

`XMLTokenizer`

All Implemented Interfaces

org.xml.sax.DocumentHandler, org.xml.sax.DTDHandler, org.xml.sax.EntityResolver, org.xml.sax.ErrorHandler, `XMLDocumentHandler`

Description

This class implements the default behaviour for the `XMLDocumentHandler` interface.

Application writers can extend this class when they need to implement only part of the interface

Member Summary

Constructors	
DefaultXMLDocumentHandler()	Constructs a default document handler
Methods	
cDATASection(char[], int, int)	Receive notification of a CDATA Section.
comment(String)	Receive notification of a comment.
endDoctype()	Receive notification of end of the DTD.
endElement(NSName)	Receive notification of the end of an element.

Member Summary	
setDoctype(DTD)	Receive notification of DTD.
setTextDecl(String, String)	Receive notification of a Text XML Declaration.
setXMLDecl(String, String, String)	Receive notification of an XML Declaration.
startElement(NSName, SAXAttrList)	Receive notification of the beginning of an element.

Inherited Member Summary

Methods inherited from class HandlerBase

characters(char[], int, int), endDocument(), endElement(String), error(SAXParseException), fatalError(SAXParseException), ignorableWhitespace(char[], int, int), notationDecl(String, String, String), processingInstruction(String, String), resolveEntity(String, String), setDocumentLocator(Locator), startDocument(), startElement(String, AttributeList), unparsedEntityDecl(String, String, String, String), warning(SAXParseException)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface DocumentHandler

characters(char[], int, int), endDocument(), endElement(String), ignorableWhitespace(char[], int, int), processingInstruction(String, String), setDocumentLocator(Locator), startDocument(), startElement(String, AttributeList)

Methods inherited from interface EntityResolver

resolveEntity(String, String)

Methods inherited from interface DTDHandler

notationDecl(String, String, String), unparsedEntityDecl(String, String, String, String)

Methods inherited from interface ErrorHandler

error(SAXParseException), fatalError(SAXParseException), warning(SAXParseException)

Constructor

DefaultXMLDocumentHandler()

```
public  DefaultXMLDocumentHandler()
```

Constructs a default document handler

Methods

cDATASection(char[], int, int)

`public void cDATASection(char[] ch, int start, int length)`
Receive notification of a CDATA Section.

The Parser will invoke this method once for each CDATA Section found.

Specified By

cDATASection(char[ ], int, int) in interface XMLDocumentHandler

Parameters

`ch` - The CDATA section characters.

`start` - The start position in the character array.

`length` - The number of characters to use from the character array.

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

comment(String)

`public void comment(java.lang.String data)`
Receive notification of a comment.

The Parser will invoke this method once for each comment found: note that comment may occur before or after the main document element.

Specified By

comment(String) in interface XMLDocumentHandler

Parameters

`data` - The comment data, or null if none was supplied.

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

endDoctype()

`public void endDoctype()`

Receive notification of end of the DTD.

Specified By

endDoctype() in interface XMLDocumentHandler

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

endElement(NSName)

`public void endElement(NSName elem)`

Receive notification of the end of an element. The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding startElement() event for every endElement() event (even when the element is empty).

By implementing this method instead of `org.xml.sax.DocumentHandler.endElement`, SAX Applications can get the Namespace support provided by `NSName`.

Specified By

endElement(NSName) in interface XMLDocumentHandler

Parameters

`elem` - NSName object

Throws

org.xml.sax.SAXException - A `SAXException` could be thrown.

See Also

org.xml.sax.DocumentHandler.endElement(String)

setDoctype(DTD)

`public void setDoctype(DTD dtd)`

Receive notification of DTD. Sets the DTD

Specified By

setDoctype(DTD) in interface XMLDocumentHandler

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

setTextDecl(String, String)

`public void setTextDecl(java.lang.String version, java.lang.String encoding)`

Receive notification of a Text XML Declaration.

The Parser will invoke this method once for each text XML Decl

Specified By

setTextDecl(String, String) in interface XMLDocumentHandler

Parameters

`version` - The version number (or null, if not specified)

`encoding` - The encoding name

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

setXMLDecl(String, String, String)

`public void setXMLDecl(java.lang.String version, java.lang.String standalone, java.lang.String encoding)`

Receive notification of an XML Declaration.

The Parser will invoke this method once for XML Decl

Specified By

setXMLDecl(String, String, String) in interface XMLDocumentHandler

Parameters

`version` - The version number

`standalone` - The standalone value (or null, if not specifed) `encoding` - The encoding name (or null, if not specifed)

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

startElement(NSName, SAXAttrList)

`public void startElement(NSName elem, SAXAttrList attrlist)`

Receive notification of the beginning of an element. The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding endElement() event for every startElement() event (even when the element is empty). All of the element's content will be reported, inorder, before the corresponding endElement() event.

By implementing this method instead of `org.xml.sax.DocumentHandler.startElement`, SAX Applications can get the Namespace support provided by `NSName` and `SAXAttrList`.

Specified By

startElement(NSName, SAXAttrList) in interface XMLDocumentHandler

Parameters

`elem` - NSName object

`attrlist` - SAXAttrList for the element

Throws

org.xml.sax.SAXException - A `SAXException` could be thrown.

See Also

org.xml.sax.DocumentHandler.startElement(String, AttributeList)

oracle.xml.parser.v2
DOMParser

Syntax

```
public class DOMParser extends XMLParser implements
oracle.xml.parser.v2.XMLConstants

java.lang.Object
  |
  +--XMLParser
        |
        +--oracle.xml.parser.v2.DOMParser
```

All Implemented Interfaces

```
oracle.xml.parser.v2.XMLConstants
```

Description

This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation to parse a XML document and build a DOM tree.

Member Summary

Constructors	
DOMParser()	Creates a new parser object.
Methods	
getDoctype()	Get the DTD
getDocument()	Gets the document
parseDTD(InputSource, String)	Parses the XML External DTD from given input source
parseDTD(InputStream, String)	Parses the XML External DTD from given input stream.
parseDTD(Reader, String)	Parses the XML External DTD from given input stream.
parseDTD(String, String)	Parses the XML External DTD from the URL indicated
parseDTD(URL, String)	Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.
setErrorStream(OutputStream)	Creates an output stream for the output of errors and warnings.

Member Summary	
setErrorStream(OutputStream, String)	Creates an output stream for the output of errors and warnings.
setErrorStream(PrintWriter)	Creates an output stream for the output of errors and warnings.
setNodeFactory(NodeFactory)	Set the node factory.
showWarnings(boolean)	Switch to determine whether to print warnings

Inherited Member Summary

Fields inherited from class XMLParser

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLParser

getReleaseVersion(), getValidationMode(), parse(InputSource), parse(InputStream), parse(Reader), parse(String), parse(URL), setBaseURL(URL), setDoctype(DTD), setLocale(Locale), setPreserveWhitespace(boolean), setValidationMode(boolean)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor

DOMParser()

`public  DOMParser()`
Creates a new parser object.

Methods

getDoctype()

`public DTD getDoctype()`
Get the DTD

Returns

The `DTD`

getDocument()

`public XMLDocument getDocument()`
Gets the document

Returns

The document being parsed

parseDTD(InputSource, String)

`public final void parseDTD(`org.xml.sax.InputSource `in, java.lang.String rootName)`
Parses the XML External DTD from given input source

Parameters

`in` - the `org.xml.sax.InputSouce` to parse

`rootName` - the element to be used as root Element

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputStream, String)

`public final void parseDTD(java.io.InputStream in, java.lang.String rootName)`
Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters

`in` - the `InputStream` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also

setBaseURL(URL)

parseDTD(Reader, String)

`public final void parseDTD(java.io.Reader r, java.lang.String rootName)`
Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters

`r` - the `Reader` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also

setBaseURL(URL)

parseDTD(String, String)

`public final void parseDTD(java.lang.String in, java.lang.String rootName)`
Parses the XML External DTD from the URL indicated

Parameters

`in` - the `String` containing the URL to parse from

`rootName` - the element to be used as root Element

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(URL, String)

`public final void parseDTD(java.net.URL url, java.lang.String rootName)`
Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters

`url` - the url points to the XML document to parse.

`rootName` - the element to be used as root Element

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

setErrorStream(OutputStream)

`public final void setErrorStream(java.io.OutputStream out)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters

`out` - The output stream to use for errors and warnings

setErrorStream(OutputStream, String)

`public final void setErrorStream(java.io.OutputStream out, java.lang.String enc)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings. Additionally, an .exception is thrown if the encoding specified is unsupported.

Parameters

`out` - The output stream to use for errors and warnings

`enc` - the encoding to use

Throws

`IOException` - if an unsupported encoding is specified

setErrorStream(PrintWriter)

`public final void setErrorStream(java.io.PrintWriter out)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters

`out` - The `PrintWriter` to use for errors and warnings

setNodeFactory(NodeFactory)

`public void setNodeFactory(NodeFactory factory)`
Set the node factory. Applications can extend the NodeFactory and register it through this method. The parser will then use the user supplied NodeFactory to create nodes of the DOM tree.

Parameters

`factory` - The `NodeFactory` to set

Throws

XMLParseException - if an invalid factory is set

See Also

NodeFactory

showWarnings(boolean)

`public void showWarnings(boolean yes)`

Switch to determine whether to print warnings

Parameters

`yes` - determines whether warnings should be shown

oracle.xml.parser.v2
DTD

Syntax

public class DTD extends XMLNode implements org.w3c.dom.DocumentType, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.DTD
```

All Implemented Interfaces

java.lang.Cloneable, org.w3c.dom.DocumentType, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

Implements the DOM DocumentType interface and holds the Document Type Definition information for an XML document.

Member Summary

Methods

cloneNode(boolean)	Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes.
findElementDecl(String)	Finds an element declaration for the given tag name.
findEntity(String, boolean)	Finds a named entity in the DTD.
findNotation(String)	Retrieves the named notation from the DTD.
getChildNodes()	A `NodeList` that contains all children of this node.
getElementDecls()	A `NamedNodeMap` containing the element declarations in the DTD.
getEntities()	A `NamedNodeMap` containing the general entities, both external and internal, declared in the DTD.
getName()	Gets the name of the DTD; i.e., the name immediately following the `DOCTYPE` keyword.
getNotations()	A `NamedNodeMap` containing the notations declared in the DTD.

Member Summary	
getPublicId()	Gets The public identifier associated with the DTD, if specified.
getSystemId()	Gets the system identifier associated with the DTD, if specified.
hasChildNodes()	This is a convenience method to allow easy determination of whether a node has any children.
printExternalDTD(OutputStream)	Writes the contents of this document to the given output stream.
printExternalDTD(OutputStream, String)	Writes the contents of the external DTD to the given output stream.
printExternalDTD(PrintWriter)	Writes the contents of this document to the given output stream.

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

Inherited Member Summary

appendChild(Node), getAttributes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), getAttributes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Methods

cloneNode(boolean)

`public` org.w3c.dom.Node `cloneNode(boolean deep)`
Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `null`.). Cloning an `Element` copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child `Text` node. Cloning any other type of node simply returns a copy of this node.

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Overrides

cloneNode(boolean) in class XMLNode

Parameters

`deep` - If `true`, recursively clone the subtree under the specified node; if `false`, clone only the node itself (and its attributes, if it is an `Element`).

Returns

The duplicate node.

findElementDecl(String)

`public final ElementDecl findElementDecl(java.lang.String name)`
Finds an element declaration for the given tag name.

Parameters

`name` - The tag name.

Returns

the element declaration object.

findEntity(String, boolean)

`public final org.w3c.dom.Entity findEntity(java.lang.String n, boolean par)`
Finds a named entity in the DTD.

Parameters

`n` - The name of the entity.

Returns

the specified `Entity` object; returns null if it is not found.

findNotation(String)

`public final org.w3c.dom.Notation findNotation(java.lang.String name)`
Retrieves the named notation from the DTD.

Parameters

`name` - The name of the notation.

Returns

the `Notation` object; returns null if it is not found.

getChildNodes()

`public org.w3c.dom.NodeList getChildNodes()`
A `NodeList` that contains all children of this node. If there are no children, this is a `NodeList` containing no nodes. The content of the returned `NodeList` is "live" in the sense that, for instance, changes to the children of the node object that it was created from are immediately reflected in the nodes returned by the `NodeList` accessors; it is not a static snapshot of the content of the node. This is true for every

`NodeList`, including the ones returned by the `getElementsByTagName` method.

Specified By

org.w3c.dom.Node.getChildNodes() in interface org.w3c.dom.Node

Overrides

getChildNodes() in class XMLNode

Returns

The children of this node

getElementDecls()

`public` org.w3c.dom.NamedNodeMap `getElementDecls()`
A `NamedNodeMap` containing the element declarations in the DTD. Every node in this map is an `ElementDecl` object.

Returns

The element declarations in the DTD The DOM Level 1 does not support editing elementdecls, therefore `elementdecls` cannot be altered in any way.

getEntities()

`public` org.w3c.dom.NamedNodeMap `getEntities()`
A `NamedNodeMap` containing the general entities, both external and internal, declared in the DTD. Duplicates are discarded. For example in:<!DOCTYPE ex SYSTEM "ex.dtd" [<!ENTITY foo "foo"> <!ENTITY bar "bar"> <!ENTITY % baz "baz">]> <ex/> the interface provides access to `foo` and `bar` but not `baz`. Every node in this map also implements the `Entity` interface. The DOM Level 1 does not support editing entities, therefore `entities` cannot be altered in any way.

Specified By

org.w3c.dom.DocumentType.getEntities() in interface org.w3c.dom.DocumentType

Returns

The entities declared in the DTD

getName()

`public java.lang.String getName()`
Gets the name of the DTD; i.e., the name immediately following the `DOCTYPE` keyword.

Specified By

org.w3c.dom.DocumentType.getName() in interface org.w3c.dom.DocumentType

Returns

Name of the DTD

getNotations()

`public` org.w3c.dom.NamedNodeMap `getNotations()`
A `NamedNodeMap` containing the notations declared in the DTD. Duplicates are discarded. Every node in this map also implements the `Notation` interface. The DOM Level 1 does not support editing notations, therefore `notations` cannot be altered in any way.

Specified By

org.w3c.dom.DocumentType.getNotations() in interface org.w3c.dom.DocumentType

Returns

The notations declared in the DTD

getPublicId()

`public java.lang.String getPublicId()`
Gets The public identifier associated with the DTD, if specified. If the public identifier was not specified, this is `null`.

Returns

the public identifier associated with the DTD

getSystemId()

`public java.lang.String getSystemId()`
Gets the system identifier associated with the DTD, if specified. If the system identifier was not specified, this is `null`.

Returns

the system identifier associated with the DTD

hasChildNodes()

`public boolean hasChildNodes()`

This is a convenience method to allow easy determination of whether a node has any children. return false always, as DTD cannot have any overrides method in XMLNode

Specified By

org.w3c.dom.Node.hasChildNodes() in interface org.w3c.dom.Node

Overrides

hasChildNodes() in class XMLNode

Returns

`false` as DTD node can not have any children,

printExternalDTD(OutputStream)

`public void printExternalDTD(java.io.OutputStream out)`

Writes the contents of this document to the given output stream.

Parameters

`out` - `OutputStream` to write to

Throws

`IOException` - if an error occurs

printExternalDTD(OutputStream, String)

`public void printExternalDTD(java.io.OutputStream out, java.lang.String enc)`

Writes the contents of the external DTD to the given output stream.

Parameters

`out` - `OutputStream` to write to

`enc` - Encoding to use for the output

Throws

`IOException` - if an invalid encoding was specified or if any other error occurs

printExternalDTD(PrintWriter)

```
public void printExternalDTD(java.io.PrintWriter out)
```

Writes the contents of this document to the given output stream.

Parameters

`out` - `PrintWriter` to write to

Throws

`IOException` - if an error occurs

oracle.xml.parser.v2
ElementDecl

Syntax

```
public class ElementDecl extends XMLNode implements java.io.Serializable

java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.ElementDecl
```

All Implemented Interfaces

`java.lang.Cloneable`, org.w3c.dom.Node, `java.io.Serializable`, `oracle.xml.parser.v2.XMLConstants`

Description

This class represents an element declaration in a DTD.

Member Summary

Fields	
ANY	Element content type - Children can be any element
ASTERISK	ContentModelParseTreeNode type - "*" node (has one children)
COMMA	ContentModelParseTreeNode type - "," node (has two children)
ELEMENT	ContentModelParseTreeNode type - 'leaf' node (has no children)
ELEMENTS	Element content type - Children can be elements as per Content Model
EMPTY	Element content type - No Children
MIXED	Element content type - Children can be PCDATA & elements as per Content Model
OR	ContentModelParseTreeNode type - "|" node (has two children)
PLUS	ContentModelParseTreeNode type - "+" node (has one children)
QMARK	ContentModelParseTreeNode type - "?" node (has one children)
Methods	

Member Summary

expectedElements(Element)	Returns vector of element names that can be appended to the element.
findAttrDecl(String)	Gets an attribute declaration object or null if not found
getAttrDecls()	Gets an enumeration of attribute declarations
getContentElements()	Returns Vector of elements that can be appended to this element
getContentType()	Returns content model of element
getParseTree()	Returns the root node of Content Model Parse Tree.
validateContent(Element)	Validates the content of a element node.

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, PERCENT, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_
REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_
ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, PERCENT, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

Inherited Member Summary

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Fields

ANY

`public static final byte ANY`

Element content type - Children can be any element

ASTERISK

`public static final int ASTERISK`

ContentModelParseTreeNode type - "*" node (has one children)

COMMA

`public static final int COMMA`

ContentModelParseTreeNode type - "," node (has two children)

ELEMENT

`public static final int ELEMENT`

ContentModelParseTreeNode type - 'leaf' node (has no children)

ELEMENTS

`public static final byte ELEMENTS`

Element content type - Children can be elements as per Content Model

EMPTY

`public static final byte EMPTY`
Element content type - No Children

MIXED

`public static final byte MIXED`
Element content type - Children can be PCDATA & elements as per Content Model

OR

`public static final int OR`
ContentModelParseTreeNode type - "|" node (has two children)

PLUS

`public static final int PLUS`
ContentModelParseTreeNode type - "+" node (has one children)

QMARK

`public static final int QMARK`
ContentModelParseTreeNode type - "?" node (has one children)

Methods

expectedElements(Element)

`public java.util.Vector expectedElements(org.w3c.dom.Element e)`
Returns vector of element names that can be appended to the element.

Parameters

`e` - Element

Returns

Vector of names

findAttrDecl(String)

`public final AttrDecl findAttrDecl(java.lang.String name)`
Gets an attribute declaration object or null if not found

Parameters

`name` - Attribute declaration to find

Returns

The `AttrDecl` object, or null, if it was not found

getAttrDecls()

`public` org.w3c.dom.NamedNodeMap `getAttrDecls()`
Gets an enumeration of attribute declarations

Returns

An enumeration of attribute declarations

getContentElements()

`public final java.util.Vector getContentElements()`
Returns Vector of elements that can be appended to this element

Returns

The `Vector` containing the element names.

getContentType()

`public int getContentType()`
Returns content model of element

Returns

The `type` of the element declaration.

getParseTree()

`public final` org.w3c.dom.Node `getParseTree()`
Returns the root node of Content Model Parse Tree. `Node.getFirstChild()` and `Node.getLastChild()` return the the parse tree branches.
`Node.getNodeType()` and `Node.getNodeName()` return the the parse tree node type and name.

Returns

The `Node` containing the Content Model parse tree root node.

validateContent(Element)

`public boolean validateContent(org.w3c.dom.Element e)`
Validates the content of a element node.

Returns

True if valid, else false

NodeFactory

Syntax

```
public class NodeFactory extends java.lang.Object implements
java.io.Serializable

java.lang.Object
  |
  +--oracle.xml.parser.v2.NodeFactory
```

All Implemented Interfaces

`java.io.Serializable`

Description

This class specifies methods to create various nodes of the DOM tree built during parsing. Applications can override these methods to create their own custom classes to be added to the DOM tree while parsing. Applications have to register their own NodeFactory using the XMLParser's setNodeFactory() method. If a null pointer is returned by these methods, then the node will not be added to the DOM tree.

See Also

`setNodeFactory(NodeFactory)`

Member Summary

Constructors	
NodeFactory()	
Methods	
createAttribute(String, String)	Creates an attribute node with the specified tag, and text.
createCDATASection(String)	Creates a CDATA node with the specified text.

Member Summary	
createComment(String)	Creates a comment node with the specified text.
createDocument()	Creates a document node.
createElement(String)	Creates an Element node with the specified tag.
createProcessingInstruction(String, String)	Creates a PI node with the specified tag, and text.
createTextNode(String)	Creates a text node with the specified text.

Inherited Member Summary
Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor

NodeFactory()

```
public  NodeFactory()
```

Methods

createAttribute(String, String)

```
public XMLAttr createAttribute(java.lang.String tag, java.lang.String text)
```

Creates an attribute node with the specified tag, and text.

Parameters

`tag` - The name of the node.

`text` - The text associated with the node.

Returns

The created attribute node.

createCDATASection(String)

```
public XMLCDATA createCDATASection(java.lang.String text)
```

Creates a CDATA node with the specified text.

Parameters

`text` - The text associated with the node.

Returns

The created CDATA node.

createComment(String)

`public XMLComment createComment(java.lang.String text)`
Creates a comment node with the specified text.

Parameters

`text` - The text associated with the node.

Returns

The created comment node.

createDocument()

`public XMLDocument createDocument()`
Creates a document node. This method cannot return a null pointer.

Returns

The created element.

createElement(String)

`public XMLElement createElement(java.lang.String tag)`
Creates an Element node with the specified tag.

Parameters

`tag` - The name of the element.

Returns

The created element.

createProcessingInstruction(String, String)

`public XMLPI createProcessingInstruction(java.lang.String tag, java.lang.String text)`

Creates a PI node with the specified tag, and text.

Parameters

`tag` - The name of the node.

`text` - The text associated with the node.

Returns

The created PI node.

createTextNode(String)

`public XMLText createTextNode(java.lang.String text)`

Creates a text node with the specified text.

Parameters

`text` - The text associated with the node.

Returns

The created text node.

NSName

Syntax

`public interface NSName`

All Known Implementing Classes:

`XMLAttr`, `XMLElement`

Description

This interface provides Namespace support for Element and Attr names

Member Summary

Methods

Member Summary	
getExpandedName()	Get the fully resolved name for this name
getLocalName()	Get the local name for this name
getNamespace()	Get the resolved Namespace for this name
getPrefix()	Get the prefix for this name
getQualifiedName()	Get the qualified name

Methods

getExpandedName()

```
public java.lang.String getExpandedName()
```

Get the fully resolved name for this name

Returns

The fully resolved name

getLocalName()

```
public java.lang.String getLocalName()
```

Get the local name for this name

Returns

The local name

getNamespace()

```
public java.lang.String getNamespace()
```

Get the resolved Namespace for this name

Returns

The resolved Namespace

getPrefix()

```
public java.lang.String getPrefix()
```

Get the prefix for this name

Returns

The prefix

getQualifiedName()

```
public java.lang.String getQualifiedName()
```

Get the qualified name

Returns

The qualified name

oracle.xml.parser.v2 NSResolver

Syntax

`public interface NSResolver`

All Known Implementing Classes

`XMLElement`

Description

This interface provides support for resolving Namespaces

Member Summary

Methods	
resolveNamespacePrefix(String)	Find the namespace definition in scope for a given namespace prefix

Methods

resolveNamespacePrefix(String)

`public java.lang.String resolveNamespacePrefix(java.lang.String prefix)`

Find the namespace definition in scope for a given namespace prefix

Parameters

`prefix` - Namespace prefix to be resolved

Returns

the resolved Namespace (null, if prefix could not be resolved)

oracle.xml.parser.v2
oraxsl

Syntax

```
public class oraxsl extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.parser.v2.oraxsl
```

Description

The oraxsl class provides a command-line interface to applying stylesheets on multiple XML documents. It accepts a number of command-line options that dictate how it should behave. The following is its invocation syntax:

```
java oraxsl options* source? stylesheet? result?
     -w                          Show warnings
     -e <error log>              A file to write errors to
     -l <xml file list>          List of files to transform
     -d <directory>              Directory with files to transform
     -x <source extension>       Extensions to exclude
     -i <source extension>       Extensions to include
     -s <stylesheet>             Stylesheet to use
     -r <result extension>       Extension to use for results
     -o <result extension>       Directory to place results
     -p <param list>             List of Params
     -t <# of threads>           Number of threads to use
     -v                                Verbose mode
```

Member Summary

Constructors

oraxsl()	

Methods

main(String[])	Invokes the oraxsl driver

Inherited Member Summary

Methods inherited from class java.lang.Object

Inherited Member Summary

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor

oraxsl()

```
public  oraxsl()
```

Methods

main(String[])

```
public static void main(java.lang.String[] args)
```

Invokes the oraxsl driver

Parameters

`args` - command line arguments

oracle.xml.parser.v2
SAXAttrList

Syntax

`public class SAXAttrList extends java.lang.Object implements` org.xml.sax.AttributeList

```
java.lang.Object
  |
  +--oracle.xml.parser.v2.SAXAttrList
```

All Implemented Interfaces

org.xml.sax.AttributeList

Description

This class implements the SAX `AttributeList` interface and also provides Namespace support. Applications that require Namespace support can explicitly cast any attribute list returned by an Oracle parser class to `SAXAttrList` and use the methods described here.

Member Summary	
Methods	
getExpandedName(int)	Get the expanded name for an attribute in the list (by position)
getLength()	Return the number of attributes in this list.
getLocalName(int)	Get the local name for an attribute in the list (by position)
getName(int)	Return the name of an attribute in this list (by position).
getNamespace(int)	Get the resolved namespace for an attribute in the list (by position)
getPrefix(int)	Get the namespace prefix for an attribute in the list (by position)
getQualifiedName(int)	Get the qualified name for an attribute in the list (by position)
getType(int)	
getType(String)	Return the type of an attribute in the list (by name).
getValue(int)	Return the value of an attribute in the list (by position).
getValue(String)	Return the value of an attribute in the list (by name).

Inherited Member Summary

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods

getExpandedName(int)

`public java.lang.String getExpandedName(int i)`
Get the expanded name for an attribute in the list (by position)

Parameters

`i` - The index of the attribute in the list.

Returns

The expanded name for the attribute

getLength()

`public int getLength()`
Return the number of attributes in this list.

The SAX parser may provide attributes in any arbitrary order, regardless of the order in which they were declared or specified. The number of attributes may be zero.

Specified By

org.xml.sax.AttributeList.getLength() in interface org.xml.sax.AttributeList

Returns

The number of attributes in the list.

getLocalName(int)

`public java.lang.String getLocalName(int i)`
Get the local name for an attribute in the list (by position)

Parameters

`i` - The index of the attribute in the list.

Returns

The local name for the attribute

getName(int)

`public java.lang.String getName(int i)`
Return the name of an attribute in this list (by position).

The names must be unique the SAX parser shall not include the same attribute twice. Attributes without values (those declared #IMPLIED without a value specified in the start tag) will be omitted from the list.

If the attribute name has a namespace prefix, the prefix will still be attached.

Specified By

org.xml.sax.AttributeList.getName(int) in interface org.xml.sax.AttributeList

Parameters

`i` - The index of the attribute in the list (starting at 0).

Returns

The name of the indexed attribute, or null if the index is out of range.

See Also

getLength()

getNamespace(int)

`public java.lang.String getNamespace(int i)`
Get the resolved namespace for an attribute in the list (by position)

Parameters

`i` - The index of the attribute in the list.

Returns

The resolved namespace for the attribute

getPrefix(int)

`public java.lang.String getPrefix(int i)`
Get the namespace prefix for an attribute in the list (by position)

Parameters

`i` - The index of the attribute in the list.

Returns

The namespace prefix for the attribute

getQualifiedName(int)

`public java.lang.String getQualifiedName(int i)`
Get the qualified name for an attribute in the list (by position)

Parameters

`i` - The index of the attribute in the list.

Returns

The qualified name for the attribute

getType(int)

`public java.lang.String getType(int i)`

Specified By

org.xml.sax.AttributeList.getType(int) in interface org.xml.sax.AttributeList

getType(String)

`public java.lang.String getType(java.lang.String s)`
Return the type of an attribute in the list (by name).

The return value is the same as the return value for getType(int).

If the attribute name has a namespace prefix in the document, the application must include the prefix here.

Specified By

org.xml.sax.AttributeList.getType(String) in interface org.xml.sax.AttributeList

Parameters

`name` - The name of the attribute.

Returns

The attribute type as a string, or null if no such attribute exists.

See Also

getType(int)

getValue(int)

`public java.lang.String getValue(int i)`
Return the value of an attribute in the list (by position).

If the attribute value is a list of tokens (IDREFS, ENTITIES, or NMTOKENS), the tokens will be concatenated into a single string separated by whitespace.

Specified By

org.xml.sax.AttributeList.getValue(int) in interface org.xml.sax.AttributeList

Parameters

`i` - The index of the attribute in the list (starting at 0).

Returns

The attribute value as a string, or null if the index is out of range.

See Also

getLength(), getValue(String)

getValue(String)

`public java.lang.String getValue(java.lang.String s)`
Return the value of an attribute in the list (by name).

The return value is the same as the return value for getValue(int).

If the attribute name has a namespace prefix in the document, the application must include the prefix here.

Specified By
org.xml.sax.AttributeList.getValue(String) in interface org.xml.sax.AttributeList

Parameters
`i` - The index of the attribute in the list.

Returns
The attribute value as a string, or null if no such attribute exists.

See Also
getValue(int)

oracle.xml.parser.v2
SAXParser

Syntax

`public class SAXParser extends XMLParser implements` org.xml.sax.Parser, `oracle.xml.parser.v2.XMLConstants`

```
java.lang.Object
  |
  +--XMLParser
        |
        +--oracle.xml.parser.v2.SAXParser
```

All Implemented Interfaces

org.xml.sax.Parser, `oracle.xml.parser.v2.XMLConstants`

Description

This class implements an eXtensible Markup Language (XML) 1.0 SAX parser according to the World Wide Web Consortium (W3C) recommendation. Applications can register a SAX handler to receive notification of various parser events.

Member Summary	
Constructors	
SAXParser()	Creates a new parser object.
Methods	
setDocumentHandler(DocumentHandler)	SAX applications can use this to register a new document event handler.
setDTDHandler(DTDHandler)	SAX applications can use this to register a new DTD event handler.
setEntityResolver(EntityResolver)	SAX applications can use this to register a new entity resolver
setErrorHandler(ErrorHandler)	SAX applications can use this to register a new error event handler.

Inherited Member Summary

Fields inherited from class XMLParser

Inherited Member Summary

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLParser

getReleaseVersion(), getValidationMode(), parse(InputSource), parse(InputStream), parse(Reader), parse(String), parse(URL), setBaseURL(URL), setDoctype(DTD), setLocale(Locale), setPreserveWhitespace(boolean), setValidationMode(boolean)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Parser

parse(InputSource), parse(String), setLocale(Locale)

Constructor

SAXParser()

```
public  SAXParser()
```

Creates a new parser object.

Methods

setDocumentHandler(DocumentHandler)

```
public void setDocumentHandler(org.xml.sax.DocumentHandler handler)
```

SAX applications can use this to register a new document event handler.

Specified By

org.xml.sax.Parser.setDocumentHandler(DocumentHandler) in interface org.xml.sax.Parser

Parameters

`handler` - `DocumentHandler` being registered

See Also

org.xml.sax.Parser.setDocumentHandler(DocumentHandler), org.xml.sax.DocumentHandler

setDTDHandler(DTDHandler)

`public void setDTDHandler(`org.xml.sax.DTDHandler `handler)`

SAX applications can use this to register a new DTD event handler.

Specified By

org.xml.sax.Parser.setDTDHandler(DTDHandler) in interface org.xml.sax.Parser

Parameters

`handler` - `DTDHandler` being registered

See Also

org.xml.sax.Parser.setDTDHandler(DTDHandler), org.xml.sax.DTDHandler

setEntityResolver(EntityResolver)

`public void setEntityResolver(`org.xml.sax.EntityResolver `resolver)`

SAX applications can use this to register a new entity resolver

Specified By

org.xml.sax.Parser.setEntityResolver(EntityResolver) in interface org.xml.sax.Parser

Parameters

`resolver` - `EntityResolver` being registered

See Also

org.xml.sax.Parser.setEntityResolver(EntityResolver), org.xml.sax.DTDHandler

setErrorHandler(ErrorHandler)

`public void setErrorHandler(`org.xml.sax.ErrorHandler `handler)`
SAX applications can use this to register a new error event handler. This replaces any previous setting for error handling.

Specified By

org.xml.sax.Parser.setErrorHandler(ErrorHandler) in interface org.xml.sax.Parser

Parameters

`handler` - `ErrorHandler` being registered

See Also

org.xml.sax.Parser.setErrorHandler(ErrorHandler), org.xml.sax.ErrorHandler

oracle.xml.parser.v2
XMLAttr

Syntax

public class XMLAttr extends XMLNode implements org.w3c.dom.Attr, NSName, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.XMLAttr
```

All Implemented Interfaces

org.w3c.dom.Attr, java.lang.Cloneable, org.w3c.dom.Node, NSName, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM Attr interface and holds information on each attribute of an element.

See Also

org.w3c.dom.Attr, NodeFactory, setNodeFactory(NodeFactory)

Member Summary

Constructors	
XMLAttr(String, String)	Construct attribute with given name and value.
XMLAttr(String, String, String, String)	Namespace support
Methods	
cloneNode(boolean)	Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes.
getExpandedName()	Get the fully resolved Name for this attribute
getLocalName()	Get the local Name for this attribute
getName()	Gets the attribute name.

Member Summary	
getNamespace()	Get the resolved Namespace for this attribute
getNodeValue()	Gets the value of this node, depending on its type
getParentNode()	Gets the parent of this node.
getPrefix()	Get the namespace prefix for this attribute
getQualifiedName()	Gets the qualified name for this attribute
getSpecified()	Returns true if the attribute was specified explicity in the element
getValue()	Gets the attribute value.
setNodeValue(String)	Sets the value of this node, depending on its type
setValue(String)	Sets the value.

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

Inherited Member Summary

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

appendChild(Node), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getOwnerDocument(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getOwnerDocument(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node)

Constructor

XMLAttr(String, String)

```
public  XMLAttr(java.lang.String n, java.lang.String v)
```

Construct attribute with given name and value.

Parameters

`n` - Name of the attribute

`v` - Value of the attribute

XMLAttr(String, String, String, String)

```
public  XMLAttr(java.lang.String name, java.lang.String prefix, java.lang.String
ns, java.lang.String v)
```

Namespace support

Methods

cloneNode(boolean)

`public org.w3c.dom.Node cloneNode(boolean deep)`

Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `null`.). Cloning an `Element` copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child `Text` node. Cloning any other type of node simply returns a copy of this node.

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Overrides

cloneNode(boolean) in class XMLNode

Parameters

`deep` - If `true`, recursively clone the subtree under the specified node; if `false`, clone only the node itself (and its attributes, if it is an `Element`).

Returns

The duplicate node.

getExpandedName()

`public java.lang.String getExpandedName()`

Get the fully resolved Name for this attribute

Specified By

getExpandedName() in interface NSName

Returns

the fully resolved Name

getLocalName()

`public java.lang.String getLocalName()`

Get the local Name for this attribute

Specified By

getLocalName() in interface NSName

Returns

the local Name

getName()

`public java.lang.String getName()`
Gets the attribute name.

Specified By

org.w3c.dom.Attr.getName() in interface org.w3c.dom.Attr

Returns

attribute name

getNamespace()

`public java.lang.String getNamespace()`
Get the resolved Namespace for this attribute

Specified By

getNamespace() in interface NSName

Returns

the resolved Namespace

getNodeValue()

`public java.lang.String getNodeValue()`
Gets the value of this node, depending on its type

Specified By

org.w3c.dom.Node.getNodeValue() in interface org.w3c.dom.Node

Overrides

getNodeValue() in class XMLNode

Returns

Value of this node

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised when the node is readonly. DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

getParentNode()

`public` org.w3c.dom.Node `getParentNode()`
Gets the parent of this node. All nodes, except `Document`, `DocumentFragment`, and `Attr` may have a parent. However, if a node has just been created and not yet added to the tree, or if it has been removed from the tree, this is `null`.

Specified By

org.w3c.dom.Node.getParentNode() in interface org.w3c.dom.Node

Overrides

getParentNode() in class XMLNode

Returns

The parent of this node

getPrefix()

`public java.lang.String getPrefix()`
Get the namespace prefix for this attribute

Specified By

getPrefix() in interface NSName

Returns

the namespace prefix

getQualifiedName()

`public java.lang.String getQualifiedName()`
Gets the qualified name for this attribute

Specified By

getQualifiedName() in interface NSName

Returns

the qualified name

getSpecified()

`public boolean getSpecified()`
Returns true if the attribute was specified explicity in the element

Specified By

org.w3c.dom.Attr.getSpecified() in interface org.w3c.dom.Attr

Returns

`true`, if the attribute was specified explicitly, `false`, if it was not

getValue()

`public java.lang.String getValue()`
Gets the attribute value.

Specified By

org.w3c.dom.Attr.getValue() in interface org.w3c.dom.Attr

Returns

attribute value

setNodeValue(String)

`public void setNodeValue(java.lang.String nodeValue)`
Sets the value of this node, depending on its type

Specified By

org.w3c.dom.Node.setNodeValue(String) in interface org.w3c.dom.Node

Overrides

setNodeValue(String) in class XMLNode

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised when the node is readonly. DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

setValue(String)

`public void setValue(java.lang.String arg)`
Sets the value.

Specified By

org.w3c.dom.Attr.setValue(String) in interface org.w3c.dom.Attr

Parameters

`arg` - Value to set

oracle.xml.parser.v2
XMLCDATA

Syntax

`public class XMLCDATA extends XMLText implements` org.w3c.dom.CDATASection, `java.io.Serializable`

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.CharData
              |
              +--XMLText
                    |
                    +--oracle.xml.parser.v2.XMLCDATA
```

All Implemented Interfaces

org.w3c.dom.CDATASection, org.w3c.dom.CharacterData, `java.lang.Cloneable`, org.w3c.dom.Node, `java.io.Serializable`, org.w3c.dom.Text, `oracle.xml.parser.v2.XMLConstants`

Description

This class implements the DOM CDATASection interface.

See Also

org.w3c.dom.CDATASection, `NodeFactory`, `setNodeFactory(NodeFactory)`

Member Summary

Constructors	
XMLCDATA(String)	Creates a CDATA node having the given name and text.

Inherited Member Summary

Fields inherited from class XMLNode

Inherited Member Summary

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_
REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLText

appendData, deleteData, getData, getLength, getNodeValue(), insertData, replaceData, setData, setNodeValue, splitText(int), substringData

Methods inherited from class oracle.xml.parser.v2.CharData

appendData, deleteData, getData, getLength, insertData, replaceData, setData, setNodeValue, substringData

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Text

Inherited Member Summary

splitText(int)

Methods inherited from interface CharacterData

appendData(String), deleteData(int, int), getData(), getLength(), insertData(int, String), replaceData(int, int, String), setData(String), substringData(int, int)

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Constructor

XMLCDATA(String)

`public  XMLCDATA(java.lang.String text)`

Creates a CDATA node having the given name and text.

Parameters

`text` - Text of the node

oracle.xml.parser.v2
XMLComment

Syntax

public class XMLComment extends oracle.xml.parser.v2.CharData implements org.w3c.dom.Comment, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.CharData
              |
              +--oracle.xml.parser.v2.XMLComment
```

All Implemented Interfaces

org.w3c.dom.CharacterData, java.lang.Cloneable, org.w3c.dom.Comment, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM Comment interface.

See Also

org.w3c.dom.Comment, NodeFactory, setNodeFactory(NodeFactory)

Member Summary

Constructors	
XMLComment(String)	Creates a new Comment node.

Inherited Member Summary

Fields inherited from class XMLNode

Inherited Member Summary

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class oracle.xml.parser.v2.CharData

appendData, deleteData, getData, getLength, insertData, replaceData, setData, setNodeValue, substringData

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface CharacterData

appendData(String), deleteData(int, int), getData(), getLength(), insertData(int, String), replaceData(int, int, String), setData(String), substringData(int, int)

Methods inherited from interface Node

Inherited Member Summary

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Constructor

XMLComment(String)

```
public  XMLComment(java.lang.String text)
```

Creates a new Comment node.

Parameters

`text` - Text of the comment node

oracle.xml.parser.v2 **XMLDocument**

Syntax

`public class XMLDocument extends XMLNode implements org.w3c.dom.Document, java.io.Serializable`

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.XMLDocument
```

All Implemented Interfaces

`java.lang.Cloneable`, `org.w3c.dom.Document`, `org.w3c.dom.Node`, `java.io.Serializable`, `oracle.xml.parser.v2.XMLConstants`

Description

This class implements the DOM Document interface, represents an entire XML document and serves the root of the Document Object Model tree. Each XML tag can either represent a node or a leaf of this tree.

According to the XML specification, the root of the tree consists of any combination of comments and processing instructions, but only one root element. A helper method `getDocumentElement` is provided as a short cut to finding the root element.

Member Summary	
Constructors	
XMLDocument()	Creates an empty document.
Methods	
cloneNode(boolean)	Returns a duplicate of this document node.
createAttribute(String)	Creates an `Attr` of the given name.
createCDATASection(String)	Creates a `CDATASection` node whose value is the specified string.
createComment(String)	Creates a `Comment` node given the specified string.

Member Summary	
createDocumentFragment()	Creates an empty `DocumentFragment` object.
createElement(String)	Creates an element of the type specified.
createEntityReference(String)	Creates an EntityReference object.
createProcessingInstruction(String, String)	Creates a `ProcessingInstruction` node given the specified name and data strings.
createTextNode(String)	Creates a `Text` node given the specified string.
expectedElements(Element)	Returns vector of element names that can be appended to the element.
getDoctype()	The Document Type Declaration (DTD) associated with this document.
getDocumentElement()	This is a convenience attribute that allows direct access to the child node that is the root element of the document.
getElementsByTagName(String)	Returns a `NodeList` of all the `Element`s with a given tag name in the order in which they would be encountered in a preorder traversal of the `Document` tree.
getEncoding()	Retrieves the character encoding information.
getImplementation()	The `DOMImplementation` object that handles this document.
getOwnerDocument()	The `Document` object associated with this node.
getStandalone()	Retrieves the standalone information.
getVersion()	Retrieves the version information.
print(OutputStream)	Writes the contents of this document to the given output stream.
print(OutputStream, String)	Writes the contents of this document to the given output stream.
print(PrintWriter)	Writes the contents of this document to the given output stream.
printExternalDTD(OutputStream)	Writes the contents of this document to the given output stream.
printExternalDTD(OutputStream, String)	Writes the contents of the external DTD to the given output stream.
printExternalDTD(PrintWriter)	Writes the contents of this document to the given output stream.
replaceChild(Node, Node)	Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node.
setEncoding(String)	Sets the character encoding for output.
setLocale(Locale)	Sets the locale for error reporting
setStandalone(String)	Sets the standalone information stored in the <?xml ...?> tag.

Member Summary	
setVersion(String)	Sets the version number stored in the <?xml ...?> tag.
validateElementContent(Element)	Validates the content of a element node.

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

appendChild(Node), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

Inherited Member Summary

appendChild(Node), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), `getNextSibling()`, getNodeName(), getNodeType(), getNodeValue(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), setNodeValue(String)

Constructor

XMLDocument()

`public XMLDocument()`
Creates an empty document.

Methods

cloneNode(boolean)

`public` org.w3c.dom.Node `cloneNode(boolean deep)`
Returns a duplicate of this document node.

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Overrides

cloneNode(boolean) in class XMLNode

Parameters

`deep` - If `true`, recursively clone the subtree under the document; if `false`, clone only the document itself

Returns

The duplicate document node.

createAttribute(String)

`public` org.w3c.dom.Attr `createAttribute(java.lang.String name)`
Creates an `Attr` of the given name. Note that the `Attr` instance can then be set on an `Element` using the `setAttribute` method.

Specified By

org.w3c.dom.Document.createAttribute(String) in interface org.w3c.dom.Document

Parameters

`name` - The name of the attribute.

Returns

A new `Attr` object.

Throws

org.w3c.dom.DOMException - INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.

createCDATASection(String)

`public` org.w3c.dom.CDATASection `createCDATASection(java.lang.String data)`
Creates a `CDATASection` node whose value is the specified string.

Specified By

org.w3c.dom.Document.createCDATASection(String) in interface org.w3c.dom.Document

Parameters

`data` - The data for the `CDATASection` contents.

Returns

The new `CDATASection` object.

Throws

org.w3c.dom.DOMException - A `DOMException` could be thrown.

createComment(String)

`public` org.w3c.dom.Comment `createComment(java.lang.String data)`
Creates a `Comment` node given the specified string.

Specified By

org.w3c.dom.Document.createComment(String) in interface org.w3c.dom.Document

Parameters

`data` - The data for the node.

Returns

The new `Comment` object.

createDocumentFragment()

`public` org.w3c.dom.DocumentFragment `createDocumentFragment()`
Creates an empty `DocumentFragment` object.

Specified By

org.w3c.dom.Document.createDocumentFragment() in interface org.w3c.dom.Document

Returns

A new `DocumentFragment`.

createElement(String)

`public` org.w3c.dom.Element `createElement(java.lang.String tagName)`
Creates an element of the type specified. Note that the instance returned implements the Element interface, so attributes can be specified directly on the returned object.

Specified By

org.w3c.dom.Document.createElement(String) in interface org.w3c.dom.Document

Parameters

`tagName` - The name of the element type to instantiate. The name is treated as case-sensitive.

Returns

A new `Element` object.

Throws

org.w3c.dom.DOMException - INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.

createEntityReference(String)

`public` org.w3c.dom.EntityReference `createEntityReference(java.lang.String name)`
Creates an EntityReference object.

Specified By

org.w3c.dom.Document.createEntityReference(String) in interface org.w3c.dom.Document

Parameters

`name` - The name of the entity to reference.

Returns

The new `EntityReference` object.

Throws

org.w3c.dom.DOMException - INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.

createProcessingInstruction(String, String)

`public` org.w3c.dom.ProcessingInstruction
`createProcessingInstruction(java.lang.String target, java.lang.String data)`
Creates a `ProcessingInstruction` node given the specified name and data strings.

Specified By

org.w3c.dom.Document.createProcessingInstruction(String, String) in interface org.w3c.dom.Document

Parameters

`target` - The target part of the processing instruction.

`data` - The data for the node.

Returns

The new `ProcessingInstruction` object.

Throws

org.w3c.dom.DOMException - INVALID_CHARACTER_ERR: Raised if an invalid character is specified.

createTextNode(String)

`public org.w3c.dom.Text createTextNode(java.lang.String data)`
Creates a `Text` node given the specified string.

Specified By

org.w3c.dom.Document.createTextNode(String) in interface org.w3c.dom.Document

Parameters

`data` - The data for the node.

Returns

The new `Text` object.

expectedElements(Element)

`public java.util.Vector expectedElements(org.w3c.dom.Element e)`
Returns vector of element names that can be appended to the element.

Parameters

`e` - Element

Returns

Vector of names

getDoctype()

`public org.w3c.dom.DocumentType getDoctype()`
The Document Type Declaration (DTD) associated with this document. For XML documents without a DTD, this returns `null`. Note that the DOM Level 1 specification does not support editing the DTD.

Specified By

org.w3c.dom.Document.getDoctype() in interface org.w3c.dom.Document

Returns

The associated DTD

See Also

org.w3c.dom.DocumentType

getDocumentElement()

`public org.w3c.dom.Element getDocumentElement()`
This is a convenience attribute that allows direct access to the child node that is the root element of the document.

Specified By

org.w3c.dom.Document.getDocumentElement() in interface org.w3c.dom.Document

Returns

The root element

getElementsByTagName(String)

`public org.w3c.dom.NodeList getElementsByTagName(java.lang.String tagname)`
Returns a `NodeList` of all the `Element`s with a given tag name in the order in which they would be encountered in a preorder traversal of the `Document` tree.

Specified By

org.w3c.dom.Document.getElementsByTagName(String) in interface org.w3c.dom.Document

Parameters

`tagname` - The name of the tag to match on. The special value "*" matches all tags.

Returns

A new `NodeList` object containing all the matched `Element`s.

getEncoding()

`public final java.lang.String getEncoding()`
Retrieves the character encoding information.

Returns

the encoding information stored in the <?xml ...?> tag or the user-defined output encoding if it has been more recently set.

getImplementation()

`public` org.w3c.dom.DOMImplementation `getImplementation()`
The `DOMImplementation` object that handles this document. A DOM application may use objects from multiple implementations.

Specified By

org.w3c.dom.Document.getImplementation() in interface org.w3c.dom.Document

Returns

The associated DOM implementation.

getOwnerDocument()

`public` org.w3c.dom.Document `getOwnerDocument()`
The `Document` object associated with this node. Since this node is a `Document` this is `null`.

Specified By

org.w3c.dom.Node.getOwnerDocument() in interface org.w3c.dom.Node

Overrides

getOwnerDocument() in class XMLNode

Returns

`null`

getStandalone()

`public final java.lang.String getStandalone()`
Retrieves the standalone information.

Returns

the standalone attribute stored in the <?xml ...?> tag.

getVersion()

`public final java.lang.String getVersion()`
Retrieves the version information.

Returns

the version number stored in the <?xml ...?> tag.

print(OutputStream)

`public void print(java.io.OutputStream out)`
Writes the contents of this document to the given output stream.

Overrides

print(OutputStream) in class XMLNode

Parameters

`out` - `OutputStream` to write to

Throws

`IOException` - if an error occurs

print(OutputStream, String)

`public void print(java.io.OutputStream out, java.lang.String enc)`
Writes the contents of this document to the given output stream.

Overrides

print(OutputStream, String) in class XMLNode

Parameters

`out` - `OutputStream` to write to

`enc` - Encoding to use for the output

Throws

`IOException` - if an invalid encoding was specified or if any other error occurs

print(PrintWriter)

`public void print(java.io.PrintWriter out)`
Writes the contents of this document to the given output stream.

Overrides

print(PrintWriter) in class XMLNode

Parameters
`out` - `PrintWriter` to write to

Throws
`IOException` - if an error occurs

printExternalDTD(OutputStream)
`public void printExternalDTD(java.io.OutputStream out)`
Writes the contents of this document to the given output stream.

Parameters
`out` - `OutputStream` to write to

Throws
`IOException` - if an error occurs

printExternalDTD(OutputStream, String)
`public void printExternalDTD(java.io.OutputStream out, java.lang.String enc)`
Writes the contents of the external DTD to the given output stream.

Parameters
`out` - `OutputStream` to write to

`enc` - Encoding to use for the output

Throws
`IOException` - if an invalid encoding was specified or if any other error occurs

printExternalDTD(PrintWriter)
`public void printExternalDTD(java.io.PrintWriter out)`
Writes the contents of this document to the given output stream.

Parameters
`out` - `PrintWriter` to write to

Throws
`IOException` - if an error occurs

replaceChild(Node, Node)

`public org.w3c.dom.Node replaceChild(org.w3c.dom.Node newChild, org.w3c.dom.Node oldChild)`

Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node. If the `newChild` is already in the tree, it is first removed. This is an override of the `XMLNode.removeChild` method

Specified By

org.w3c.dom.Node.replaceChild(Node, Node) in interface org.w3c.dom.Node

Overrides

replaceChild(Node, Node) in class XMLNode

Parameters

`newChild` - The new node to put in the child list.

`oldChild` - The node being replaced in the list.

Returns

The node replaced.

Throws

org.w3c.dom.DOMException - HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node. WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than this one. NOT_FOUND_ERR: Raised if `oldChild` is not a child of this node.

setEncoding(String)

`public final void setEncoding(java.lang.String encoding)`

Sets the character encoding for output. Eventually it sets the ENCODING stored in the <?xml ...?> tag, but not until the document is saved. You should not call this method until the Document has been loaded.

Parameters

`encoding` - The character encoding to set

setLocale(Locale)

`public final void setLocale(java.util.Locale locale)`

Sets the locale for error reporting

Parameters

`locale` - Locale for error reporting.

setStandalone(String)

`public final void setStandalone(java.lang.String value)`
Sets the standalone information stored in the <?xml ...?> tag.

Parameters

`value` - The attribute value ('yes' or 'no').

setVersion(String)

`public final void setVersion(java.lang.String version)`
Sets the version number stored in the <?xml ...?> tag.

Parameters

`version` - The version information to set.

validateElementContent(Element)

`public boolean validateElementContent(org.w3c.dom.Element e)`
Validates the content of a element node.

Parameters

`e` - Element to be validated

Returns

True if valid, else false

oracle.xml.parser.v2
XMLDocumentFragment

Syntax

public class XMLDocumentFragment extends XMLNode implements org.w3c.dom.DocumentFragment, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.XMLDocumentFragment
```

All Implemented Interfaces

java.lang.Cloneable, org.w3c.dom.DocumentFragment, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM DocumentFragment interface.

See Also

org.w3c.dom.DocumentFragment, NodeFactory, setNodeFactory(NodeFactory)

Member Summary

Constructors	
XMLDocumentFragment()	Creates an empty document fragment
Methods	
getParentNode()	Gets the parent of this node

Inherited Member Summary

Fields inherited from class XMLNode

Inherited Member Summary

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Constructor

XMLDocumentFragment()

`public XMLDocumentFragment()`
Creates an empty document fragment

Methods

getParentNode()

`public` org.w3c.dom.Node `getParentNode()`
Gets the parent of this node

Specified By

org.w3c.dom.Node.getParentNode() in interface org.w3c.dom.Node

Overrides

getParentNode() in class XMLNode

Returns

The parent of this node (always null)

oracle.xml.parser.v2
XMLDocumentHandler

Syntax

`public interface XMLDocumentHandler extends` org.xml.sax.DocumentHandler

All Superinterfaces

org.xml.sax.DocumentHandler

All Known Implementing Classes

`oracle.xml.parser.v2`

Description

This interface extends the `org.xml.sax.DocumentHandler` interface. SAX Applications requiring Namespace support must implement this interface and register with the SAX Parser via `Parser.setDocumentHandler()`.

Member Summary

Methods	
cDATASection(char[ ], int, int)	Receive notification of a CDATA Section.
comment(String)	Receive notification of a comment.
endDoctype()	Receive notification of end of the DTD.
endElement(NSName)	Receive notification of the end of an element.
setDoctype(DTD)	Receive notification of a DTD (Document Type node).
setTextDecl(String, String)	Receive notification of a Text XML Declaration.
setXMLDecl(String, String, String)	Receive notification of a XML Declaration.
startElement(NSName, SAXAttrList)	Receive notification of the beginning of an element.

Inherited Member Summary

Methods inherited from interface DocumentHandler

Inherited Member Summary

characters(char[], int, int), endDocument(), endElement(String), ignorableWhitespace(char[], int, int), processingInstruction(String, String), setDocumentLocator(Locator), startDocument(), startElement(String, AttributeList)

Methods

cDATASection(char[ ], int, int)

`public void cDATASection(char[] ch, int start, int length)`
Receive notification of a CDATA Section.

The Parser will invoke this method once for each CDATA Section found.

Parameters

`ch` - The CDATA section characters.

`start` - The start position in the character array.

`length` - The number of characters to use from the character array.

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

comment(String)

`public void comment(java.lang.String data)`
Receive notification of a comment.

The Parser will invoke this method once for each comment found note that comment may occur before or after the main document element.

Parameters

`data` - The comment data, or null if none was supplied.

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

endDoctype()

`public void endDoctype()`
Receive notification of end of the DTD.

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

endElement(NSName)

`public void endElement(NSName elem)`
Receive notification of the end of an element. The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding startElement() event for every endElement() event (even when the element is empty).

By implementing this method instead of `org.xml.sax.DocumentHandler.endElement`, SAX Applications can get the Namespace support provided by `NSName`.

Parameters

`elem` - NSName object

Throws

org.xml.sax.SAXException - A `SAXException` could be thrown.

See Also: org.xml.sax.DocumentHandler.endElement(String)

setDoctype(DTD)

`public void setDoctype(DTD dtd)`
Receive notification of a DTD (Document Type node).

The Parser will invoke this method after calling startDocument to register the DTD used.

Parameters

`DTD` - The DTD node

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

setTextDecl(String, String)

`public void setTextDecl(java.lang.String version, java.lang.String encoding)`
Receive notification of a Text XML Declaration.

The Parser will invoke this method once for each text XML Decl

Parameters

`version` - The version number (or null, if not specified)

`encoding` - The encoding name

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

setXMLDecl(String, String, String)

```
public void setXMLDecl(java.lang.String version, java.lang.String standalone,
java.lang.String encoding)
```
Receive notification of a XML Declaration.

The Parser will invoke this method once for XML Decl

Parameters

`version` - The version number

`standalone` - The standalone value (or null, if not specifed)

`encoding` - The encoding name (or null, if not specifed)

Throws

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

startElement(NSName, SAXAttrList)

`public void startElement(NSName elem, SAXAttrList attrlist)`

Receive notification of the beginning of an element. The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding endElement() event for every startElement() event (even when the element is empty). All of the element's content will be reported, inorder, before the corresponding endElement() event.

By implementing this method instead of `org.xml.sax.DocumentHandler.startElement`, SAX Applications can get the Namespace support provided by `NSName` and `SAXAttrList`.

Parameters

`elem` - NSName object

`attrlist` - SAXAttrList for the element

Throws

org.xml.sax.SAXException - A `SAXException` could be thrown.

See Also org.xml.sax.DocumentHandler.startElement(String, AttributeList):

oracle.xml.parser.v2
XMLElement

Syntax

public class XMLElement extends XMLNode implements org.w3c.dom.Element, java.io.Serializable, NSName, NSResolver

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.XMLElement
```

All Implemented Interfaces

ava.lang.Cloneable, org.w3c.dom.Element, org.w3c.dom.Node, NSName, NSResolver, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM `Element` interface. Elements are created by the XML parser using the default NodeFactory or the user defined NodeFactory if registered using setNodeFactoty() method.

See Also: org.w3c.dom.Element, XMLParser, NodeFactory, setNodeFactory(NodeFactory):

Member Summary

Constructors	
XMLElement(String)	Creates an element with the given name
XMLElement(String, String, String)	Creates an element with the given name, prefix, and namespace
Methods	
checkNamespace(String, String)	Returns if the element belongs to the namespace specified.
cloneNode(boolean)	Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes.
getAttribute(String)	Retrieves an attribute value by name.
getAttributeNode(String)	Retrieves an `Attr` node by name.

Member Summary

getAttributes()	A `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.
getChildrenByTagName(String)	Returns a `NodeList` of all immediate children with a given tag name,
getChildrenByTagName(String, String)	Returns a `NodeList` of all immediate children with a given tag name and namespace
getElementsByTagName(String)	Returns a `NodeList` of all descendant elements with a given tag name, in the order in which they would be encountered in a preorder traversal of the `Element` tree.
getElementsByTagName(String, String)	Returns a `NodeList` of all descendant elements with a given tag name, and namespace in the order in which they would be encountered in a preorder traversal of the `Element` tree.
getExpandedName()	Get the fully resolved name for this element.
getLocalName()	Get the local Name for this element.
getNamespace()	Get the resolved Namespace for this element.
getPrefix()	Get the namespace prefix for this element.
getQualifiedName()	Get the qualified name for this element.
getTagName()	Gets the name of the element.
normalize()	Puts all `Text` nodes in the full depth of the sub-tree underneath this `Element` into a "normal" form where only markup (e.g., tags, comments, processing instructions, CDATA sections, and entity references) separates `Text` nodes, i.e., there are no adjacent `Text` nodes.
removeAttribute(String)	Removes an attribute by name.
removeAttributeNode(Attr)	Removes the specified attribute.
resolveNamespacePrefix(String)	Given a namespace prefix, find the namespace definition in scope in this element.
setAttribute(String, String)	Adds a new attribute.
setAttributeNode(Attr)	Adds a new attribute.

Inherited Member Summary

Fields inherited from class XMLNode

Inherited Member Summary

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

appendChild(Node), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling( ), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Constructor

XMLElement(String)

`public  XMLElement(java.lang.String tag)`
Creates an element with the given name

XMLElement(String, String, String)

`public  XMLElement(java.lang.String name, java.lang.String prefix, java.lang.String namespace)`
Creates an element with the given name, prefix, and namespace

Methods

checkNamespace(String, String)

`public boolean checkNamespace(java.lang.String localname, java.lang.String ns)`
Returns if the element belongs to the namespace specified.

Parameters

`ns` - Expanded namespace string

Returns

true - if the element belongs to the namespace

cloneNode(boolean)

`public` org.w3c.dom.Node `cloneNode(boolean deep)`
Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `null`.). Cloning an `Element` copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child `Text` node. Cloning any other type of node simply returns a copy of this node.

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Overrides

cloneNode(boolean) in class XMLNode

Parameters

`deep` - If `true`, recursively clone the subtree under the specified node; if `false`, clone only the node itself (and its attributes, if it is an `Element`).

Returns

The duplicate node.

getAttribute(String)

`public java.lang.String getAttribute(java.lang.String name)`
Retrieves an attribute value by name.

Specified By

org.w3c.dom.Element.getAttribute(String) in interface org.w3c.dom.Element

Parameters

`name` - The name of the attribute to retrieve.

Returns

The `Attr` value as a string, or the empty string if that attribute does not have a specified or default value.

getAttributeNode(String)

`public` org.w3c.dom.Attr `getAttributeNode(java.lang.String name)`
Retrieves an `Attr` node by name.

Specified By

org.w3c.dom.Element.getAttributeNode(String) in interface org.w3c.dom.Element

Parameters

`name` - The name of the attribute to retrieve.

Returns

The `Attr` node with the specified attribute name or `null` if there is no such attribute.

getAttributes()

`public` org.w3c.dom.NamedNodeMap `getAttributes()`
A `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.

Specified By

org.w3c.dom.Node.getAttributes() in interface org.w3c.dom.Node

Overrides

getAttributes() in class XMLNode

Returns

The list of attributes of this element

getChildrenByTagName(String)

`public` org.w3c.dom.NodeList `getChildrenByTagName(java.lang.String name)`
Returns a `NodeList` of all immediate children with a given tag name,

Parameters

`name` - The name of the tag to match on.

Returns

A list of matching children

getChildrenByTagName(String, String)

`public` org.w3c.dom.NodeList `getChildrenByTagName(java.lang.String name, java.lang.String ns)`
Returns a `NodeList` of all immediate children with a given tag name and namespace

Parameters

`name` - The name of the tag to match on. (should be local name)

`ns` - The name space

Returns

A list of matching children

getElementsByTagName(String)

`public` org.w3c.dom.NodeList `getElementsByTagName(java.lang.String name)`
Returns a `NodeList` of all descendant elements with a given tag name, in the order in which they would be encountered in a preorder traversal of the `Element` tree.

Specified By

org.w3c.dom.Element.getElementsByTagName(String) in interface org.w3c.dom.Element

Parameters

`name` - The name of the tag to match on. The special value "*" matches all tags.

Returns

A list of matching `Element` nodes.

getElementsByTagName(String, String)

`public` org.w3c.dom.NodeList `getElementsByTagName(java.lang.String name, java.lang.String ns)`
Returns a `NodeList` of all descendant elements with a given tag name, and namespace in the order in which they would be encountered in a preorder traversal of the `Element` tree.

Parameters

`name` - The name of the tag to match on. The special value "*" matches all tags. (should be local name)

`ns` - The namespace of the elements

Returns

A list of matching `Element` nodes.

getExpandedName()

`public java.lang.String getExpandedName()`
Get the fully resolved name for this element.

Specified By

getExpandedName() in interface NSName

Returns
the fully resolved name

getLocalName()

`public java.lang.String getLocalName()`
Get the local Name for this element.

Specified By
getLocalName() in interface NSName

Returns
the local Name

getNamespace()

`public java.lang.String getNamespace()`
Get the resolved Namespace for this element.

Specified By
getNamespace() in interface NSName

Returns
the resolved Namespace

getPrefix()

`public java.lang.String getPrefix()`
Get the namespace prefix for this element.

Specified By
getPrefix() in interface NSName

Returns
the namespace prefix

getQualifiedName()

`public java.lang.String getQualifiedName()`
Get the qualified name for this element.

Specified By

getQualifiedName() in interface NSName

Returns

the qualified name

getTagName()

`public java.lang.String getTagName()`
Gets the name of the element. For example, in <elementExample id="demo"> ... </elementExample> , `tagName` has the value `"elementExample"`. Note that this is case-preserving in XML, as are all of the operations of the DOM. The HTML DOM returns the `tagName` of an HTML element in the canonical uppercase form, regardless of the case in the source HTML document.

Specified By

org.w3c.dom.Element.getTagName() in interface org.w3c.dom.Element

Returns

The element name

normalize()

`public void normalize()`
Puts all `Text` nodes in the full depth of the sub-tree underneath this `Element` into a "normal" form where only markup (e.g., tags, comments, processing instructions, CDATA sections, and entity references) separates `Text` nodes, i.e., there are no adjacent `Text` nodes. This can be used to ensure that the DOM view of a document is the same as if it were saved and re-loaded, and is useful when operations (such as XPointer lookups) that depend on a particular doc1ument tree structure are to be used.

Specified By

org.w3c.dom.Element.normalize() in interface org.w3c.dom.Element

removeAttribute(String)

`public void removeAttribute(java.lang.String name)`
Removes an attribute by name. If the removed attribute has a default value it is immediately replaced.

Specified By

org.w3c.dom.Element.removeAttribute(String) in interface org.w3c.dom.Element

Parameters

`name` - The name of the attribute to remove.

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly.

removeAttributeNode(Attr)

`public org.w3c.dom.Attr removeAttributeNode(org.w3c.dom.Attr oldAttr)`
Removes the specified attribute.

Specified By

org.w3c.dom.Element.removeAttributeNode(Attr) in interface org.w3c.dom.Element

Parameters

`oldAttr` - The `Attr` node to remove from the attribute list. If the removed `Attr` has a default value it is immediately replaced.

Returns

The `Attr` node that was removed.

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly. NOT_FOUND_ERR: Raised if `oldAttr` is not an attribute of the element.

resolveNamespacePrefix(String)

`public java.lang.String resolveNamespacePrefix(java.lang.String prefix)`
Given a namespace prefix, find the namespace definition in scope in this element.

Specified By

resolveNamespacePrefix(String) in interface NSResolver

Parameters

`prefix` - Namespace prefix to be resolved

Returns

the resolved Namespace (null, if prefix could not be resolved)

setAttribute(String, String)

`public void setAttribute(java.lang.String name, java.lang.String value)`
Adds a new attribute. If an attribute with that name is already present in the element, its value is changed to be that of the value parameter. This value is a simple string, it is not parsed as it is being set. So any markup (such as syntax to be recognized as an entity reference) is treated as literal text, and needs to be appropriately escaped by the implementation when it is written out. In order to assign an attribute value that contains entity references, the user must create an `Attr` node plus any `Text` and `EntityReference` nodes, build the appropriate subtree, and use `setAttributeNode` to assign it as the value of an attribute.

Specified By

org.w3c.dom.Element.setAttribute(String, String) in interface org.w3c.dom.Element

Parameters

`name` - The name of the attribute to create or alter.

`value` - Value to set in string form.

Throws

org.w3c.dom.DOMException - INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly.

setAttributeNode(Attr)

`public` org.w3c.dom.Attr `setAttributeNode(`org.w3c.dom.Attr `newAttr)`
Adds a new attribute. If an attribute with that name is already present in the element, it is replaced by the new one.

Specified By

org.w3c.dom.Element.setAttributeNode(Attr) in interface org.w3c.dom.Element

Parameters

`newAttr` - The `Attr` node to add to the attribute list.

Returns

If the `newAttr` attribute replaces an existing attribute with the same name, the previously existing `Attr` node is returned, otherwise `null` is returned.

Throws

org.w3c.dom.DOMException - WRONG_DOCUMENT_ERR: Raised if `newAttr` was created from a different document than the one that created the element. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly. INUSE_ATTRIBUTE_ERR: Raised if `newAttr` is already an attribute of another `Element` object. The DOM user must explicitly clone `Attr` nodes to re-use them in other elements.

oracle.xml.parser.v2
XMLEntityReference

Syntax

```
public class XMLEntityReference extends XMLNode implements
org.w3c.dom.EntityReference, oracle.xml.parser.v2.XMLConstants,
java.lang.Cloneable, java.io.Serializable

java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.XMLEntityReference
```

All Implemented Interfaces

java.lang.Cloneable, org.w3c.dom.EntityReference, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

Member Summary

Constructors

XMLEntityReference(String)

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

Inherited Member Summary

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), setNodeValue(String), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Constructor

XMLEntityReference(String)

```
public  XMLEntityReference(java.lang.String tag)
```

oracle.xml.parser.v2
XMLNode

Syntax

public class XMLNode extends java.lang.Object implements org.w3c.dom.Node, oracle.xml.parser.v2.XMLConstants, java.lang.Cloneable, java.io.Serializable

```
java.lang.Object
  |
  +--oracle.xml.parser.v2.XMLNode
```

Direct Known Subclasses

AttrDecl, oracle.xml.parser.v2.CharData, DTD, ElementDecl, XMLAttr, XMLDocument, XMLDocumentFragment, XMLElement, XMLEntityReference

All Implemented Interfaces

java.lang.Cloneable, org.w3c.dom.Node, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

Implements the DOM `Node` interface and serves as the primary datatype for the entire Document Object Model. It represents a single node in the document tree.

The attributes `nodeName`, `nodeValue` and `attributes` are included as a mechanism to get at node information without casting down to the specific derived instance. In cases where there is no obvious mapping of these attributes for a specific `nodeType` (e.g., `nodeValue` for an Element or `attributes` for a Comment), this returns `null`. Note that the derived classes may contain additional and more convenient mechanisms to get and set the relevant information.

Member Summary

Fields	
ATTRDECL	A attribute declaration node
ELEMENTDECL	An element declaration node.
Methods	
appendChild(Node)	Adds the node `newChild` to the end of the list of children of this node.

Member Summary	
cloneNode(boolean)	Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes.
getAttributes()	Gets a `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.
getChildNodes()	Gets a `NodeList` that contains all children of this node.
getFirstChild()	Gets the first child of this node.
getLastChild()	Gets the last child of this node.
getNextSibling()	Gets The node immediately following this node.
getNodeName()	Gets the name of this node, depending on its type
getNodeType()	Gets a code representing the type of the underlying object
getNodeValue()	Gets the value of this node, depending on its type
getOwnerDocument()	Gets the `Document` object associated with this node.
getParentNode()	Gets the parent of this node.
getPreviousSibling()	Gets the node immediately preceding this node.
hasChildNodes()	This is a convenience method to allow easy determination of whether a node has any children.
insertBefore(Node, Node)	Inserts the node `newChild` before the existing child node `refChild`.
print(OutputStream)	Writes the contents of this node to the given output stream.
print(OutputStream, String)	Writes the contents of this node to the given output stream.
print(PrintWriter)	Writes the contents of this node using the given print writer.
removeChild(Node)	Removes the child node indicated by `oldChild` from the list of children, and returns it.
replaceChild(Node, Node)	Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node.
selectNodes(String, NSResolver)	Selects nodes from the tree which match the given pattern
selectSingleNode(String, NSResolver)	Selects the first node from the tree that matches the given pattern
setNodeValue(String)	Sets the value of this node, depending on its type
transformNode(XSLStylesheet)	Transforms a node in the tree using the given stylesheet
valueOf(String, NSResolver)	Selects the value of the first node from the tree that matches the given pattern

Inherited Member Summary

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

ATTRDECL

`public static final short ATTRDECL`

A attribute declaration node

ELEMENTDECL

`public static final short ELEMENTDECL`

An element declaration node.

Methods

appendChild(Node)

`public org.w3c.dom.Node appendChild(org.w3c.dom.Node newChild)`

Adds the node `newChild` to the end of the list of children of this node. If the `newChild` is already in the tree, it is first removed.

Specified By

org.w3c.dom.Node.appendChild(Node) in interface org.w3c.dom.Node

Parameters

`newChild` - The node to add.If it is a `DocumentFragment` object, the entire contents of the document fragment are moved into the child list of this node

Returns

The node added.

Throws

org.w3c.dom.DOMException - HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or if the node to append is one of this node's ancestors. WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly.

cloneNode(boolean)

`public` org.w3c.dom.Node `cloneNode(boolean deep)`
Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `null`.). Cloning an `Element` copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child `Text` node. Cloning any other type of node simply returns a copy of this node.

Specified By

org.w3c.dom.Node.cloneNode(boolean) in interface org.w3c.dom.Node

Parameters

`deep` - If `true`, recursively clone the subtree under the specified node; if `false`, clone only the node itself (and its attributes, if it is an `Element`).

Returns

The duplicate node.

getAttributes()

`public` org.w3c.dom.NamedNodeMap `getAttributes()`
Gets a `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.

Specified By

org.w3c.dom.Node.getAttributes() in interface org.w3c.dom.Node

Returns

the attributes of this node

getChildNodes()

`public` org.w3c.dom.NodeList `getChildNodes()`
Gets a `NodeList` that contains all children of this node. If there are no children, this is a `NodeList` containing no nodes. The content of the returned `NodeList` is "live" in the sense that, for instance, changes to the children of the node object that it was created from are immediately reflected in the nodes returned by the `NodeList` accessors; it is not a static snapshot of the content of the node. This is true for every `NodeList`, including the ones returned by the `getElementsByTagName` method.

Specified By

org.w3c.dom.Node.getChildNodes() in interface org.w3c.dom.Node

Returns

The children of this node

getFirstChild()

`public` org.w3c.dom.Node `getFirstChild()`
Gets the first child of this node. If there is no such node, this returns `null`.

Specified By

org.w3c.dom.Node.getFirstChild() in interface org.w3c.dom.Node

Returns

The first child of this node

getLastChild()

`public` org.w3c.dom.Node `getLastChild()`
Gets the last child of this node. If there is no such node, this returns `null`.

Specified By

org.w3c.dom.Node.getLastChild() in interface org.w3c.dom.Node

Returns

The last child of this node

getNextSibling()

`public` org.w3c.dom.Node `getNextSibling()`
Gets The node immediately following this node. If there is no such node, this returns `null`.

Specified By

org.w3c.dom.Node.getNextSibling() in interface org.w3c.dom.Node

Returns

the next node

getNodeName()

`public java.lang.String getNodeName()`
Gets the name of this node, depending on its type

Specified By

org.w3c.dom.Node.getNodeName() in interface org.w3c.dom.Node

Returns

Name of this node

getNodeType()

`public short getNodeType()`
Gets a code representing the type of the underlying object

Specified By

org.w3c.dom.Node.getNodeType() in interface org.w3c.dom.Node

Returns

type of the node

getNodeValue()

`public java.lang.String getNodeValue()`

Gets the value of this node, depending on its type

Specified By

org.w3c.dom.Node.getNodeValue() in interface org.w3c.dom.Node

Returns

Value of this node

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised when the node is readonly. DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

getOwnerDocument()

`public` org.w3c.dom.Document `getOwnerDocument()`

Gets the `Document` object associated with this node. This is also the `Document` object used to create new nodes. When this node is a `Document` this is `null`.

Specified By

org.w3c.dom.Node.getOwnerDocument() in interface org.w3c.dom.Node

Returns

The document associated with this node

getParentNode()

`public` org.w3c.dom.Node `getParentNode()`

Gets the parent of this node. All nodes, except `Document`, `DocumentFragment`, and `Attr` may have a parent. However, if a node has just been created and not yet added to the tree, or if it has been removed from the tree, this is `null`.

Specified By

org.w3c.dom.Node.getParentNode() in interface org.w3c.dom.Node

Returns

The parent of this node

getPreviousSibling()

`public` org.w3c.dom.Node `getPreviousSibling()`
Gets the node immediately preceding this node. If there is no such node, this returns `null`.

Specified By

org.w3c.dom.Node.getPreviousSibling() in interface org.w3c.dom.Node

Returns

the previous node

hasChildNodes()

`public boolean hasChildNodes()`
This is a convenience method to allow easy determination of whether a node has any children.

Specified By

org.w3c.dom.Node.hasChildNodes() in interface org.w3c.dom.Node

Returns

`true` if the node has any children, `false` if the node has no children.

insertBefore(Node, Node)

`public` org.w3c.dom.Node `insertBefore(`org.w3c.dom.Node `newChild,` org.w3c.dom.Node `refChild)`
Inserts the node `newChild` before the existing child node `refChild`. If `refChild` is `null`, insert `newChild` at the end of the list of children. If `newChild` is a `DocumentFragment` object, all of its children are inserted, in the same order, before `refChild`. If the `newChild` is already in the tree, it is first removed.

Specified By

org.w3c.dom.Node.insertBefore(Node, Node) in interface org.w3c.dom.Node

Parameters

`newChild` - The node to insert.

`refChild` - The reference node, i.e., the node before which the new node must be inserted.

Returns

The node being inserted.

Throws

org.w3c.dom.DOMException - HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or if the node to insert is one of this node's ancestors. WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly. NOT_FOUND_ERR: Raised if `refChild` is not a child of this node.

print(OutputStream)

`public void print(java.io.OutputStream out)`
Writes the contents of this node to the given output stream.

Parameters

`out` - `OutputStream` to write to

Throws

`IOException` - if an error occurs

print(OutputStream, String)

`public void print(java.io.OutputStream out, java.lang.String enc)`
Writes the contents of this node to the given output stream.

Parameters

`out` - `OutputStream` to write to

`enc` - Encoding to use for the output

Throws

`IOException` - if an invalid encoding was specified or if any other error occurs

print(PrintWriter)

`public void print(java.io.PrintWriter out)`
Writes the contents of this node using the given print writer.

Parameters

`out` - `PrintWriter` to use

Throws

`IOException` - if an error occurs

removeChild(Node)

`public` org.w3c.dom.Node `removeChild`(org.w3c.dom.Node `oldChild`)
Removes the child node indicated by `oldChild` from the list of children, and returns it.

Specified By

org.w3c.dom.Node.removeChild(Node) in interface org.w3c.dom.Node

Parameters

`oldChild` - The node being removed.

Returns

The node removed.

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly. NOT_FOUND_ERR: Raised if `oldChild` is not a child of this node.

replaceChild(Node, Node)

`public` org.w3c.dom.Node `replaceChild`(org.w3c.dom.Node `newChild`, org.w3c.dom.Node `oldChild`)
Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node. If the `newChild` is already in the tree, it is first removed.

Specified By

org.w3c.dom.Node.replaceChild(Node, Node) in interface org.w3c.dom.Node

Parameters

`newChild` - The new node to put in the child list.

`oldChild` - The node being replaced in the list.

Returns

The node replaced.

Throws

org.w3c.dom.DOMException - HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or it the node to put in is one of this node's ancestors. WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly. NOT_FOUND_ERR: Raised if `oldChild` is not a child of this node.

selectNodes(String, NSResolver)

`public` org.w3c.dom.NodeList `selectNodes(java.lang.String pattern, NSResolver nsr)`
Selects nodes from the tree which match the given pattern

Parameters

`pattern` - XSL pattern to match

`nsr` - NSResolver to resolve any prefixes that occur in given pattern

Returns

a list of matching nodes

Throws

XSLException - Raised if there is an error while doing the match

selectSingleNode(String, NSResolver)

`public` org.w3c.dom.Node `selectSingleNode(java.lang.String pattern, NSResolver nsr)`
Selects the first node from the tree that matches the given pattern

Parameters

`pattern` - XSL pattern to match

`nsr` - NSResolver to resolve any prefixes that occur in given pattern

Returns

matching node

Throws

XSLException - Raised if there is an error while doing the match

setNodeValue(String)

`public void setNodeValue(java.lang.String nodeValue)`
Sets the value of this node, depending on its type

Specified By

org.w3c.dom.Node.setNodeValue(String) in interface org.w3c.dom.Node

Throws

org.w3c.dom.DOMException - NO_MODIFICATION_ALLOWED_ERR: Raised when the node is readonly. DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

transformNode(XSLStylesheet)

`public` org.w3c.dom.DocumentFragment `transformNode(XSLStylesheet xsl)`
Transforms a node in the tree using the given stylesheet

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

Returns

a document fragment

Throws

XSLException - Raised if there is an error while doing the XSL transformation.

valueOf(String, NSResolver)

`public java.lang.String valueOf(java.lang.String pattern, NSResolver nsr)`
Selects the value of the first node from the tree that matches the given pattern

Parameters

`pattern` - XSL pattern to match

`nsr` - NSResolver to resolve any prefixes that occur in given pattern

Returns

value of the matching node

Throws

XSLException - Raised if there is an error while doing the match

oracle.xml.parser.v2
XMLParseException

Syntax

public class XMLParseException extends org.xml.sax.SAXParseException

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.xml.sax.SAXException
                    |
                    +--org.xml.sax.SAXParseException
                          |
                          +--oracle.xml.parser.v2.XMLParseException
```

All Implemented Interfaces

java.io.Serializable

Description

Indicates that a parsing exception occurred while processing an XML document

Member Summary

Fields	
ERROR	Code for non-fatal error
FATAL_ERROR	Code for fatal error
WARNING	Code for warning
Constructors	
XMLParseException(String, String, String, int, int, int)	
Methods	
getColumnNumber(int)	Get the column number of error at specified index
getException(int)	Get the exception (if exists) that occured in error at specified index

Member Summary	
getLineNumber(int)	Get the line number of error at specified index
getMessage(int)	Get the error message at specified index
getMessageType(int)	Get the type of the error message at specified index
getNumMessages()	Return the total number of errors / warnings found during parsing
getPublicId(int)	Get the public ID of input when error at specified index occured
getSystemId(int)	Get the system ID of input when error at specified index occured

Inherited Member Summary

Methods inherited from interface SAXParseException

getColumnNumber(), getLineNumber(), getPublicId(), getSystemId()

Methods inherited from interface SAXException

getException(), getMessage(), toString()

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, printStackTrace, printStackTrace, printStackTrace

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Fields

ERROR

```
public static final int ERROR
```
Code for non-fatal error

FATAL_ERROR

```
public static final int FATAL_ERROR
```
Code for fatal error

WARNING

```
public static final int WARNING
```
Code for warning

Constructor

XMLParseException(String, String, String, int, int, int)

```
public  XMLParseException(java.lang.String mesg, java.lang.String pubId,
java.lang.String sysId, int line, int col, int type)
```

Methods

getColumnNumber(int)

```
public int getColumnNumber(int i)
```

Get the column number of error at specified index

Returns

The column number

getException(int)

```
public java.lang.Exception getException(int i)
```

Get the exception (if exists) that occured in error at specified index

Returns

The exception

getLineNumber(int)

```
public int getLineNumber(int i)
```

Get the line number of error at specified index

Returns

The line number

getMessage(int)

```
public java.lang.String getMessage(int i)
```

Get the error message at specified index

Returns

The error message

getMessageType(int)

`public int getMessageType(int i)`
Get the type of the error message at specified index

Returns

The error message type

getNumMessages()

`public int getNumMessages()`
Return the total number of errors / warnings found during parsing

Returns

The number of errors / warnings

getPublicId(int)

`public java.lang.String getPublicId(int i)`
Get the public ID of input when error at specified index occured

Returns

The public ID

getSystemId(int)

`public java.lang.String getSystemId(int i)`
Get the system ID of input when error at specified index occured

Returns

The system ID

oracle.xml.parser.v2
XMLParser

Syntax

```
public abstract class XMLParser extends java.lang.Object implements
oracle.xml.parser.v2.XMLConstants

java.lang.Object
  |
  +--oracle.xml.parser.v2.XMLParser
```

Direct Known Subclasses

DOMParser, SAXParser

All Implemented Interfaces

oracle.xml.parser.v2.XMLConstants

Description

This class serves as a base class for the `DOMParser` and `SAXParser` classes. It contains methods to parse eXtensible Markup Language (XML) 1.0 documents according to the World Wide Web Consortium (W3C) recommendation. This class can not be instantiated (applications may use the DOM or SAX parser depending on their requirements).

Member Summary

Methods	
getReleaseVersion()	Returns the release version of the Oracle XML Parser
getValidationMode()	Returns the validation mode
parse(InputSource)	Parses the XML from given input source
parse(InputStream)	Parses the XML from given input stream.
parse(Reader)	Parses the XML from given input stream.
parse(String)	Parses the XML from the URL indicated
parse(URL)	Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Member Summary	
setBaseURL(URL)	Set the base URL for loading external enitites and DTDs.
setDoctype(DTD)	Set the DTD
setLocale(Locale)	Applications can use this to set the locale for error reporting.
setPreserveWhitespace(boolean)	Set the white space preserving mode
setValidationMode(boolean)	Set the validation mode

Inherited Member Summary

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods

getReleaseVersion()

```
public static java.lang.String getReleaseVersion()
```

Returns the release version of the Oracle XML Parser

Returns

the release version string

getValidationMode()

```
public boolean getValidationMode()
```

Returns the validation mode

Returns

`true` if the XML parser is validating `false` if not

parse(InputSource)

`public final void parse(org.xml.sax.InputSource in)`
Parses the XML from given input source

Parameters

`in` - the `org.xml.sax.InputSouce` to parse

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(InputStream)

`public final void parse(java.io.InputStream in)`
Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters

`in` - the `InputStream` containing XML data to parse.

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also: setBaseURL(URL):

parse(Reader)

`public final void parse(java.io.Reader r)`
Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters
`r` - the `Reader` containing XML data to parse.

Throws
XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also
setBaseURL(URL)

parse(String)

`public final void parse(java.lang.String in)`
Parses the XML from the URL indicated

Parameters
`in` - the `String` containing the URL to parse from

Throws
XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(URL)

`public final void parse(java.net.URL url)`
Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters
`url` - the url points to the XML document to parse.

Throws
XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

setBaseURL(URL)

`public void setBaseURL(java.net.URL url)`
Set the base URL for loading external enitites and DTDs. This method should to be called if the parse(InputStream) is used to parse the XML Document

Parameters

`url` - The base URL

setDoctype(DTD)

`public void setDoctype(DTD dtd)`
Set the DTD

Parameters

`dtd` - `DTD` to set and used while parsing

setLocale(Locale)

`public void setLocale(java.util.Locale locale)`
Applications can use this to set the locale for error reporting.

Parameters

`locale` - `Locale` to set

Throws

org.xml.sax.SAXException - A `SAXException` could be thrown.

See Also

org.xml.sax.Parser.setLocale(Locale)

setPreserveWhitespace(boolean)

`public void setPreserveWhitespace(boolean flag)`
Set the white space preserving mode

Parameters

`flag` - preserving mode

setValidationMode(boolean)

```
public void setValidationMode(boolean yes)
```

Set the validation mode

Parameters

`yes` - determines whether the XML parser should be validating

oracle.xml.parser.v2
XMLPI

Syntax

public class XMLPI extends oracle.xml.parser.v2.CharData implements org.w3c.dom.ProcessingInstruction, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.CharData
              |
              +--oracle.xml.parser.v2.XMLPI
```

All Implemented Interfaces

org.w3c.dom.CharacterData, java.lang.Cloneable, org.w3c.dom.Node, org.w3c.dom.ProcessingInstruction, java.io.Serializable, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM Processing Instruction interface.

See Also

org.w3c.dom.ProcessingInstruction, NodeFactory, setNodeFactory(NodeFactory)

Member Summary

Constructors	
XMLPI(String, String)	Creates a new ProcessingInstruction node with the given target and the data.
Methods	
getTarget()	Returns the target of this PI.

Inherited Member Summary

Fields inherited from class XMLNode

Inherited Member Summary

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class oracle.xml.parser.v2.CharData

appendData, deleteData, getData, getLength, insertData, replaceData, setData, setNodeValue, substringData

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface ProcessingInstruction

getData(), setData(String)

Methods inherited from interface Node

Inherited Member Summary

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getNodeValue(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(String)

Methods inherited from interface CharacterData

appendData(String), deleteData(int, int), getLength(), insertData(int, String), replaceData(int, int, String), substringData(int, int)

Constructor

XMLPI(String, String)

`public  XMLPI(java.lang.String target, java.lang.String data)`

Creates a new ProcessingInstruction node with the given target and the data.

Parameters

`target` - The target of this PI

`data` - The content of this PI

Methods

getTarget()

`public java.lang.String getTarget()`

Returns the target of this PI. XML defines this as the first token following markup that begins the processing instruction.

Specified By

org.w3c.dom.ProcessingInstruction.getTarget() in interface org.w3c.dom.ProcessingInstruction

Returns

The target of the PI.

oracle.xml.parser.v2
XMLText

Syntax

public class XMLText extends oracle.xml.parser.v2.CharData implements org.w3c.dom.Text, java.io.Serializable

```
java.lang.Object
  |
  +--XMLNode
        |
        +--oracle.xml.parser.v2.CharData
              |
              +--oracle.xml.parser.v2.XMLText
```

Direct Known Subclasses:

XMLCDATA

All Implemented Interfaces

org.w3c.dom.CharacterData, java.lang.Cloneable, org.w3c.dom.Node, java.io.Serializable, org.w3c.dom.Text, oracle.xml.parser.v2.XMLConstants

Description

This class implements the DOM Text interface.

See Also

org.w3c.dom.Text, NodeFactory, setNodeFactory(NodeFactory)

Member Summary	
Constructors	
XMLText(String)	
Methods	
getNodeValue()	
splitText(int)	Breaks `Text` node into two Text nodes at specified offset, so they are both siblings, and the node only contains content up to the offset.

Inherited Member Summary

Fields inherited from class XMLNode

AMP, ASTERISK, ATTRDECL, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, ELEMENTDECL, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Fields inherited from interface Node

ATTRIBUTE_NODE, CDATA_SECTION_NODE, COMMENT_NODE, DOCUMENT_FRAGMENT_NODE, DOCUMENT_NODE, DOCUMENT_TYPE_NODE, ELEMENT_NODE, ENTITY_NODE, ENTITY_REFERENCE_NODE, NOTATION_NODE, PROCESSING_INSTRUCTION_NODE, TEXT_NODE

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class oracle.xml.parser.v2.CharData

appendData, deleteData, getData, getLength, insertData, replaceData, setData, setNodeValue, substringData

Methods inherited from class XMLNode

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(), getNodeName(), getNodeType(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), print(OutputStream), print(OutputStream, String), print(PrintWriter), removeChild(Node), replaceChild(Node, Node), selectNodes(String, NSResolver), selectSingleNode(String, NSResolver), transformNode(XSLStylesheet), valueOf(String, NSResolver)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface CharacterData

Inherited Member Summary

appendData(String), deleteData(int, int), getData(), getLength(), insertData(int, String), replaceData(int, int, String), setData(String), substringData(int, int)

Methods inherited from interface Node

appendChild(Node), cloneNode(boolean), getAttributes(), getChildNodes(), getFirstChild(), getLastChild(), getNextSibling(),getNodeName(), getNodeType(), getOwnerDocument(), getParentNode(), getPreviousSibling(), hasChildNodes(), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node), setNodeValue(`String`)

Constructor

XMLText(String)

```
public  XMLText(java.lang.String text)
```

Methods

getNodeValue()

```
public java.lang.String getNodeValue()
```

Specified By

org.w3c.dom.Node.getNodeValue() in interface org.w3c.dom.Node

Overrides

getNodeValue() in class XMLNode

splitText(int)

`public` org.w3c.dom.Text `splitText(int offset)`

Breaks `Text` node into two Text nodes at specified offset, so they are both siblings, and the node only contains content up to the offset. New node inserted as next sibling contains all content at and after the offset point.

Specified By

org.w3c.dom.Text.splitText(int) in interface org.w3c.dom.Text

Parameters

`offset` - Offset at which to split, starting from 0

Returns

New `Text` node

Throws

org.w3c.dom.DOMException - INDEX_SIZE_ERR: Raised if specified offset is negative or greater than number of characters in `data`. NO_MODIFICATION_ALLOWED_ERR: Raised if this node is readonly.

oracle.xml.parser.v2
XMLToken

Syntax

`public interface XMLToken`

Description

Basic interface for XMLToken

All XMLParser applications with Tokenizer feature must implement this interface. The interface has to be registered using `XMLParser` method `setTokenHandler(XMLToken handler)`.

If XMLtoken handler != null then for each registered and found token the parser calls the XMLToken call-back method `token(int token, String value)`. During tokenizing the parser doesn't validate the document and doesn't include / read internal / external entities. If XMLtoken handler == null then the parser parses as usual.

A request for XML token is registered (on / off) using XMLParser method `setToken (int token, boolean set)`. The requests could be registered during the parsing (from inside the call-back method) as well.

The XML tokens are defined as public constants in `XMLToken` interface. They correspond to the XML syntax variables from W3C XML Syntax Specification.

Member Summary

Fields

AttListDecl	AttListDecl ::= '<' '!' 'ATTLIST' S Name AttDef* S? '>'
AttName	AttName ::= Name
Attribute	Attribute ::= AttName Eq AttValue
AttValue	AttValue ::= '"' ([^<&"] | Reference)* '"'
CDSect	CDSect ::= CDStart CData CDEnd

Member Summary	
CharData	CharData ::= [^<&]* - ([^<&]* ']]>' [^<&]*)
Comment	Comment ::= '<' '!' '--' ((Char - '-') | ('-' (Char - '-')))* '-->'
DTDName	DTDName ::= name
ElemDeclName	ElemDeclName ::= name
elementdecl	elementdecl ::= '<' '!ELEMENT' S ElemDeclName S contentspec S? '>'
EmptyElemTag	EmptyElemTag ::= '<' STagName (S Attribute)* S? '/' '>'
EntityDecl	EntityDecl ::= '<' '!' ENTITY' S EntityDeclName S EntityDef S? '>'
EntityDeclName	EntityValue ::= '"' ([^%&"] | PEReference | Reference)* '"'
EntityValue	EntityDeclName ::= Name
ETag	ETag ::= '<' '/' ETagName S? '>'
ETagName	ETagName ::= Name
ExternalID	ExternalID ::= 'SYSTEM' S SystemLiteral
NotationDecl	NotationDecl ::= '<' '!NOTATION' S Name S (ExternalID | PublicID) S? '>'
PI	PI ::= '<' '?' PITarget (S (Char* - (Char* '?>' Char*)))? '?' '>'
PITarget	PITarget ::= Name - (('X' | 'x') ('M' | 'm') ('L' | 'l'))
Reference	Reference ::= EntityRef | CharRef | PEReference
STag	STag ::= '<' STagName (S Attribute)* S? '>'
STagName	STagName ::= Name
TextDecl	TextDecl ::= '<' '?' 'xml' VersionInfo? EncodingDecl S? '?>'
XMLDecl	XMLDecl ::= '<' '?' 'xml' VersionInfo EncodingDecl? SDDecl? S? '?' '>'
Methods	
token(int, String)	The interface call-back method.

Fields

AttListDecl

```
public static final int AttListDecl
```

AttListDecl ::= '<' '!' 'ATTLIST' S Name AttDef* S? '>'

AttName

`public static final int AttName`
AttName ::= Name

Attribute

`public static final int Attribute`
Attribute ::= AttName Eq AttValue

AttValue

`public static final int AttValue`
AttValue ::= '"' ([^<&"] | Reference)* '"'

| "'" ([^<&'] | Reference)* "'"

CDSect

`public static final int CDSect`
CDSect ::= CDStart CData CDEnd

CDStart ::= '<' '!' '[CDATA['

CData ::= (Char* - (Char* ']]>' Char*))

CDEnd ::= ']]>'

CharData

`public static final int CharData`
CharData ::= [^<&]* - ([^<&]* ']]>' [^<&]*)

Comment

`public static final int Comment`
Comment ::= '<' '!' '--' ((Char - '-') | ('-' (Char - '-')))* '-->'

DTDName

`public static final int DTDName`
DTDName ::= name

ElemDeclName

`public static final int ElemDeclName`
ElemDeclName ::= name

elementdecl

`public static final int elementdecl`

elementdecl ::= '<' '!ELEMENT' S ElemDeclName S contentspec S? '>'

EmptyElemTag

`public static final int EmptyElemTag`

EmptyElemTag ::= '<' STagName (S Attribute)* S? '/' '>'

EntityDecl

`public static final int EntityDecl`

EntityDecl ::= '<' '!' ENTITY' S EntityDeclName S EntityDef S? '>'

| '<' '!' ENTITY' S '%' S EntityDeclName S PEDef S? '>'

EntityDef ::= EntityValue | (ExternalID NDataDecl?)

PEDef ::= EntityValue | ExternalID

EntityDeclName

`public static final int EntityDeclName`

EntityValue ::= '"' ([^%&"] | PEReference | Reference)* '"'

| "'" ([^%&'] | PEReference | Reference)* "'"

EntityValue

`public static final int EntityValue`

EntityDeclName ::= Name

ETag

`public static final int ETag`

ETag ::= '<' '/' ETagName S? '>'

ETagName

`public static final int ETagName`

ETagName ::= Name

ExternalID

`public static final int ExternalID`

ExternalID ::= 'SYSTEM' S SystemLiteral

| 'PUBLIC' S PubidLiteral S SystemLiteral

NotationDecl

`public static final int NotationDecl`
NotationDecl ::= '<' '!NOTATION' S Name S (ExternalID | PublicID) S? '>'

PI

`public static final int PI`
PI ::= '<' '?' PITarget (S (Char* - (Char* '?>' Char*)))? '?' '>'

PITarget

`public static final int PITarget`
PITarget ::= Name - (('X' | 'x') ('M' | 'm') ('L' | 'l'))

Reference

`public static final int Reference`
Reference ::= EntityRef | CharRef | PEReference

EntityRef ::= '&' Name ';'

PEReference ::= '%' Name ';'

CharRef ::= '&#' [0-9]+ ';' | '&#x' [0-9a-fA-F]+ ';

STag

`public static final int STag`
STag ::= '<' STagName (S Attribute)* S? '>'

STagName

`public static final int STagName`
STagName ::= Name

TextDecl

`public static final int TextDecl`
TextDecl ::= '<' '?' 'xml' VersionInfo? EncodingDecl S? '?>'

XMLDecl

`public static final int XMLDecl`
XMLDecl ::= '<' '?' 'xml' VersionInfo EncodingDecl? SDDecl? S? '?' '>'

Methods

token(int, String)

```
public void token(int token, java.lang.String value)
```

The interface call-back method. Receives an XML token and it's corresponding value

Parameters

`token` - The XML token constant as specified in the interface.

`value` - The corresponding substring from the parsed text.

oracle.xml.parser.v2
XMLTokenizer

Syntax

```
public class XMLTokenizer extends oracle.xml.parser.v2 implements
oracle.xml.parser.v2.XMLConstants

java.lang.Object
  |
  +--org.xml.sax.HandlerBase
        |
        +--oracle.xml.parser.v2
              |
              +--oracle.xml.parser.v2.XMLTokenizer
```

All Implemented Interfaces

org.xml.sax.DocumentHandler, org.xml.sax.DTDHandler, org.xml.sax.EntityResolver, org.xml.sax.ErrorHandler, oracle.xml.parser.v2.XMLConstants, XMLDocumentHandler

Description

This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.

Member Summary	
Constructors	
XMLTokenizer()	Creates a new Tokenizer object.
XMLTokenizer(XMLToken)	Creates a new Tokenizer object.
Methods	
parseDocument()	Document ::= Prolog Element Misc*
setErrorHandler(ErrorHandler)	Applications can use this to register a new error event handler.
setErrorStream(OutputStream)	Register a output stream for errors
setToken(int, boolean)	Applications can use this to register a new token for XML tokenizer.
setTokenHandler(XMLToken)	Applications can use this to register a new XML tokenizer event handler.
tokenize(InputSource)	Tokenizes the XML from given input source

Member Summary

tokenize(InputStream)	Tokenizes the XML from given input stream.
tokenize(Reader)	Tokenizes the XML from given input stream.
tokenize(String)	Tokenizes the XML from the URL indicated
tokenize(URL)	Tokenizes the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Inherited Member Summary

Fields inherited from interface oracle.xml.parser.v2.XMLConstants

AMP, ASTERISK, cANY, cATTLIST, cCDATA, cCDATAEND, cCDATASTART, cCOMMENTEND, cCOMMENTSTART, cDECCREF, cDECLSTART, cDOCTYPE, cELEMENT, cEMPTY, cEMPTYTAGEND, cENCODING, cENDTAGSTART, cENTITIES, cENTITY, cFIXED, cHEXCREF, cID, cIDREF, cIDREFS, cIGNORE, cIMPLIED, cINCLUDE, cNDATA, cNMTOKEN, cNMTOKENS, cNOTATION, COLON, COMMA, cPIEND, cPISTART, cPUBLIC, cREQUIRED, cSTANDALONE, cSYSTEM, cVERSION, cXML, DOUBLEQUOTE, EOF, EQ, ERROR, FATAL_ERROR, FDIGIT, FLETTER, FMISCNAME, FSTARTNAME, FWHITESPACE, HASH, ICOUNT, ISTART, LEFTSQB, LPAREN, nameCDATA, nameCOMMENT, nameDOCUMENT, nameDOCUMENTFRAGMENT, nameENCODING, nameNameSpace, nameSpaceSeparator, nameSTANDALONE, nameTEXT, nameVERSION, nameXML, nameXMLLang, nameXMLNamespace, nameXMLNSNamespace, nameXMLSpace, nameXSLPI, NONVALIDATING, OR, PERCENT, PLUS, QMARK, QUOTE, RIGHTSQB, RPAREN, SEMICOLON, SLASH, TAGEND, TAGSTART, VALIDATING, WARNING

Methods inherited from class oracle.xml.parser.v2

cDATASection(char[], int, int), comment(String), endDoctype(), endElement(NSName), setDoctype(DTD), setTextDecl(String, String), setXMLDecl(String, String, String), startElement(NSName, SAXAttrList)

Methods inherited from class HandlerBase

characters(char[], int, int), endDocument(), endElement(String), error(SAXParseException), fatalError(SAXParseException), ignorableWhitespace(char[], int, int), notationDecl(String, String, String), processingInstruction(String, String), resolveEntity(String, String), setDocumentLocator(Locator), startDocument(), startElement(String, AttributeList), unparsedEntityDecl(String, String, String, String), warning(SAXParseException)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface XMLDocumentHandler

cDATASection(char[ ], int, int), comment(String), endDoctype(), endElement(NSName), setDoctype(DTD), setTextDecl(String, String), setXMLDecl(String, String, String), startElement(NSName, SAXAttrList)

Methods inherited from interface DocumentHandler

Inherited Member Summary

characters(char[], int, int), endDocument(), endElement(String), ignorableWhitespace(char[], int, int), processingInstruction(String, String), setDocumentLocator(Locator), startDocument(), startElement(String, AttributeList)

Methods inherited from interface EntityResolver

resolveEntity(String, String)

Methods inherited from interface DTDHandler

notationDecl(String, String, String), unparsedEntityDecl(String, String, String, String)

Methods inherited from interface ErrorHandler

error(SAXParseException), fatalError(SAXParseException), warning(SAXParseException)

Constructors

XMLTokenizer()

`public XMLTokenizer()`
Creates a new Tokenizer object.

XMLTokenizer(XMLToken)

`public XMLTokenizer(XMLToken handler)`
Creates a new Tokenizer object.

Methods

parseDocument()

`public void parseDocument()`
Document ::= Prolog Element Misc*

setErrorHandler(ErrorHandler)

`public void setErrorHandler(`org.xml.sax.ErrorHandler `handler)`
Applications can use this to register a new error event handler. This replaces any previous setting for error handling.

Parameters

`handler` - `ErrorHandler` being registered

setErrorStream(OutputStream)

`public void setErrorStream(java.io.OutputStream out)`
Register a output stream for errors

setToken(int, boolean)

`public void setToken(int token, boolean val)`
Applications can use this to register a new token for XML tokenizer.

Parameters

`token` - `XMLToken` being set

setTokenHandler(XMLToken)

`public void setTokenHandler(XMLToken handler)`
Applications can use this to register a new XML tokenizer event handler.

Parameters

`handler` - `XMLToken` being registered

tokenize(InputSource)

`public final void tokenize(`org.xml.sax.InputSource `in)`
Tokenizes the XML from given input source

Parameters

`in` - the `org.xml.sax.InputSouce` to parse

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

tokenize(InputStream)

`public final void tokenize(java.io.InputStream in)`
Tokenizes the XML from given input stream.

Parameters

`in` - the `InputStream` containing XML data to parse.

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also

setBaseURL(URL)

tokenize(Reader)

`public final void tokenize(java.io.Reader r)`
Tokenizes the XML from given input stream.

Parameters

`r` - the `Reader` containing XML data to parse.

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also

setBaseURL(URL)

tokenize(String)

`public final void tokenize(java.lang.String in)`
Tokenizes the XML from the URL indicated

Parameters

`in` - the `String` containing the URL to parse from

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

tokenize(URL)

`public final void tokenize(java.net.URL url)`
Tokenizes the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters

`url` - the url points to the XML document to parse.

Throws

XMLParseException - if syntax or other error encountered.

org.xml.sax.SAXException - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

oracle.xml.parser.v2
XSLException

Syntax

```
public class XSLException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--oracle.xml.parser.v2.XSLException
```

All Implemented Interfaces

`java.io.Serializable`

Description

Indicates that an exception occurred during XSL tranformation

Inherited Member Summary

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, getMessage, printStackTrace, printStackTrace, printStackTrace, toString

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

oracle.xml.parser.v2
XSLProcessor

Syntax

```
public class XSLProcessor extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.parser.v2.XSLProcessor
```

Description

This class provides methods to transform an input XML document using a previously constructed `XSLStylesheet`. The transformation effected is as specified by the XSLT 1.0 specification.

Member Summary

Constructors

XSLProcessor()

Methods

processXSL(XSLStylesheet, InputStream, URL)	Transform input XML document using given InputStream and stylesheet.
processXSL(XSLStylesheet, Reader, URL)	Transform input XML document using given Reader and stylesheet.
processXSL(XSLStylesheet, URL, URL)	Transform input XML document using given URL and stylesheet.
processXSL(XSLStylesheet, XMLDocument)	Transform input XML document using given XMLDocument and stylesheet.
processXSL(XSLStylesheet, XMLDocumentFragment)	Transform input XML document using given XMLDocument and stylesheet.
processXSL(XSLStylesheet, XMLDocumentFragment, OutputStream)	Transform input XML using given XMLDocumentFragment and stylesheet.
processXSL(XSLStylesheet, XMLDocumentFragment, PrintWriter)	Transform input XML using given XMLDocumentFragment and stylesheet.

Member Summary	
processXSL(XSLStylesheet, XMLDocument, OutputStream)	Transform input XML document using given XMLDocument and stylesheet.
processXSL(XSLStylesheet, XMLDocument, PrintWriter)	Transform input XML document using given XMLDocument and stylesheet.
setErrorStream(OutputStream)	Creates an output stream for the output of warnings.
setLocale(Locale)	Applications can use this to set the locale for error reporting.
showWarnings(boolean)	Switch to determine whether to output warnings.

Inherited Member Summary
Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor

XSLProcessor()

```
public  XSLProcessor()
```

Methods

processXSL(XSLStylesheet, InputStream, URL)

```
public XMLDocumentFragment processXSL(XSLStylesheet xsl, java.io.InputStream
xml, java.net.URL ref)
```

Transforms input XML document using given InputStream and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a java.io.Inputstream)

`ref` - Reference URL to resolve external entities in input xml file

Returns

`XMLDocumentFragment`

Throws

XSLException - on error.

processXSL(XSLStylesheet, Reader, URL)

`public XMLDocumentFragment processXSL(XSLStylesheet xsl, java.io.Reader xml, java.net.URL ref)`
Transform input XML document using given Reader and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a java.io.Reader)

`ref` - Reference URL to resolve external entities in input xml file

Returns

`XMLDocumentFragment`

Throws

XSLException - on error.

processXSL(XSLStylesheet, URL, URL)

`public XMLDocumentFragment processXSL(XSLStylesheet xsl, java.net.URL xml, java.net.URL ref)`
Transform input XML document using given URL and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a java.net.URL)

`ref` - Reference URL to resolve external entities in input xml file

Returns

`XMLDocumentFragment`

Throws

XSLException - on error.

processXSL(XSLStylesheet, XMLDocument)

`public XMLDocumentFragment processXSL(XSLStylesheet xsl, XMLDocument xml)`
Transform input XML document using given XMLDocument and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a DOM Tree)

Returns

`XMLDocumentFragment`

Throws

XSLException - on error.

processXSL(XSLStylesheet, XMLDocumentFragment)

`public XMLDocumentFragment processXSL(XSLStylesheet xsl, XMLDocumentFragment inp)`
Transform input XML document using given XMLDocument and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a DOM Tree)

Returns

`XMLDocumentFragment`

Throws

XSLException - on error.

processXSL(XSLStylesheet, XMLDocumentFragment, OutputStream)

`public void processXSL(XSLStylesheet xsl, XMLDocumentFragment xml, java.io.OutputStream out)`
Transform input XML using given XMLDocumentFragment and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation `xml` - XML input to be transformed (as a DOM Tree)

`out` - Outputstream to which the result is printed

Throws

`XSLException,` - IOException on error.

processXSL(XSLStylesheet, XMLDocumentFragment, PrintWriter)

`public void processXSL(XSLStylesheet xsl, XMLDocumentFragment xml, java.io.PrintWriter pw)`
Transform input XML using given XMLDocumentFragment and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a DOM Tree)

`pw` - PrintWriter to which the result is printed

Throws

`XSLException,` - IOException on error.

processXSL(XSLStylesheet, XMLDocument, OutputStream)

`public void processXSL(XSLStylesheet xsl, XMLDocument xml, java.io.OutputStream out)`
Transform input XML document using given XMLDocument and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a DOM Tree)

`out` - Outputstream to which the result is printed

Throws

`XSLException,` - IOException on error.

processXSL(XSLStylesheet, XMLDocument, PrintWriter)

`public void processXSL(XSLStylesheet xsl, XMLDocument xml, java.io.PrintWriter pw)`

Transform input XML document using given XMLDocument and stylesheet.

Parameters

`xsl` - `XSLStylesheet` to be used for transformation

`xml` - XML input to be transformed (as a DOM Tree)

`pw` - PrintWriter to which the result is printed

Throws

`XSLException,` - IOException on error.

setErrorStream(OutputStream)

`public final void setErrorStream(java.io.OutputStream out)`
Creates an output stream for the output of warnings. If an output stream for warnings is not specified, the processor will not output any warnings

Parameters

`out` - The output stream to use for errors and warnings

setLocale(Locale)

`public void setLocale(java.util.Locale locale)`
Applications can use this to set the locale for error reporting.

Parameters

`locale` - `Locale` to set

showWarnings(boolean)

`public final void showWarnings(boolean yes)`
Switch to determine whether to output warnings.

Parameters

`yes` - determines whether warnings should be shown By default, warnings are not output

oracle.xml.parser.v2
XSLStylesheet

Syntax

```
public class XSLStylesheet extends java.lang.Object implements
oracle.xml.parser.v2.XSLConstants

java.lang.Object
  |
  +--oracle.xml.parser.v2.XSLStylesheet
```

All Implemented Interfaces

`oracle.xml.parser.v2.XSLConstants`

Description

The class holds XSL stylesheet information such as templates, keys, variables, and attribute sets. The same stylesheet, once constructed, can be used to transform multiple XML documents.

Member Summary

Constructors	
XSLStylesheet(InputStream, URL)	Constructs an XSLStylesheet using the given Inputstream
XSLStylesheet(Reader, URL)	Constructs an XSLStylesheet using the given Reader
XSLStylesheet(URL, URL)	Constructs an XSLStylesheet using the given URL
XSLStylesheet(XMLDocument, URL)	Constructs an XSLStylesheet using the given XMLDocument
Methods	
setParam(String, String)	Sets the value of a top-level stylesheet parameter.

Inherited Member Summary

Fields inherited from interface oracle.xml.parser.v2.XSLConstants

Inherited Member Summary

APPLY_IMPORTS, APPLY_TEMPLATES, ATTRIBUTE, ATTRIBUTE_SET, CALL_TEMPLATE, CHOOSE, COMMENT, COPY, COPY_OF, DISABLEOUTESC, ELEMENT, FOR_EACH, HREF, IF, IMPORT, INCLUDE, KEY, LOCALE, MATCH, MESSAGE, NAME, NEGINFPRIORITY, NUMBER, ORACLE_NAME, ORACLE_URL, OTHERWISE, OUTPUT, PARAM, PARAM_VARIABLE, PI, PRESERVE_SPACE, RESULT_ROOT, SORT, STRIP_SPACE, TEMPLATE, TEXT, USE, USE_ATTRIBUTE_SETS, VALUE_OF, VARIABLE, WHEN, XSL_ROOT, XSLEXTFUNCNS, XSLNAMESPACE, XSLT_SPEC_VERSION

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSLStylesheet(InputStream, URL)

`public  XSLStylesheet(java.io.InputStream xsl, java.net.URL ref)`

Constructs an XSLStylesheet using the given Inputstream

Parameters

`xsl` - XSL input as an Inputstream

`ref` - Reference URL for include, import and external entities

Throws

XSLException - on error.

XSLStylesheet(Reader, URL)

`public  XSLStylesheet(java.io.Reader xsl, java.net.URL ref)`

Constructs an XSLStylesheet using the given Reader

Parameters

`xsl` - XSL input as a Reader

`ref` - Reference URL for include, import and external entities

Throws

XSLException - on error.

XSLStylesheet(URL, URL)

`public  XSLStylesheet(java.net.URL xsl, java.net.URL ref)`

Constructs an XSLStylesheet using the given URL

Parameters

`xsl` - XSL input as a URL

`ref` - Reference URL for include, import and external entities

Throws

XSLException - on error.

XSLStylesheet(XMLDocument, URL)

```
public  XSLStylesheet(XMLDocument xsl, java.net.URL ref)
```

Constructs an XSLStylesheet using the given XMLDocument

Parameters

`xsl` - XSL input as a DOM Tree

`ref` - Reference URL for include, import

Throws

XSLException - on error.

Methods

setParam(String, String)

```
public void setParam(java.lang.String name, java.lang.String value)
```

Sets the value of a top-level stylesheet parameter. The parameter value is expected to be a valid XPath expression (note that string literal values would therefore have to be explicitly quoted).

Parameters

`name` - parameter name

`value` - parameter value as an XPath expression

Throws

XSLException - on error

2

XML Class Generator for Java

Description

Class Summary	
Classes	
CGDocument	Serves as the base document class for the Class Generated generated classes
CGNode	Serves as the base class for nodes generated by the Class Generated
ClassGenerator	This class is used by the Class Generated to generate classes
Exceptions	
InvalidContentException	Definition of InvalidContentException thrown by dtdcompiler classes

oracle.xml.classgen

CGDocument

Syntax

```
public abstract class CGDocument extends CGNode

java.lang.Object
  |
  +--CGNode
        |
        +--oracle.xml.classgen.CGDocument
```

Description

Serves as the base document class for the Class Generated generated classes

Inherited Member Summary

Methods inherited from class CGNode

setDocument(CGDocument)

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

oracle.xml.classgen

CGNode

Syntax

```
public abstract class CGNode extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.classgen.CGNode
```

Direct Known Subclasses:

`CGDocument`

Description

Serves as the base class for nodes generated by the Class Generated

Member Summary

Methods	
setDocument(CGDocument)	Sets the base document (root Element)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods

setDocument(CGDocument)

`public void setDocument(CGDocument d)`

Sets the base document (root Element)

Parameters:

`d` - Base CGDocument

oracle.xml.classgen ClassGenerator

Syntax

```
public class ClassGenerator extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.classgen.ClassGenerator
```

Description

This class is used by the Class Generated to generate classes

Member Summary	
Constructors	
ClassGenerator()	Default constructor for ClassGenerator.
Methods	
generate(DTD, String)	Traverses the DTD with element `doctype` as root and generates Java classes
setGenerateComments(boolean)	Switch to determine whether to generate java doc comments Default - TRUE
setJavaPackage(String)	Sets the package for the classes generated Default - No package
setOutputDirectory(String)	Sets the output directory Default - current directory
setSerializationMode(boolean)	Switch to determine if the DTD should be saved as a serialized object or as text file.
setValidationMode(boolean)	Switch to determine whether the classes generated should validate the XML Document being constructed Default - TRUE

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ClassGenerator()

`public  ClassGenerator()`

Default constructor for ClassGenerator.

Methods

generate(DTD, String)

`public void generate(oracle.xml.classgen.DTD dtd, java.lang.String doctype)`

Traverses the DTD with element `doctype` as root and generates Java classes

Parameters:

`DTD` - The DTD used to generate the classes

`doctype` - Name of the root Element

setGenerateComments(boolean)

`public void setGenerateComments(boolean comments)`

Switch to determine whether to generate java doc comments Default - TRUE

Parameters:

`comments` - boolean flag

setJavaPackage(String)

`public void setJavaPackage(java.lang.String packageName)`

Sets the package for the classes generated Default - No package

Parameters:

`packageName` - Name of the package

setOutputDirectory(String)

`public void setOutputDirectory(java.lang.String dir)`

Sets the output directory Default - current directory

Parameters:

`dir` - Output directory

setSerializationMode(boolean)

`public void setSerializationMode(boolean flag)`

Switch to determine if the DTD should be saved as a serialized object or as text file. Serializing the DTD improves the performance when the generated classes are used to author XML files. Default - FALSE (DTD is saved a text file)

Parameters:

`flag` - boolean flag

setValidationMode(boolean)

`public void setValidationMode(boolean flag)`

Switch to determine whether the classes generated should validate the XML Document being constructed Default - TRUE

Parameters:

`flag` - boolean flag

oracle.xml.classgen InvalidContentException

Syntax

```
public class InvalidContentException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--oracle.xml.classgen.InvalidContentException
```

All Implemented Interfaces:

`java.io.Serializable`

Description

Definition of InvalidContentException thrown by dtdcompiler classes

Member Summary

Constructors

InvalidContentException()

InvalidContentException(String)

Inherited Member Summary

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, getMessage, printStackTrace, printStackTrace, printStackTrace, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

InvalidContentException()

```
public  InvalidContentException()
```

InvalidContentException(String)

```
public  InvalidContentException(java.lang.String s)
```

3 XSQL Servlet

Description

Class Summary

Interfaces	
XSQLActionHandler	Interface that must be implemented by all XSQL Action Element Handlers
XSQLPageRequest	Interface representing a request for an XSQL Page
Classes	
Res	
XSQLActionHandlerImpl	Base Implementation of XSQLActionHandler that can be extended to create your own custom handlers.
XSQLCommandLine	Command-line Utility to process XSQL Pages.
XSQLDiagnostic	
XSQLHttpUtil	
XSQLPageRequestImpl	Base implementation of the XSQLPageRequest interface that case be used to derive new kinds of page request implementations.
XSQLParserHelper	Common XML Parsing Routines
XSQLRequest	Programmatically process a request for an XSQL Page.
XSQLServlet	Servlet to enable HTTP GET-ing of and POST-ing to XSQL Pages
XSQLServletPageRequest	Implementation of XSQLPageRequest for Servlet-based XSQL Page requests.
XSQLStylesheetProcessor	XSLT Stylesheet Processing Engine
XSQLUtil	

oracle.xml.xsql
Res

Syntax

```
public class Res extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.Res
```

Description

Member Summary

Fields

CANTREAD_XSQL

CANTREAD_XSQL_MSG

CLASSNOTFOUND

CLASSNOTFOUND_MSG

CONN_FILE

CONN_FILE_MSG

ERR_OUTPUT

ERR_OUTPUT_MSG

ERRORINCLUDING

ERRORINCLUDING_MSG

ERRORLOADINGURL

ERRORLOADINGURL_MSG

ERRORREADINGPARAM

ERRORREADINGPARAM_MSG

FATAL_SHEETPOOL

FATAL_SHEETPOOL_MSG

Member Summary

ILLFORMEDXMLPARAMVAL

ILLFORMEDXMLPARAMVAL_MSG

ILLFORMEDXMLRESOURCE

ILLFORMEDXMLRESOURCE_MSG

INSTANTIATIONERR

INSTANTIATIONERR_MSG

INVALID_URI

INVALID_URI_MSG

INVALIDURL

INVALIDURL_MSG

MISSING_ARGS

MISSING_ARGS_MSG

MISSING_ATTR

MISSING_ATTR_MSG

NAMED_CONN

NAMED_CONN_MSG

NO_CONN

NO_CONN_DEF

NO_CONN_DEF_MSG

NO_CONN_MSG

NO_XSQL_FILE

NO_XSQL_FILE_MSG

NOFUNCTIONNAME

NOFUNCTIONNAME_MSG

NOPOSTEDXML

NOPOSTEDXML_MSG

NOQUERYSUPPLIED

NOQUERYSUPPLIED_MSG

Member Summary

NOTANACTIONHANDLER

NOTANACTIONHANDLER_MSG

NULLPARAM

NULLPARAM_MSG

OWAXMLMALFORMED

OWAXMLMALFORMED_MSG

POSTEDXML_ERR

POSTEDXML_ERR_MSG

UNHANDLED_ERR

UNHANDLED_ERR_MSG

UNHANDLED_ERR_XSL_PR

UNHANDLED_ERR_XSL_PR_MSG

UNHANDLED_ERR_XSL_RD

UNHANDLED_ERR_XSL_RD_MSG

XML_INS_ERR

XML_INS_ERR_MSG

XML_PARSE

XML_PARSE_MSG

XML_SQL_ERR

XML_SQL_ERR_MSG

XSL_ERRORS

XSL_ERRORS_MSG

XSL_NOFILE

XSL_NOFILE_MSG

XSL_PARSE

XSL_PARSE_MSG

XSLNOTFOUND

XSLNOTFOUND_MSG

Member Summary

Constructors

Res()

Methods

format(int, String)

getString(int)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

CANTREAD_XSQL

```
public static final int CANTREAD_XSQL
```

CANTREAD_XSQL_MSG

```
public static final java.lang.String CANTREAD_XSQL_MSG
```

CLASSNOTFOUND

```
public static final int CLASSNOTFOUND
```

CLASSNOTFOUND_MSG

```
public static final java.lang.String CLASSNOTFOUND_MSG
```

CONN_FILE

```
public static final int CONN_FILE
```

CONN_FILE_MSG

```
public static final java.lang.String CONN_FILE_MSG
```

ERR_OUTPUT

```
public static final int ERR_OUTPUT
```

ERR_OUTPUT_MSG

```
public static final java.lang.String ERR_OUTPUT_MSG
```

ERRORINCLUDING

```
public static final int ERRORINCLUDING
```

ERRORINCLUDING_MSG

```
public static final java.lang.String ERRORINCLUDING_MSG
```

ERRORLOADINGURL

```
public static final int ERRORLOADINGURL
```

ERRORLOADINGURL_MSG

```
public static final java.lang.String ERRORLOADINGURL_MSG
```

ERRORREADINGPARAM

```
public static final int ERRORREADINGPARAM
```

ERRORREADINGPARAM_MSG

```
public static final java.lang.String ERRORREADINGPARAM_MSG
```

FATAL_SHEETPOOL

```
public static final int FATAL_SHEETPOOL
```

FATAL_SHEETPOOL_MSG

```
public static final java.lang.String FATAL_SHEETPOOL_MSG
```

ILLFORMEDXMLPARAMVAL

```
public static final int ILLFORMEDXMLPARAMVAL
```

ILLFORMEDXMLPARAMVAL_MSG

```
public static final java.lang.String ILLFORMEDXMLPARAMVAL_MSG
```

ILLFORMEDXMLRESOURCE

```
public static final int ILLFORMEDXMLRESOURCE
```

ILLFORMEDXMLRESOURCE_MSG

```
public static final java.lang.String ILLFORMEDXMLRESOURCE_MSG
```

INSTANTIATIONERR

```
public static final int INSTANTIATIONERR
```

INSTANTIATIONERR_MSG

```
public static final java.lang.String INSTANTIATIONERR_MSG
```

INVALID_URI

```
public static final int INVALID_URI
```

INVALID_URI_MSG

```
public static final java.lang.String INVALID_URI_MSG
```

INVALIDURL

```
public static final int INVALIDURL
```

INVALIDURL_MSG

```
public static final java.lang.String INVALIDURL_MSG
```

MISSING_ARGS

```
public static final int MISSING_ARGS
```

MISSING_ARGS_MSG

```
public static final java.lang.String MISSING_ARGS_MSG
```

MISSING_ATTR

```
public static final int MISSING_ATTR
```

MISSING_ATTR_MSG

```
public static final java.lang.String MISSING_ATTR_MSG
```

NAMED_CONN

```
public static final int NAMED_CONN
```

NAMED_CONN_MSG

```
public static final java.lang.String NAMED_CONN_MSG
```

NO_CONN

```
public static final int NO_CONN
```

NO_CONN_DEF

```
public static final int NO_CONN_DEF
```

NO_CONN_DEF_MSG

```
public static final java.lang.String NO_CONN_DEF_MSG
```

NO_CONN_MSG

```
public static final java.lang.String NO_CONN_MSG
```

NO_XSQL_FILE

```
public static final int NO_XSQL_FILE
```

NO_XSQL_FILE_MSG

```
public static final java.lang.String NO_XSQL_FILE_MSG
```

NOFUNCTIONNAME

```
public static final int NOFUNCTIONNAME
```

NOFUNCTIONNAME_MSG

```
public static final java.lang.String NOFUNCTIONNAME_MSG
```

NOPOSTEDXML

```
public static final int NOPOSTEDXML
```

NOPOSTEDXML_MSG

```
public static final java.lang.String NOPOSTEDXML_MSG
```

NOQUERYSUPPLIED

```
public static final int NOQUERYSUPPLIED
```

NOQUERYSUPPLIED_MSG

```
public static final java.lang.String NOQUERYSUPPLIED_MSG
```

NOTANACTIONHANDLER

```
public static final int NOTANACTIONHANDLER
```

NOTANACTIONHANDLER_MSG

```
public static final java.lang.String NOTANACTIONHANDLER_MSG
```

NULLPARAM

```
public static final int NULLPARAM
```

NULLPARAM_MSG

```
public static final java.lang.String NULLPARAM_MSG
```

OWAXMLMALFORMED

```
public static final int OWAXMLMALFORMED
```

OWAXMLMALFORMED_MSG

```
public static final java.lang.String OWAXMLMALFORMED_MSG
```

POSTEDXML_ERR

```
public static final int POSTEDXML_ERR
```

POSTEDXML_ERR_MSG

```
public static final java.lang.String POSTEDXML_ERR_MSG
```

UNHANDLED_ERR

```
public static final int UNHANDLED_ERR
```

UNHANDLED_ERR_MSG

```
public static final java.lang.String UNHANDLED_ERR_MSG
```

UNHANDLED_ERR_XSL_PR

```
public static final int UNHANDLED_ERR_XSL_PR
```

UNHANDLED_ERR_XSL_PR_MSG

```
public static final java.lang.String UNHANDLED_ERR_XSL_PR_MSG
```

UNHANDLED_ERR_XSL_RD

```
public static final int UNHANDLED_ERR_XSL_RD
```

UNHANDLED_ERR_XSL_RD_MSG

```
public static final java.lang.String UNHANDLED_ERR_XSL_RD_MSG
```

XML_INS_ERR

```
public static final int XML_INS_ERR
```

XML_INS_ERR_MSG

```
public static final java.lang.String XML_INS_ERR_MSG
```

XML_PARSE

```
public static final int XML_PARSE
```

XML_PARSE_MSG

```
public static final java.lang.String XML_PARSE_MSG
```

XML_SQL_ERR

```
public static final int XML_SQL_ERR
```

XML_SQL_ERR_MSG

```
public static final java.lang.String XML_SQL_ERR_MSG
```

XSL_ERRORS

```
public static final int XSL_ERRORS
```

XSL_ERRORS_MSG

```
public static final java.lang.String XSL_ERRORS_MSG
```

XSL_NOFILE

```
public static final int XSL_NOFILE
```

XSL_NOFILE_MSG

```
public static final java.lang.String XSL_NOFILE_MSG
```

XSL_PARSE

```
public static final int XSL_PARSE
```

XSL_PARSE_MSG

```
public static final java.lang.String XSL_PARSE_MSG
```

XSLNOTFOUND

```
public static final int XSLNOTFOUND
```

XSLNOTFOUND_MSG

```
public static final java.lang.String XSLNOTFOUND_MSG
```

Constructors

Res()

```
public  Res()
```

Methods

format(int, String)

```
public static java.lang.String format(int id, java.lang.String s)
```

getString(int)

```
public static java.lang.String getString(int id)
```

oracle.xml.xsql XSQLActionHandler

Syntax

`public interface XSQLActionHandler`

All Known Implementing Classes:

`XSQLActionHandlerImpl`

Description

Interface that must be implemented by all XSQL Action Element Handlers

Upon encountering an XSQL Action Element of the form <xsql:xxxx> in an XSQL page, the XSQL Page Processor invokes the associated XSQL Action Handler by:

1. Constructing an instance of the handler using the no-args constructor

Invoking the XSQL Action Handler's init() method

Invoking the XSQL Action Handler's handleAction() method

NOTE: conn parameter can be null if no connection specified for the XSQL page being processed.

Member Summary	
Methods	
handleAction(Node)	Handle the action, typically by executing some code and appending new child DOM nodes to the rootNode.
init(XSQLPageRequest, Element)	Initialize the Action Handler

Methods

handleAction(Node)

`public void handleAction(oracle.xml.xsql.Node rootNode)`

Handle the action, typically by executing some code and appending new child DOM nodes to the rootNode.

The XSQL Page Processor replaces the action element in the XSQL Page being processed with the document fragment of nodes that your handleAction method appends to the rootNode.

Parameters:

`rootNode` - Root node of generated document fragment

init(XSQLPageRequest, Element)

`public void init(XSQLPageRequest env, oracle.xml.xsql.Element e)`

Initialize the Action Handler

Parameters:

`env` - XSQLPageRequest object

`e` - DOM element representing the Action Element being handled

oracle.xml.xsql XSQLActionHandlerImpl

Syntax

`public abstract class XSQLActionHandlerImpl extends java.lang.Object implements XSQLActionHandler`

```
java.lang.Object
  |
  +--oracle.xml.xsql.XSQLActionHandlerImpl
```

All Implemented Interfaces:

`XSQLActionHandler`

Description

Base Implementation of XSQLActionHandler that can be extended to create your own custom handlers.

Includes a set of useful helper methods.

NOTE: If you extend this class and override the init() method, make sure to call:

`super.init(env,e);`

Member Summary

Constructors

XSQLActionHandlerImpl()

Methods

init(XSQLPageRequest, Element)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface XSQLActionHandler

Inherited Member Summary

handleAction(Node)

Constructors

XSQLActionHandlerImpl()

```
public  XSQLActionHandlerImpl()
```

Methods

init(XSQLPageRequest, Element)

```
public void init(XSQLPageRequest env, oracle.xml.xsql.Element e)
```

Specified By:

init(XSQLPageRequest, Element) in interface XSQLActionHandler

oracle.xml.xsql XSQLCommandLine

Syntax

```
public final class XSQLCommandLine extends java.lang.Object
```

```
java.lang.Object
  |
  +--oracle.xml.xsql.XSQLCommandLine
```

Description

Command-line Utility to process XSQL Pages.

Member Summary

Constructors

XSQLCommandLine()

Methods

main(String[])

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLCommandLine()

```
public  XSQLCommandLine()
```

Methods

main(String[])

```
public static void main(java.lang.String[] args)
```

oracle.xml.xsql
XSQLDiagnostic

Syntax

```
public final class XSQLDiagnostic extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLDiagnostic
```

Description

Member Summary

Constructors

XSQLDiagnostic(String)

Methods

debugPrintToFile(String, String)

msg(String)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLDiagnostic(String)

```
public  XSQLDiagnostic(java.lang.String diagFile)
```

Methods

debugPrintToFile(String, String)

```
public static void debugPrintToFile(java.lang.String msg, java.lang.String
filename)
```

msg(String)

```
public void msg(java.lang.String text)
```

oracle.xml.xsql XSQLHttpUtil

Syntax

```
public final class XSQLHttpUtil extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLHttpUtil
```

Description

Member Summary
Constructors
XSQLHttpUtil()
Methods
HttpRequestAsXMLDocument(HttpServletRequest, String)
XL(String, String)

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLHttpUtil()

```
public  XSQLHttpUtil()
```

Methods

HttpRequestAsXMLDocument(HttpServletRequest, String)

```
public static oracle.xml.xsql.XMLDocument
HttpRequestAsXMLDocument(oracle.xml.xsql.HttpServletRequest req,
java.lang.String XSQLPageEncoding)
```

XL(String, String)

```
public static java.lang.String XL(java.lang.String s, java.lang.String enc)
```

oracle.xml.xsql
XSQLPageRequest

Syntax

```
public interface XSQLPageRequest
```

All Known Implementing Classes:

XSQLPageRequestImpl

Description

Interface representing a request for an XSQL Page

Member Summary

Methods	
createNestedRequest(URL, Dictionary)	Returns an instance of a nested Request
getConnectionName()	Returns the name of the connection being used for this request May be null if no connection set / in-use.
getErrorWriter()	Returns a PrintWriter to print out errors processing this request
getJDBCConnection()	Gets the JDBC connection being used for this request (can be null)
getPageEncoding()	Returns encoding of source XSQL Page associated with this request
getParameter(String)	Returns the value of the requested parameter
getPostedDocument()	Returns the content of Posted XML for this request as an XML Document
getRequestParamsAsXMLDocument()	Returns the content of a Request parameters as an XML Document
getRequestType()	Returns a string identifying the type of page request being made.
getSourceDocumentURI()	Returns a String representation of the requested document's URI
getStylesheetParameter(String)	Gets a stylesheet parameter by name
getStylesheetParameters()	Gets an enumeration of stylesheet parameter names
getStylesheetURI()	Returns the URI of the stylesheet to be used to process the result.
getUserAgent()	Returns a String identifier of the requesting program

Member Summary	
getWriter()	Returns a PrintWriter used for writing out the results of a page request
getXSQLConnection()	Gets the XSQLConnection Object being used for this request Might be null.
isIncludedRequest()	Returns true if this request is being included in another.
isOracleDriver()	Returns true if the current connection uses the Oracle JDBC Driver
printedErrorHeader()	Returns the state of whether an Error Header has been printed
requestProcessed()	Allows Page Request to Perform end-of-request processing
setConnectionName(String)	Sets the connection name to use for this request
setContentType(String)	Sets the content type of the resulting page
setIncludingRequest(XSQLPageRequest)	Sets the Including Page Request object for this request.
setPageEncoding(String)	Sets encoding of source XSQL page associated with this request.
setPageParam(String, String)	Sets a dynamic page parameter value.
setPostedDocument(Document)	Allows programmatic setting of the Posted Document
setPrintedErrorHeader(boolean)	Sets whether an Error Header has been printed
setStylesheetParameter(String, String)	Sets the value of a parameter to be passed to the associated stylesheet
setStylesheetURI(String)	Sets the URI of the stylesheet to be used to process the result.
translateURL(String)	Returns a string representing an absolute URL resolved relative to the base URI for this request.
useConnectionPooling()	Returns true if connection pooling is desired for this request
useHTMLErrors()	Returns true if HTML-formatted error messages are desired for this request

Methods

createNestedRequest(URL, Dictionary)

```
public XSQLPageRequest createNestedRequest(java.net.URL pageurl,
java.util.Dictionary params)
```

Returns an instance of a nested Request

getConnectionName()

```
public java.lang.String getConnectionName()
```

Returns the name of the connection being used for this request May be null if no connection set/in-use.

getErrorWriter()

`public java.io.PrintWriter getErrorWriter()`
Returns a PrintWriter to print out errors processing this request

getJDBCConnection()

`public java.sql.Connection getJDBCConnection()`
Gets the JDBC connection being used for this request (can be null)

getPageEncoding()

`public java.lang.String getPageEncoding()`
Returns encoding of source XSQL Page associated with this request

getParameter(String)

`public java.lang.String getParameter(java.lang.String name)`
Returns the value of the requested parameter

Parameters:

`name` - the name of the parameter

getPostedDocument()

`public oracle.xml.xsql.Document getPostedDocument()`
Returns the content of Posted XML for this request as an XML Document

getRequestParamsAsXMLDocument()

`public oracle.xml.xsql.Document getRequestParamsAsXMLDocument()`
Returns the content of a Request parameters as an XML Document

getRequestType()

`public java.lang.String getRequestType()`
Returns a string identifying the type of page request being made.

getSourceDocumentURI()

`public java.lang.String getSourceDocumentURI()`
Returns a String representation of the requested document's URI

getStylesheetParameter(String)

`public java.lang.String getStylesheetParameter(java.lang.String name)`
Gets a stylesheet parameter by name

getStylesheetParameters()

`public java.util.Enumeration getStylesheetParameters()`
Gets an enumeration of stylesheet parameter names

getStylesheetURI()

`public java.lang.String getStylesheetURI()`
Returns the URI of the stylesheet to be used to process the result.

getUserAgent()

`public java.lang.String getUserAgent()`
Returns a String identifier of the requesting program

getWriter()

`public java.io.PrintWriter getWriter()`
Returns a PrintWriter used for writing out the results of a page request

getXSQLConnection()

`public oracle.xml.xsql.XSQLConnection getXSQLConnection()`
Gets the XSQLConnection Object being used for this request Might be null.

isIncludedRequest()

`public boolean isIncludedRequest()`
Returns true if this request is being included in another.

isOracleDriver()

`public boolean isOracleDriver()`
Returns true if the current connection uses the Oracle JDBC Driver

printedErrorHeader()

`public boolean printedErrorHeader()`
Returns the state of whether an Error Header has been printed

requestProcessed()

`public void requestProcessed()`
Allows Page Request to Perform end-of-request processing

setConnectionName(String)

`public void setConnectionName(java.lang.String connName)`
Sets the connection name to use for this request

setContentType(String)

`public void setContentType(java.lang.String mimetype)`
Sets the content type of the resulting page

setIncludingRequest(XSQLPageRequest)

`public void setIncludingRequest(XSQLPageRequest includingEnv)`
Sets the Including Page Request object for this request.

setPageEncoding(String)

`public void setPageEncoding(java.lang.String enc)`
Sets encoding of source XSQL page associated with this request.

setPageParam(String, String)

`public void setPageParam(java.lang.String name, java.lang.String value)`
Sets a dynamic page parameter value.

setPostedDocument(Document)

`public void setPostedDocument(oracle.xml.xsql.Document doc)`
Allows programmatic setting of the Posted Document

setPrintedErrorHeader(boolean)

`public void setPrintedErrorHeader(boolean yes)`
Sets whether an Error Header has been printed

setStylesheetParameter(String, String)

```
public void setStylesheetParameter(java.lang.String name, java.lang.String
value)
```
Sets the value of a parameter to be passed to the associated stylesheet

setStylesheetURI(String)

```
public void setStylesheetURI(java.lang.String uri)
```

Sets the URI of the stylesheet to be used to process the result.

translateURL(String)

```
public java.lang.String translateURL(java.lang.String url)
```

Returns a string representing an absolute URL resolved relative to the base URI for this request.

useConnectionPooling()

```
public boolean useConnectionPooling()
```

Returns true if connection pooling is desired for this request

useHTMLErrors()

```
public boolean useHTMLErrors()
```

Returns true if HTML-formatted error messages are desired for this request

oracle.xml.xsql XSQLPageRequestImpl

Syntax

```
public abstract class XSQLPageRequestImpl extends java.lang.Object implements
XSQLPageRequest

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLPageRequestImpl
```

Direct Known Subclasses:

`XSQLServletPageRequest`

All Implemented Interfaces:

`XSQLPageRequest`

Description

Base implementation of the XSQLPageRequest interface that case be used to derive new kinds of page request implementations.

Member Summary

Constructors	
XSQLPageRequestImpl()	
XSQLPageRequestImpl(Hashtable)	
XSQLPageRequestImpl(String, Hashtable)	
Methods	
getConnectionName()	Returns the name of the connection being used for this request May be null if no connection set/in-use.
getErrorWriter()	
getJDBCConnection()	Gets the JDBC connection being used for this request (can be null)
getPageEncoding()	Returns encoding of source XSQL Page associated with this request

Member Summary

getParameter(String)	
getPostedDocument()	
getRequestParamsAsXMLDocument()	
getSourceDocumentURI()	
getStylesheetParameter(String)	Gets a stylesheet parameter by name
getStylesheetParameters()	Gets an enumeration of stylesheet parameter names
getStylesheetURI()	
getUserAgent()	
getWriter()	
getXSQLConnection()	Gets the XSQLConnection Object being used for this request Might be null.
isIncludedRequest()	Returns true if this request is being included in another.
isOracleDriver()	Returns true if the current connection uses the Oracle JDBC Driver
printedErrorHeader()	
requestProcessed()	Allows Page Request to Perform end-of-request processing
setConnectionName(String)	Sets the connection being used for this request (can be null)
setContentType(String)	
setIncludingRequest(XSQLPageRequest)	Sets the Including Page Request object for this request.
setPageEncoding(String)	Associates an XSQL Page with the request
setPageParam(String, String)	Sets a dynamic page parameter value.
setPostedDocument(Document)	
setPrintedErrorHeader(boolean)	
setStylesheetParameter(String, String)	
setStylesheetURI(String)	
translateURL(String)	
useConnectionPooling()	
useHTMLErrors()	

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface XSQLPageRequest

createNestedRequest(URL, Dictionary), getRequestType()

Constructors

XSQLPageRequestImpl()

```
public  XSQLPageRequestImpl()
```

XSQLPageRequestImpl(Hashtable)

```
public  XSQLPageRequestImpl(java.util.Hashtable parameters)
```

XSQLPageRequestImpl(String, Hashtable)

```
public  XSQLPageRequestImpl(java.lang.String pageurl, java.util.Hashtable
parameters)
```

Methods

getConnectionName()

```
public java.lang.String getConnectionName()
```

Returns the name of the connection being used for this request May be null if no connection set/in-use.

Specified By:

getConnectionName() in interface XSQLPageRequest

getErrorWriter()

```
public java.io.PrintWriter getErrorWriter()
```

Specified By:

getErrorWriter() in interface XSQLPageRequest

getJDBCConnection()

`public java.sql.Connection getJDBCConnection()`
Gets the JDBC connection being used for this request (can be null)

Specified By:

getJDBCConnection() in interface XSQLPageRequest

getPageEncoding()

`public java.lang.String getPageEncoding()`
Returns encoding of source XSQL Page associated with this request

Specified By:

getPageEncoding() in interface XSQLPageRequest

getParameter(String)

`public java.lang.String getParameter(java.lang.String name)`

Specified By:

getParameter(String) in interface XSQLPageRequest

getPostedDocument()

`public oracle.xml.xsql.Document getPostedDocument()`

Specified By:

getPostedDocument() in interface XSQLPageRequest

getRequestParamsAsXMLDocument()

`public oracle.xml.xsql.Document getRequestParamsAsXMLDocument()`

Specified By:

getRequestParamsAsXMLDocument() in interface XSQLPageRequest

getSourceDocumentURI()

`public java.lang.String getSourceDocumentURI()`

Specified By:

getSourceDocumentURI() in interface XSQLPageRequest

getStylesheetParameter(String)

```
public java.lang.String getStylesheetParameter(java.lang.String name)
```

Gets a stylesheet parameter by name

Specified By:

getStylesheetParameter(String) in interface XSQLPageRequest

getStylesheetParameters()

```
public java.util.Enumeration getStylesheetParameters()
```

Gets an enumeration of stylesheet parameter names

Specified By:

getStylesheetParameters() in interface XSQLPageRequest

getStylesheetURI()

```
public java.lang.String getStylesheetURI()
```

Specified By:

getStylesheetURI() in interface XSQLPageRequest

getUserAgent()

```
public java.lang.String getUserAgent()
```

Specified By:

getUserAgent() in interface XSQLPageRequest

getWriter()

```
public java.io.PrintWriter getWriter()
```

Specified By:

getWriter() in interface XSQLPageRequest

getXSQLConnection()

`public oracle.xml.xsql.XSQLConnection getXSQLConnection()`

Gets the XSQLConnection Object being used for this request Might be null.

Specified By:

getXSQLConnection() in interface XSQLPageRequest

isIncludedRequest()

`public boolean isIncludedRequest()`

Returns true if this request is being included in another.

Specified By:

isIncludedRequest() in interface XSQLPageRequest

isOracleDriver()

`public boolean isOracleDriver()`

Returns true if the current connection uses the Oracle JDBC Driver

Specified By:

isOracleDriver() in interface XSQLPageRequest

printedErrorHeader()

`public boolean printedErrorHeader()`

Specified By:

printedErrorHeader() in interface XSQLPageRequest

requestProcessed()

`public void requestProcessed()`

Allows Page Request to Perform end-of-request processing

Specified By:

requestProcessed() in interface XSQLPageRequest

setConnectionName(String)

`public void setConnectionName(java.lang.String connName)`

Sets the connection being used for this request (can be null)

Specified By:
setConnectionName(String) in interface XSQLPageRequest

setContentType(String)

`public void setContentType(java.lang.String mimetype)`

Specified By:
setContentType(String) in interface XSQLPageRequest

setIncludingRequest(XSQLPageRequest)

`public void setIncludingRequest(XSQLPageRequest includingEnv)`

Sets the Including Page Request object for this request.

Specified By:
setIncludingRequest(XSQLPageRequest) in interface XSQLPageRequest

setPageEncoding(String)

`public void setPageEncoding(java.lang.String enc)`

Associates an XSQL Page with the request

Specified By:
setPageEncoding(String) in interface XSQLPageRequest

setPageParam(String, String)

`public void setPageParam(java.lang.String name, java.lang.String value)`

Sets a dynamic page parameter value.

Specified By:
setPageParam(String, String) in interface XSQLPageRequest

setPostedDocument(Document)

`public void setPostedDocument(oracle.xml.xsql.Document doc)`

Specified By:

setPostedDocument(Document) in interface XSQLPageRequest

setPrintedErrorHeader(boolean)

```
public void setPrintedErrorHeader(boolean yes)
```

Specified By:

setPrintedErrorHeader(boolean) in interface XSQLPageRequest

setStylesheetParameter(String, String)

```
public void setStylesheetParameter(java.lang.String name, java.lang.String
value)
```

Specified By:

setStylesheetParameter(String, String) in interface XSQLPageRequest

setStylesheetURI(String)

```
public void setStylesheetURI(java.lang.String uri)
```

Specified By:

setStylesheetURI(String) in interface XSQLPageRequest

translateURL(String)

```
public java.lang.String translateURL(java.lang.String filename)
```

Specified By:

translateURL(String) in interface XSQLPageRequest

useConnectionPooling()

```
public boolean useConnectionPooling()
```

Specified By:

useConnectionPooling() in interface XSQLPageRequest

useHTMLErrors()

```
public boolean useHTMLErrors()
```

Specified By:

useHTMLErrors() in interface XSQLPageRequest

oracle.xml.xsql
XSQLParserHelper

Syntax

```
public final class XSQLParserHelper extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLParserHelper
```

Description

Common XML Parsing Routines

Member Summary

Constructors

XSQLParserHelper()

Methods

newDocument()

parse(InputStream, URL, PrintWriter)

parse(Reader, PrintWriter)

parse(URL, PrintWriter)

parseFromString(StringBuffer, PrintWriter)

parseFromString(String, PrintWriter)

print(Document, PrintWriter)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLParserHelper()

```
public  XSQLParserHelper()
```

Methods

newDocument()

```
public static oracle.xml.xsql.Document newDocument()
```

parse(InputStream, URL, PrintWriter)

```
public static oracle.xml.xsql.Document parse(java.io.InputStream is,
java.net.URL baseUrl, java.io.PrintWriter errorWriter)
```

parse(Reader, PrintWriter)

```
public static oracle.xml.xsql.Document parse(java.io.Reader r,
java.io.PrintWriter errorWriter)
```

parse(URL, PrintWriter)

```
public static oracle.xml.xsql.Document parse(java.net.URL url,
java.io.PrintWriter errorWriter)
```

parseFromString(StringBuffer, PrintWriter)

```
public static oracle.xml.xsql.Document parseFromString(java.lang.StringBuffer
xmlString, java.io.PrintWriter errorWriter)
```

parseFromString(String, PrintWriter)

```
public static oracle.xml.xsql.Document parseFromString(java.lang.String
xmlString, java.io.PrintWriter errorWriter)
```

print(Document, PrintWriter)

```
public static void print(oracle.xml.xsql.Document d, java.io.PrintWriter out)
```

oracle.xml.xsql
XSQLRequest

Syntax

```
public class XSQLRequest extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLRequest
```

Description

Programmatically process a request for an XSQL Page.

Member Summary

Constructors	
XSQLRequest(String)	Create a Request for an XSQL Page
XSQLRequest(String, XSQLPageRequest)	Create a Request for an XSQL Page
XSQLRequest(URL)	Create a Request for an XSQL Page
XSQLRequest(URL, XSQLPageRequest)	Create a Request for an XSQL Page
Methods	
process()	Process the request, writing output/errors to System.out/System.err
process(Dictionary)	Process the request, writing output/errors to System.out/System.err
process(Dictionary, PrintWriter, PrintWriter)	Process the request, writing output/errors to respective PrintWriters
process(PrintWriter, PrintWriter)	Process the request, writing output/errors to respective PrintWriters
processToXML()	Process the request, writing output/errors to System.out/System.err
processToXML(Dictionary)	Process the request, writing output/errors to System.out/System.err
processToXML(Dictionary, PrintWriter)	Process the request, writing output/errors to respective PrintWriters
processToXML(PrintWriter)	Process the request, writing errors to respective PrintWriters

Member Summary	
setPostedDocument(Document)	Programmatically set an XML Document to be treated the same as if it were posted as part of the request.

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLRequest(String)

```
public  XSQLRequest(java.lang.String url)
```

Create a Request for an XSQL Page

Parameters:

`url` - String representation of an URL to an XSQL Page

XSQLRequest(String, XSQLPageRequest)

```
public  XSQLRequest(java.lang.String url, XSQLPageRequest env)
```

Create a Request for an XSQL Page

Parameters:

`url` - String representation of an URL to an XSQL Page

`env` - Calling XSQLPageRequest environment

XSQLRequest(URL)

```
public  XSQLRequest(java.net.URL url)
```

Create a Request for an XSQL Page

Parameters:

`url` - URL to an XSQL Page

XSQLRequest(URL, XSQLPageRequest)

`public XSQLRequest(java.net.URL url, XSQLPageRequest env)`
Create a Request for an XSQL Page

Parameters:

`url` - URL to an XSQL Page

`env` - Calling XSQLPageRequest environment

Methods

process()

`public void process()`
Process the request, writing output/errors to System.out/System.err

process(Dictionary)

`public void process(java.util.Dictionary params)`
Process the request, writing output/errors to System.out/System.err

Parameters:

`params` - Dictionary (e.g. Hashtable) with XSQL Page parameters

process(Dictionary, PrintWriter, PrintWriter)

```
public void process(java.util.Dictionary params, java.io.PrintWriter out,
java.io.PrintWriter err)
```
Process the request, writing output/errors to respective PrintWriters

Parameters:

`params` - Dictionary (e.g. Hashtable) with XSQL Page parameters

`out` - PrintWriter to use to write the resulting page results

`err` - PrintWriter to use to write the resulting page errors

process(PrintWriter, PrintWriter)

`public void process(java.io.PrintWriter out, java.io.PrintWriter err)`
Process the request, writing output/errors to respective PrintWriters

Parameters:

`out` - PrintWriter to use to write the resulting page results

`err` - PrintWriter to use to write the resulting page errors

processToXML()

`public org.w3c.dom.Document processToXML()`
Process the request, writing output / errors to System.out / System.err

processToXML(Dictionary)

`public org.w3c.dom.Document processToXML(java.util.Dictionary params)`
Process the request, writing output / errors to System.out / System.err

Parameters:

`params` - Dictionary (e.g. Hashtable) with XSQL Page parameters

processToXML(Dictionary, PrintWriter)

```
public org.w3c.dom.Document processToXML(java.util.Dictionary params,
java.io.PrintWriter err)
```
Process the request, writing output / errors to respective PrintWriters

Parameters:

`params` - Dictionary (e.g. Hashtable) with XSQL Page parameters

`err` - PrintWriter to use to write the resulting page errors

processToXML(PrintWriter)

`public org.w3c.dom.Document processToXML(java.io.PrintWriter err)`
Process the request, writing errors to respective PrintWriters

Parameters:

`err` - PrintWriter to use to write the resulting page errors

setPostedDocument(Document)

`public void setPostedDocument(org.w3c.dom.Document postedDoc)`
Programmatically set an XML Document to be treated the same as if it were posted as part of the request.

Parameters:

`postedDoc` - DOM Document

oracle.xml.xsql XSQLServlet

Syntax

```
public final class XSQLServlet
```

`oracle.xml.xsql.XSQLServlet`

Description

Servlet to enable HTTP GET-ing of and POST-ing to XSQL Pages

Member Summary

Constructors

XSQLServlet()

Methods

doGet(HttpServletRequest, HttpServletResponse)

doPost(HttpServletRequest, HttpServletResponse)

getServletInfo()

init(ServletConfig)

inJServ()

Constructors

XSQLServlet()

```
public  XSQLServlet()
```

Methods

doGet(HttpServletRequest, HttpServletResponse)

```
public void doGet(oracle.xml.xsql.HttpServletRequest request,
oracle.xml.xsql.HttpServletResponse response)
```

doPost(HttpServletRequest, HttpServletResponse)

```
public void doPost(oracle.xml.xsql.HttpServletRequest request,
oracle.xml.xsql.HttpServletResponse response)
```

getServletInfo()

```
public java.lang.String getServletInfo()
```

init(ServletConfig)

```
public void init(oracle.xml.xsql.ServletConfig config)
```

inJServ()

```
public static boolean inJServ()
```

oracle.xml.xsql
XSQLServletPageRequest

Syntax

```
public final class XSQLServletPageRequest extends XSQLPageRequestImpl
```

```
java.lang.Object
  |
  +--XSQLPageRequestImpl
        |
        +--oracle.xml.xsql.XSQLServletPageRequest
```

All Implemented Interfaces:

XSQLPageRequest

Description

Implementation of XSQLPageRequest for Servlet-based XSQL Page requests.

Member Summary

Constructors

XSQLServletPageRequest(HttpServletRequest, HttpServletResponse)	

Methods

createNestedRequest(URL, Dictionary)	Returns an instance of a nested Request
getCookie(String)	
getHttpServletRequest()	Get the HttpServletRequest that initiated this XSQL Page Request.
getHttpServletResponse()	Get the HttpServletResponse that is associated with this XSQL Page Request
getParameter(String)	Use HTTP Parameters as the source of parameters instead
getPostedDocument()	
getRequestParamsAsXMLDocument()	
getRequestType()	

Member Summary

getUserAgent()

setContentType(String)

setPageEncoding(String)

translateURL(String)

useHTMLErrors()

Inherited Member Summary

Methods inherited from class XSQLPageRequestImpl

getConnectionName(), getErrorWriter(), getJDBCConnection(), getPageEncoding(), getSourceDocumentURI(), getStylesheetParameter(String), getStylesheetParameters(), getStylesheetURI(), getWriter(), getXSQLConnection(), isIncludedRequest(), isOracleDriver(), printedErrorHeader(), requestProcessed(), setConnectionName(String), setIncludingRequest(XSQLPageRequest), setPageParam(String, String), setPostedDocument(Document), setPrintedErrorHeader(boolean), setStylesheetParameter(String, String), setStylesheetURI(String), useConnectionPooling()

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface XSQLPageRequest

getConnectionName(), getErrorWriter(), getJDBCConnection(), getPageEncoding(), getSourceDocumentURI(), getStylesheetParameter(String), getStylesheetParameters(), getStylesheetURI(), getWriter(), getXSQLConnection(), isIncludedRequest(), isOracleDriver(), printedErrorHeader(), requestProcessed(), setConnectionName(String), setIncludingRequest(XSQLPageRequest), setPageParam(String, String), setPostedDocument(Document), setPrintedErrorHeader(boolean), setStylesheetParameter(String, String), setStylesheetURI(String), useConnectionPooling()

Constructors

XSQLServletPageRequest(HttpServletRequest, HttpServletResponse)

```
public  XSQLServletPageRequest(oracle.xml.xsql.HttpServletRequest req,
oracle.xml.xsql.HttpServletResponse resp)
```

Methods

createNestedRequest(URL, Dictionary)

```
public XSQLPageRequest createNestedRequest(java.net.URL pageurl,
```

`java.util.Dictionary params)`
Returns an instance of a nested Request

getCookie(String)

`public java.lang.String getCookie(java.lang.String name)`

getHttpServletRequest()

`public oracle.xml.xsql.HttpServletRequest getHttpServletRequest()`
Get the HttpServletRequest that initiated this XSQL Page Request.

getHttpServletResponse()

`public oracle.xml.xsql.HttpServletResponse getHttpServletResponse()`
Get the HttpServletResponse that is associated with this XSQL Page Request

getParameter(String)

`public java.lang.String getParameter(java.lang.String name)`
Use HTTP Parameters as the source of parameters instead

Overrides:

getParameter(String) in class XSQLPageRequestImpl

getPostedDocument()

`public oracle.xml.xsql.Document getPostedDocument()`

Overrides:

getPostedDocument() in class XSQLPageRequestImpl

getRequestParamsAsXMLDocument()

`public oracle.xml.xsql.Document getRequestParamsAsXMLDocument()`

Overrides:

getRequestParamsAsXMLDocument() in class XSQLPageRequestImpl

getRequestType()

`public java.lang.String getRequestType()`

getUserAgent()

```
public java.lang.String getUserAgent()
```

Overrides:

getUserAgent() in class XSQLPageRequestImpl

setContentType(String)

```
public void setContentType(java.lang.String mimetype)
```

Overrides:

setContentType(String) in class XSQLPageRequestImpl

setPageEncoding(String)

```
public void setPageEncoding(java.lang.String enc)
```

Overrides:

setPageEncoding(String) in class XSQLPageRequestImpl

translateURL(String)

```
public java.lang.String translateURL(java.lang.String path)
```

Overrides:

translateURL(String) in class XSQLPageRequestImpl

useHTMLErrors()

```
public boolean useHTMLErrors()
```

Overrides:

useHTMLErrors() in class XSQLPageRequestImpl

oracle.xml.xsql XSQLStylesheetProcessor

Syntax

```
public final class XSQLStylesheetProcessor extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLStylesheetProcessor
```

Description

XSLT Stylesheet Processing Engine

Member Summary

Constructors

XSQLStylesheetProcessor()

Methods

processToDocument(Document, String, XSQLPageRequest)

processToWriter(Document, String, XSQLPageRequest)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLStylesheetProcessor()

```
public  XSQLStylesheetProcessor()
```

Methods

processToDocument(Document, String, XSQLPageRequest)

```
public static oracle.xml.xsql.Document
processToDocument(oracle.xml.xsql.Document xml, java.lang.String xslURI,
XSQLPageRequest env)
```

processToWriter(Document, String, XSQLPageRequest)

```
public static void processToWriter(oracle.xml.xsql.Document xml,
java.lang.String xslURI, XSQLPageRequest env)
```

oracle.xml.xsql XSQLUtil

Syntax

```
public final class XSQLUtil extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.xsql.XSQLUtil
```

Description

Member Summary

Constructors

XSQLUtil()

Methods

DictionaryOfParamsAsXMLDocument (Dictionary)

safeURLAsString(URL)

select(Document, String)

select(Element, String)

select(XMLDocument, String)

select(XMLElement, String)

selectFirst(Document, String)

selectFirst(Element, String)

selectFirst(XMLDocument, String)

selectFirst(XMLElement, String)

stringParamValue(Object)

translate(String, String)

translate(URL, String)

valueOf(Element, String)

Member Summary

valueOf(Node, String)

valueOf(XMLElement, String)

valueOf(XMLNode, String)

XL(String, String)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSQLUtil()

```
public  XSQLUtil()
```

Methods

DictionaryOfParamsAsXMLDocument(Dictionary)

```
public static oracle.xml.xsql.XMLDocument
DictionaryOfParamsAsXMLDocument(java.util.Dictionary dict)
```

safeURLAsString(URL)

```
public static java.lang.String safeURLAsString(java.net.URL u)
```

select(Document, String)

```
public static oracle.xml.xsql.NodeList select(oracle.xml.xsql.Document d,
java.lang.String pattern)
```

select(Element, String)

```
public static oracle.xml.xsql.NodeList select(oracle.xml.xsql.Element n,
java.lang.String pattern)
```

select(XMLDocument, String)

```
public static oracle.xml.xsql.NodeList select(oracle.xml.xsql.XMLDocument d,
java.lang.String pattern)
```

select(XMLElement, String)

```
public static oracle.xml.xsql.NodeList select(oracle.xml.xsql.XMLElement n,
java.lang.String pattern)
```

selectFirst(Document, String)

```
public static oracle.xml.xsql.Node selectFirst(oracle.xml.xsql.Document d,
java.lang.String pattern)
```

selectFirst(Element, String)

```
public static oracle.xml.xsql.Node selectFirst(oracle.xml.xsql.Element n,
java.lang.String pattern)
```

selectFirst(XMLDocument, String)

```
public static oracle.xml.xsql.Node selectFirst(oracle.xml.xsql.XMLDocument d,
java.lang.String pattern)
```

selectFirst(XMLElement, String)

```
public static oracle.xml.xsql.Node selectFirst(oracle.xml.xsql.XMLElement n,
java.lang.String pattern)
```

stringParamValue(Object)

```
public static java.lang.String stringParamValue(java.lang.Object val)
```

translate(String, String)

```
public static java.lang.String translate(java.lang.String u, java.lang.String
path)
```

translate(URL, String)

```
public static java.lang.String translate(java.net.URL u, java.lang.String path)
```

valueOf(Element, String)

```
public static java.lang.String valueOf(oracle.xml.xsql.Element n,
java.lang.String pattern)
```

valueOf(Node, String)

```
public static java.lang.String valueOf(oracle.xml.xsql.Node n, java.lang.String
pattern)
```

valueOf(XMLElement, String)

```
public static java.lang.String valueOf(oracle.xml.xsql.XMLElement n,
java.lang.String pattern)
```

valueOf(XMLNode, String)

```
public static java.lang.String valueOf(oracle.xml.xsql.XMLNode n,
java.lang.String pattern)
```

XL(String, String)

```
public static java.lang.String XL(java.lang.String s, java.lang.String enc)
```

4

Transviewer Beans

oracle.xml.async Package Description

Class Summary

Interfaces	
DOMBuilderErrorListener	This interface must be implemented in order to receive notifications when error is found during parsing.
DOMBuilderListener	This interface must be implemented in order to receive notifications about events during the asyncronous parsing.
XSLTransformerErrorListener	This interface must be implemented in order to receive notifications about error events during the asyncronous transformation.
XSLTransformerListener	This interface must be implemented in order to receive notifications about events during the asyncronous transformation.
Classes	
DOMBuilder	This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.
DOMBuilderBeanInfo	
DOMBuilderErrorEvent	This class defines the error event which is sent when parse exception occurs.
DOMBuilderEvent	The event object that DOMBuilder uses to notify all registered listeners about parse events.
ResourceManager	Simple semaphore that maintains access to fixed number of logical resources.
XSLTransformer	Applies XSL transformation in a background thread.
XSLTransformerBeanInfo	
XSLTransformerErrorEvent	The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.

oracle.xml.async
DOMBuilder

Syntax

```
public class DOMBuilder extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.DOMBuilderConstants, java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.DOMBuilder
```

All Implemented Interfaces:

`oracle.xml.async.DOMBuilderConstants, java.lang.Runnable, java.io.Serializable`

Description

This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation. to parse a XML

d
o

cument and build a DOM tree. The parsing is done in a separate thread and DOMBuilderListener interface must be used for notification when the tree is built.

Member Summary

Constructors	
DOMBuilder()	Creates a new parser object.
Creates a new parser object.	Creates a new parser object with a given id.
Methods	
addDOMBuilderErrorListener(DOMBuilderErrorListener)	Adds DOMBuilderErrorListener
addDOMBuilderListener(DOMBuilderListener)	Adds DOMBuilderListener
	Get the DTD
getDocument()	Gets the document
getId()	Returns the parser object id.
getReleaseVersion()	Returns the release version of the Oracle XML Parser
	Gets the document
getValidationMode()	Returns the validation mode
parse(InputSource)	Parses the XML from given input source
	Parses the XML from given input stream.
parse(Reader)	Parses the XML from given input stream.
parse(String)	Parses the XML from the URL indicated
parse(URL)	Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.
parseDTD(InputSource, String)	Parses the XML External DTD from given input source

parseDTD(InputStrear

Member Summary

	Parses the XML External DTD from given input stream.
parseDTD(String, String)	Parses the XML External DTD from the URL indicated
parseDTD(URL, String)	Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.
removeDOMBuilderErrorListener(DOMBuilderErrorListener)	Remove DOMBuilderErrorListener
removeDOMBuilderListener(DOMBuilderListener)	Remove DOMBuilderListener
run()	This method runs in a thread
setBaseURL(URL)	Set the base URL for loading external enitites and DTDs.
setDebugMode(boolean)	Sets a flag to turn on debug information in the document
setDoctype(DTD)	Set the DTD
setErrorStream(OutputStream)	Creates an output stream for the output of errors and warnings.
setErrorStream(OutputStream, String)	Creates an output stream for the output of errors and warnings.
setErrorStream(PrintWriter)	Creates an output stream for the output of errors and warnings.
setNodeFactory(NodeFactory)	Set the node factory.
setPreserveWhitespace(boolean)	Set the white space preserving mode
setValidationMode(boolean)	Set the validation mode
showWarnings(boolean)	Switch to determine whether to print warnings

Inherited Member Summary

Fields inherited from interface oracle.xml.async.DOMBuilderConstants

DOM_GEN_ENDED, DOM_GEN_ERROR, DOM_GEN_STARTED, DOM_METHOD1, DOM_METHOD10, DOM_METHOD2, DOM_METHOD3, DOM_METHOD4, DOM_METHOD5, DOM_METHOD6, DOM_METHOD7, DOM_METHOD8, DOM_METHOD9

Inherited Membe

Methods inherited

Inherited Member Summary

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

DOMBuilder()

```
public  DOMBuilder()
```

Creates a new parser object.

DOMBuilder(int)

```
public  DOMBuilder(int id)
```

Creates a new parser object with a given id.

Methods

addDOMBuilderErrorListener(DOMBuilderErrorListener)

```
public void addDOMBuilderErrorListener(DOMBuilderErrorListener p0)
```

Adds DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to add

addDOMBuilderListener(DOMBuilderListener)

```
public void addDOMBuilderListener(DOMBuilderListener p0)
```

Adds DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to add

getDoctype()

`public synchronized oracle.xml.async.DTD getDoctype()`
Get the DTD

Returns:

The `DTD`

getDocument()

`public synchronized oracle.xml.async.XMLDocument getDocument()`
Gets the document

Returns:

The document being parsed

getId()

`public int getId()`
Returns the parser object id.

getReleaseVersion()

`public synchronized java.lang.String getReleaseVersion()`
Returns the release version of the Oracle XML Parser

Returns:

the release version string

getResult()

`public synchronized org.w3c.dom.Document getResult()`

Gets the document

Returns:

The document being parsed

getValidationMode()

`public synchronized boolean getValidationMode()`
Returns the validation mode

Returns:

`true` if the XML parser is validating `false` if not

parse(InputSource)

`public final synchronized void parse(org.xml.sax.InputSource in)`
Parses the XML from given input source

Parameters:

`in` - the `org.xml.sax.InputSouce` to parse

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(InputStream)

`public final synchronized void parse(java.io.InputStream in)`
Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(Reader)

```
public final synchronized void parse(java.io.Reader r)
```

Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(String)

```
public final synchronized void parse(java.lang.String in)
```

P
a

rses the XML from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(URL)

```
public final synchronized void parse(java.net.URL url)
```

Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputSource, String)

```
public final synchronized void parseDTD(org.xml.sax.InputSource in,
java.lang.String rootName)
```

Parses the XML External DTD from given input source

P
a

rameters:

`in` - the `org.xml.sax.InputSouce` to parse

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputStream, String)

```
public final synchronized void parseDTD(java.io.InputStream in, java.lang.String
rootName)
```

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(Reader, String)

```
public final synchronized void parseDTD(java.io.Reader r, java.lang.String
rootName)
```

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(String, String)

```
public final synchronized void parseDTD(java.lang.String in, java.lang.String
rootName)
```

Parses the XML External DTD from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(URL, String)

```
public final synchronized void parseDTD(java.net.URL url, java.lang.String
rootName)
```

Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

removeDOMBuilderErrorListener(DOMBuilderErrorListener)

```
public synchronized void removeDOMBuilderErrorListener(DOMBuilderErrorListener
p0)
```

Remove DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to remove

BuilderListener(DOMBuilderListener)

`public synchronized void removeDOMBuilderListener(DOMBuilderListener p0)`
Remove DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to remove

run()

`public void run()`
This method runs in a thread

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setBaseURL(URL)

`public synchronized void setBaseURL(java.net.URL url)`
Set the base URL for loading external enitites and DTDs. This method should to be called if the parse(InputStream) is used to parse the XML Document

Parameters:

`url` - The base URL

setDebugMode(boolean)

`public void setDebugMode(boolean flag)`
Sets a flag to turn on debug information in the document

Parameters:

`flag` - determines whether debug info is stored

set-Doc-type(DTD)

```
public synchronized void setDoctype(oracle.xml.async.DTD dtd)
```
Set the DTD

Parameters:

`dtd` - `DTD` to set and used while parsing

setErrorStream(OutputStream)

```
public final synchronized void setErrorStream(java.io.OutputStream out)
```
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The output stream to use for errors and warnings

setErrorStream(OutputStream, String)

```
public final synchronized void setErrorStream(java.io.OutputStream out,
java.lang.String enc)
```
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings. Additionally, an .exception is thrown if the encoding specified is unsupported.

Parameters:

`out` - The output stream to use for errors and warnings

`enc` - the encoding to use

Throws:

`IOException` - if an unsupported encoding is specified

Stream(PrintWriter)

`public final synchronized void setErrorStream(java.io.PrintWriter out)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The `PrintWriter` to use for errors and warnings

setNodeFactory(NodeFactory)

`public synchronized void setNodeFactory(oracle.xml.async.NodeFactory factory)`
Set the node factory. Applications can extend the NodeFactory and register it through this method. The parser will then use the user supplied NodeFactory to create nodes of the DOM tree.

Parameters:

`factory` - The `NodeFactory` to set

Throws:

`XMLParseException` - if an invalid factory is set

See Also:

`NodeFactory`

setPreserveWhitespace(boolean)

`public synchronized void setPreserveWhitespace(boolean flag)`
Set the white space preserving mode

Parameters:

`flag` - preserving mode

setValidation-Mode(boolean)

`public synchronized void setValidationMode(boolean yes)`

Set the validation mode

Parameters:

`yes` - determines whether the XML parser should be validating

showWarnings(boolean)

`public synchronized void showWarnings(boolean yes)`

Switch to determine whether to print warnings

Parameters:

`yes` - determines whether warnings should be shown

oracle.xml.async
DOMBuilderBeanInfo

Syntax

```
public class DOMBuilderBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.DOMBuilderBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

DOMBuilderBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Inherited Member
Methods inherited

Inherited Member Summary

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

DOMBuilderBeanInfo()

```
public  DOMBuilderBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async DOMBuilderErrorEvent

Syntax

```
public class DOMBuilderErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

This class defines the error event which is sent when parse exception occurs.

Member Summary

Constructors	
DOMBuilderErrorEvent(Object, Exception)	Constructor for DOMBuilderErrorEvent.
Methods	
getException()	Gets the Exception
getMessage()	Returns the error message generated by the parser

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderErrorEvent(Object, Exception)

```
public  DOMBuilderErrorEvent(java.lang.Object p0, java.lang.Exception e)
```

Constructor for DOMBuilderErrorEvent.

Parameters:

`p0` - the `Object` that create this object

`e` - the `Exception` raised.

Methods

getException()

`public java.lang.Exception getException()`

Gets the Exception

Returns:

The Exception beind raised

getMessage()

`public java.lang.String getMessage()`

Returns the error message generated by the parser

Returns:

the error message string

oracle.xml.async
DOMBuilderErrorListener

Syntax

`public interface DOMBuilderErrorListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications when error is found during parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderErrorListener method.

Member Summary

Methods	
domBuilderErrorCalled(DOMBuilderErrorEvent)	

Methods

domBuilderErrorCalled(DOMBuilderErrorEvent)

`public void domBuilderErrorCalled(DOMBuilderErrorEvent p0)`

Parameters:

`p0` -- The DOMBuilderErrorEvent object produced by the DOMBuilder as result of parsing error

oracle.xml.async
DOMBuilderEvent

Syntax

```
public class DOMBuilderEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that DOMBuilder uses to notify all registered listeners about parse events.

Member Summary	
Constructors	
DOMBuilderEvent(Object, int)	
Methods	
getID()	Returns the DOMBuilder object unique id.

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderEvent(Object, int)

```
public  DOMBuilderEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the DOMBuilder object unique id. Can be used to identify which instance of the DOMBuilder generated this event in case when multiple instances of DOMBuilder work in the background.

Returns:

The event id

oracle.xml.async
DOMBuilderListener

Syntax

```
public interface DOMBuilderListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

Description

This interface must be implemented in order to receive notifications about events during the asyncronous parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderListener method.

Member Summary

Methods	
domBuilderError(DOMBuilderEvent)	This method is called when parse error occur.
domBuilderOver(DOMBuilderEvent)	This method is called when the parse is complete
domBuilderStarted(DOMBuilderEvent)	This method is called when parse starts

Methods

domBuilderError(DOMBuilderEvent)

```
public void domBuilderError(DOMBuilderEvent p0)
```

This method is called when parse error occur.

Parameters:

`p0` - - The DOMBuilderEvent object produced by the DOMBuilder

domBuilderOver(DOMBuilderEvent)

`public void domBuilderOver(DOMBuilderEvent p0)`

This method is called when the parse is complete

Parameters:

`p0` - - The DOMBuilderEvent object produced by the DOMBuilder

domBuilderStarted(DOMBuilderEvent)

`public void domBuilderStarted(DOMBuilderEvent p0)`

This method is called when parse starts

Parameters:

`p0` - - The DOMBuilderEvent object produced by the DOMBuilder

oracle.xml.async
ResourceManager

Syntax

```
public class ResourceManager extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.async.ResourceManager
```

Description

Simple semaphore that maintains access to fixed number of logical resources.

Member Summary

Constructors	
ResourceManager(int)	The ResourceManager constructor
Methods	
activeFound()	Returns how many resources are avialable.
getResource()	If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns.
releaseResource()	Releases a resource.
sleep(int)	Let's you use Thread.sleep() without try/catch

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ResourceManager(int)

`public  ResourceManager(int i)`
The ResourceManager constructor

Parameters:

`i` -- the number of resources to manage

Methods

activeFound()

`public boolean activeFound()`
Returns how many resources are avialable.

getResource()

`public synchronized void getResource()`
If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns. Othervise, waits until resource is released.

releaseResource()

`public void releaseResource()`
Releases a resource. When this method is called, the number of resources avialable is increased by one.

sleep(int)

`public void sleep(int i)`
Let's you use Thread.sleep() without try/catch

oracle.xml.async
XSLTransformer

Syntax

```
public class XSLTransformer extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.XSLTransformerConstants,
java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.XSLTransformer
```

All Implemented Interfaces:

```
java.lang.Runnable, java.io.Serializable,
oracle.xml.async.XSLTransformerConstants
```

Description

Applies XSL transformation in a background thread.

Member Summary

Constructors	
XSLTransformer()	XSLTransformet constructor
XSLTransformer(int)	XSLTransformer constructor
Methods	
addXSLTransformerErrorListener(XSLTransformerErrorListener)	Adds an error event listener
addXSLTransformerListener(XSLTransformerListener)	Adds a listener
	Returns the unique XSLTransformer id

Member Summar

getResult()

Member Summary

processXSL(XSLStylesheet, InputStream, URL)	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, Reader, URL)	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, URL, URL)	Initiate XSL Transformation in the background.
	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, XMLDocument, OutputStream)	Initiate XSL Transformation in the background.
removeDOMTransformerErrorListener (XSLTransformerErrorListener)	Renmoves an error event listener
removeXSLTransformerListener(XSLTransformerListener)	Removes a listener
run()	
	Sets the error stream used by the XSL processor
showWarnings(boolean)	Sets the showWarnings flage used by the XSL processor

Inherited Member Summary

Fields inherited from interface oracle.xml.async.XSLTransformerConstants

XSL_GEN_ENDED, XSL_GEN_ERROR, XSL_GEN_STARTED

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSLTransformer()

`public  XSLTransformer()`
XSLTransformet constructor

XSLTransformer(int)

`public  XSLTransformer(int id)`
XSLTransformer constructor

Parameters:

`id` -- A unique integer that can be used to identify the XSLTransformer instance during event processing

Methods

addXSLTransformerErrorListener(XSLTransformerErrorListener)

`public void addXSLTransformerErrorListener(XSLTransformerErrorListener p0)`
Adds an error event listener

Parameters:

`p0` -- XSLTransformerErrorListener

addXSLTransformerListener(XSLTransformerListener)

`public void addXSLTransformerListener(XSLTransformerListener p0)`
Adds a listener

Parameters:

`p0` -- XSLTransformerListener

getId()

`public int getId()`

Returns the unique XSLTransformer id

Returns:

The id

getResult()

`public synchronized oracle.xml.async.DocumentFragment getResult()`

Returns the document fragment for the resulting document. Call this method only after receiving notification that the transformation is complete. Since the transformation occurs in background and asyncronously, calling this method immediately after processXSL will result in holding the control until tre result is avialable.

processXSL(XSLStylesheet, InputStream, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.InputStream xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, Reader, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.Reader xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, URL, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.net.URL xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument, OutputStream)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml, java.io.OutputStream os)`

Initiate XSL Transformation in the background. The control is returned immediately.

removeDOMTransformerErrorListener(XSLTransformerErrorListener)

`public synchronized void removeDOMTransformerErrorListener(XSLTransformerErrorListener p0)`

Renmoves an error event listener

Parameters:

`p0` -- XSLTransformerErrorListener

removeXSLTransformerListener(XSLTransformerListener)

`public synchronized void removeXSLTransformerListener(XSLTransformerListener p0)`

Removes a listener

Parameters:

`p0` -- XSLTransformerListener

run()

`public void run()`

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setErrorStream(OutputStream)

`public final void setErrorStream(java.io.OutputStream out)`

Sets the error stream used by the XSL processor

showWarnings(boolean)

`public final void showWarnings(boolean yes)`

Sets the showWarnings flage used by the XSL processor

oracle.xml.async
XSLTransformerBeanInfo

Syntax

```
public class XSLTransformerBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.XSLTransformerBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XSLTransformerBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XSLTransformerBeanInfo()

```
public  XSLTransformerBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async XSLTransformerErrorEvent

Syntax

```
public class XSLTransformerErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.

Member Summary	
Constructors	
XSLTransformerErrorEvent(Object, Exception)	
Methods	
getException()	Returns the exception that XSLTransformer encountered object unique id.
getMessage()	Returns the error message that describes the error that XSLTransformer encountered

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerErrorEvent(Object, Exception)

```
public  XSLTransformerErrorEvent(java.lang.Object p0, java.lang.Exception e)
```

Methods

getException()

```
public java.lang.Exception getException()
```

Returns the exception that XSLTransformer encountered object unique id. Can be used to

Returns:

The transformation exception

getMessage()

```
public java.lang.String getMessage()
```

Returns the error message that describes the error that XSLTransformer encountered

Returns:

The error message

oracle.xml.async XSLTransformerErrorListener

Syntax

`public interface XSLTransformerErrorListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about error events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet error event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerErrorCalled(XSLTransformerErrorEvent)	This method is called when parse or transformation error occur.

Methods

xslTransformerErrorCalled(XSLTransformerErrorEvent)

`public void xslTransformerErrorCalled(XSLTransformerErrorEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` -- The XSLTransformerErrorEvent object produced by the XSLTransformer

oracle.xml.async XSLTransformerEvent

Syntax

```
public class XSLTransformerEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that XSLTransformer uses to notify all registered listeners about transformation events.

Member Summary	
Constructors	
XSLTransformerEvent(Object, int)	
Methods	
getID()	Returns the XSLTransformer object unique id.

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerEvent(Object, int)

```
public  XSLTransformerEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the XSLTransformer object unique id. Can be used to identify which instance of the XSLTransformer generated this event in case when multiple instances of XSLTransformer work in the background.

Returns:

The event id

oracle.xml.async XSLTransformerListener

Syntax

`public interface XSLTransformerListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerError(XSLTransformerEvent)	This method is called when parse or transformation error occur.
xslTransformerOver(XSLTransformerEvent)	This method is called when the transformation is complete
xslTransformerStarted(XSLTransformerEvent)	This method is called when the transformation starts

Methods

xslTransformerError(XSLTransformerEvent)

`public void xslTransformerError(XSLTransformerEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` -- The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerOver(XSLTransformerEvent)

`public void xslTransformerOver(XSLTransformerEvent p0)`
This method is called when the transformation is complete

Parameters:

`p0` -- The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerStarted(XSLTransformerEvent)

`public void xslTransformerStarted(XSLTransformerEvent p0)`
This method is called when the transformation starts

Parameters:

`p0` -- The XSLTransformerEvent object produced by the XSLTransformer

oracle.xml.srcviewer Package

Description

Class Summary	
Classes	
oracle.xml.srcviewer.XMLSourceView	Shows an XML document.
XMLSourceViewBeanInfo	

oracle.xml.srcviewer.XMLSourceView

Syntax

```
public class XMLSourceView extends javax.swing.JPanel implements
java.io.Serializable

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.srcviewer.XMLSourceView
```

All Implemented Interfaces:

`javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable`

Description

Shows an XML document. Recognizes the following XML token types: `Tag, Attribute Name, Attribute Value, Comment, CDATA, PCDATA, PI Data, PI Name and NOTATION Symbol`. Each token type has a foreground color and font. The default color/font settings can be changed by the user. Takes as input an org.w3c.dom.Document object.

Member Summary

Constructors	
XMLSourceView()	The class constructor.
Methods	
fontGet(AttributeSet)	Extracts and returns the font from a given attributeset.
fontSet(MutableAttributeSet, Font)	Sets the mutableattributeset font.
getAttributeNameFont()	Returns the Attribute Value font.
getAttributeNameForeground()	Returns the Attribute Name foreground color.

Member Summary	
getAttributeValueFont()	Returns the Attribute Value font.
getAttributeValueForeground()	Returns the Attribute Value foreground color.
getBackground()	Returns the background color.
getCDATAFont()	Returns the CDATA font.
getCDATAForeground()	Returns the CDATA foreground color.
getCommentDataFont()	Returns the Comment Data font.
getCommentDataForeground()	Returns the Comment Data foreground color.
getEditedText()	Returns the edited text.
getJTextPane()	Returns the viewer `JTextPane` component.
getMinimumSize()	Returns the XMLSourceView minimal size.
getNodeAtOffset(int)	Returns the XML node at a given offset.
getPCDATAFont()	Returns the PCDATA font.
getPCDATAForeground()	Returns the PCDATA foreground color.
getPIDataFont()	Returns the PI Data font.
getPIDataForeground()	Returns the PI Data foreground color.
getPINameFont()	Returns the PI Name font.
getPINameForeground()	Returns the PI Data foreground color.
getSymbolFont()	Returns the NOTATION Symbol font.
getSymbolForeground()	Returns the NOTATION Symbol foreground color.
getTagFont()	Returns the Tag font.
getTagForeground()	Returns the Tag foreground color.
getText()	Returns the XML document as a String.
isEditable()	Returns boolean to indicate whether this object is editable.
selectNodeAt(int)	Moves the cursor to XML Node at offset `i`.
setAttributeNameFont(Font)	Sets the Attribute Name font.
setAttributeNameForeground(Color)	Sets the Attribute Name foreground color.
setAttributeValueFont(Font)	Sets the Attribute Value font.
setAttributeValueForeground(Color)	Sets the Attribute Value foreground color.

Member Summary	
setBackground(Color)	Sets the background color.
setCDATAFont(Font)	Sets the CDATA font.
setCDATAForeground(Color)	Sets the CDATA foreground color.
setCommentDataFont(Font)	Sets the Comment font.
setCommentDataForeground(Color)	Sets the Comment foreground color.
setEditable(boolean)	Sets the specified boolean to indicate whether this object should be editable.
setPCDATAFont(Font)	Sets the PCDATA font.
setPCDATAForeground(Color)	Sets the PCDATA foreground color.
setPIDataFont(Font)	Sets the PI Data font.
setPIDataForeground(Color)	Sets the PI Data foreground color.
setPINameFont(Font)	Sets the PI Name font.
setPINameForeground(Color)	Sets the PI Name foreground color.
setSelectedNode(Node)	Sets the cursor position at the selected XML node.
setSymbolFont(Font)	Sets the NOTATION Symbol font.
setSymbolForeground(Color)	Sets the NOTATION Symbol foreground color.
setTagFont(Font)	Sets the Tag font.
setTagForeground(Color)	Sets the Tag foreground color.
setXMLDocument(Document)	Associates the XMLviewer with a XML document.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getNextFocusableComponent, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

Inherited Member Summary

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLSourceView()

```
public  XMLSourceView()
```

The class constructor. Creates an object of type `XMLSourceView`.

Methods

fontGet(AttributeSet)

```
public static java.awt.Font fontGet(javax.swing.text.AttributeSet attributeset)
```

Extracts and returns the font from a given attributeset.

Parameters:

`attributeset` - The source `Attributeset`.

Returns:

The extracted `Font`.

fontSet(MutableAttributeSet, Font)

```
public static void fontSet(javax.swing.text.MutableAttributeSet
mutableattributeset, java.awt.Font font)
```

Sets the mutableattributeset font.

Parameters:

`mutableattributeset` - The `mutableattributeset` to update.

`font` - The new `Font` for the mutableattributeset.

getAttributeNameFont()

```
public java.awt.Font getAttributeNameFont()
```

Returns the Attribute Value font.

Returns:

The `Font` object.

getAttributeNameForeground()

```
public java.awt.Color getAttributeNameForeground()
```

Returns the Attribute Name foreground color.

Returns:

The `Color` object.

getAttributeValueFont()

```
public java.awt.Font getAttributeValueFont()
```

Returns the Attribute Value font.

Returns:

The `Font` object.

getAttributeValueForeground()

```
public java.awt.Color getAttributeValueForeground()
```

Returns the Attribute Value foreground color.

Returns:

The `Color` object.

getBackground()

`public java.awt.Color getBackground()`
Returns the background color.

Overrides:

java.awt.Component.getBackground() in class java.awt.Component

Returns:

The `Color` object.

getCDATAFont()

`public java.awt.Font getCDATAFont()`
Returns the CDATA font.

Returns:

The `Font` object.

getCDATAForeground()

`public java.awt.Color getCDATAForeground()`
Returns the CDATA foreground color.

Returns:

The `Color` object.

getCommentDataFont()

`public java.awt.Font getCommentDataFont()`
Returns the Comment Data font.

Returns:

The `Font` object.

getCommentDataForeground()

`public java.awt.Color getCommentDataForeground()`

Returns the Comment Data foreground color.

Returns:

The `Color` object.

getEditedText()

`public java.lang.String getEditedText()`
Returns the edited text.

Returns:

The `String` object containing the edited text.

getJTextPane()

`public javax.swing.JTextPane getJTextPane()`
Returns the viewer `JTextPane` component.

Returns:

The `JTextPane` object used by XMLSourceViewer

getMinimumSize()

`public java.awt.Dimension getMinimumSize()`
Returns the XMLSourceView minimal size.

Overrides:

javax.swing.JComponent.getMinimumSize() in class javax.swing.JComponent

Returns:

The `Dimension` object containing the XMLSourceView minimum size.

getNodeAtOffset(int)

`public org.w3c.dom.Node getNodeAtOffset(int i)`
Returns the XML node at a given offset.

Parameters:

`i` - The node offset.

Returns:

The `Node` object from offset `i`.

getPCDATAFont()

`public java.awt.Font getPCDATAFont()`

Returns the PCDATA font.

Returns:

The `Font` object.

getPCDATAForeground()

`public java.awt.Color getPCDATAForeground()`

Returns the PCDATA foreground color.

Returns:

The `Color` object.

getPIDataFont()

`public java.awt.Font getPIDataFont()`

Returns the PI Data font.

Returns:

The `Font` object

getPIDataForeground()

`public java.awt.Color getPIDataForeground()`

Returns the PI Data foreground color.

Returns:

The `Color` object.

getPINameFont()

`public java.awt.Font getPINameFont()`

Returns the PI Name font.

Returns:
The `Font` object.

getPINameForeground()

```
public java.awt.Color getPINameForeground()
```
Returns the PI Data foreground color.

Returns:
The `Color` object.

getSymbolFont()

```
public java.awt.Font getSymbolFont()
```
Returns the NOTATION Symbol font.

Returns:
The `Font` object.

getSymbolForeground()

```
public java.awt.Color getSymbolForeground()
```
Returns the NOTATION Symbol foreground color.

Returns:
The `Color` object.

getTagFont()

```
public java.awt.Font getTagFont()
```
Returns the Tag font.

Returns:
The `Font` object.

getTagForeground()

```
public java.awt.Color getTagForeground()
```
Returns the Tag foreground color.

Returns:

The `Color` object.

getText()

`public java.lang.String getText()`
Returns the XML document as a String.

Returns:

The `String` object containing the XML document.

isEditable()

`public boolean isEditable()`
Returns boolean to indicate whether this object is editable.

selectNodeAt(int)

`public void selectNodeAt(int i)`
Moves the cursor to XML Node at offset `i`.

Parameters:

`i` - The node offset.

setAttributeNameFont(Font)

`public void setAttributeNameFont(java.awt.Font font)`
Sets the Attribute Name font.

Parameters:

`font` - The new `Font` for Attribute Name.

setAttributeNameForeground(Color)

`public void setAttributeNameForeground(java.awt.Color color)`
Sets the Attribute Name foreground color.

Parameters:

`color` - The new `Color` for Attribute Name.

setAttributeValueFont(Font)

`public void setAttributeValueFont(java.awt.Font font)`
Sets the Attribute Value font.

Parameters:

`font` - The new `Font` for Attribute Value.

setAttributeValueForeground(Color)

`public void setAttributeValueForeground(java.awt.Color color)`
Sets the Attribute Value foreground color.

Parameters:

`color` - The new `Color` for Attribute Value.

setBackground(Color)

`public void setBackground(java.awt.Color color)`
Sets the background color.

Overrides:

javax.swing.JComponent.setBackground(java.awt.Color) in class javax.swing.JComponent

Parameters:

`color` - The new background `Color`.

setCDATAFont(Font)

`public void setCDATAFont(java.awt.Font font)`
Sets the CDATA font.

Parameters:

`font` - The new `Font` for CDATA.

setCDATAForeground(Color)

`public void setCDATAForeground(java.awt.Color color)`
Sets the CDATA foreground color.

Parameters:

`color` - The new `Color` for CDATA.

setCommentDataFont(Font)

```
public void setCommentDataFont(java.awt.Font font)
```

Sets the Comment font.

Parameters:

`font` - The new `Font` for the XML Comments.

setCommentDataForeground(Color)

```
public void setCommentDataForeground(java.awt.Color color)
```

Sets the Comment foreground color.

Parameters:

`color` - The new `Color` for Comment.

setEditable(boolean)

```
public void setEditable(boolean edit)
```

Sets the specified boolean to indicate whether this object should be editable.

Parameters:

`doc` - The new `boolean` value.

setPCDATAFont(Font)

```
public void setPCDATAFont(java.awt.Font font)
```

Sets the PCDATA font.

Parameters:

`font` - The new `Font` for PCDATA.

setPCDATAForeground(Color)

```
public void setPCDATAForeground(java.awt.Color color)
```

Sets the PCDATA foreground color.

Parameters:

`color` - The new `Color` for PCDATA.

setPIDataFont(Font)

`public void setPIDataFont(java.awt.Font font)`

Sets the PI Data font.

Parameters:

`font` - The new `Font` for PI Data.

setPIDataForeground(Color)

`public void setPIDataForeground(java.awt.Color color)`

Sets the PI Data foreground color.

Parameters:

`color` - The new `Color` for PI Data.

setPINameFont(Font)

`public void setPINameFont(java.awt.Font font)`

Sets the PI Name font.

Parameters:

`font` - The new `Font` for the PI Names.

setPINameForeground(Color)

`public void setPINameForeground(java.awt.Color color)`

Sets the PI Name foreground color.

Parameters:

`color` - The new `Color` for PI Name.

setSelectedNode(Node)

`public void setSelectedNode(org.w3c.dom.Node node)`

Sets the cursor position at the selected XML node.

Parameters:

`node` - The selected node.

setSymbolFont(Font)

`public void setSymbolFont(java.awt.Font font)`

Sets the NOTATION Symbol font.

Parameters:

`color` - The new `Font` for NOTATION Symbol.

setSymbolForeground(Color)

`public void setSymbolForeground(java.awt.Color color)`

Sets the NOTATION Symbol foreground color.

Parameters:

`color` - The new `Color` for NOTATION Symbol.

setTagFont(Font)

`public void setTagFont(java.awt.Font font)`

Sets the Tag font.

Parameters:

`font` - The new `Font` for the XML Tags.

setTagForeground(Color)

`public void setTagForeground(java.awt.Color color)`

Sets the Tag foreground color.

Parameters:

`color` - The new `Color` for the XML Tags.

setXMLDocument(Document)

`public void setXMLDocument(org.w3c.dom.Document document)`

Associates the XMLviewer with a XML document.

Parameters:

`doc` - The `Document` document to display.

See Also:

getText()

oracle.xml.srcviewer XMLSourceViewBeanInfo

Syntax

```
public class XMLSourceViewBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.srcviewer.XMLSourceViewBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLSourceViewBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLSourceViewBeanInfo()

```
public  XMLSourceViewBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async Package

Description

Class Summary	
Interfaces	
DOMBuilderErrorListener	This interface must be implemented in order to receive notifications when error is found during parsing.
DOMBuilderListener	This interface must be implemented in order to receive notifications about events during the asyncronous parsing.
XSLTransformerErrorListener	This interface must be implemented in order to receive notifications about error events during the asyncronous transformation.
XSLTransformerListener	This interface must be implemented in order to receive notifications about events during the asyncronous transformation.
Classes	
DOMBuilder	This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.
DOMBuilderBeanInfo	
DOMBuilderErrorEvent	This class defines the error event which is sent when parse exception occurs.
DOMBuilderEvent	The event object that DOMBuilder uses to notify all registered listeners about parse events.
ResourceManager	Simple semaphore that maintains access to fixed number of logical resources.
XSLTransformer	Applies XSL transformation in a background thread.
XSLTransformerBeanInfo	
XSLTransformerErrorEvent	The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.
XSLTransformerEvent	The event object that XSLTransformer uses to notify all registered listeners about transformation events.

oracle.xml.async DOMBuilder

Syntax

```
public class DOMBuilder extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.DOMBuilderConstants, java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.DOMBuilder
```

All Implemented Interfaces:

`oracle.xml.async.DOMBuilderConstants, java.lang.Runnable, java.io.Serializable`

Description

This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation. to parse a XML document and build a DOM tree. The parsing is done in a separate thread and DOMBuilderListener interface must be used for notification when the tree is built.

Member Summary	
Constructors	
DOMBuilder()	Creates a new parser object.
Creates a new parser object.	Creates a new parser object with a given id.
Methods	
addDOMBuilderErrorListener(DOMBuilderErrorListener)	Adds DOMBuilderErrorListener
addDOMBuilderListener(DOMBuilderListener)	Adds DOMBuilderListener
	Get the DTD
getDocument()	Gets the document
getId()	Returns the parser object id.
getReleaseVersion()	Returns the release version of the Oracle XML Parser
	Gets the document

Member Summary	
getValidationMode()	Returns the validation mode
parse(InputSource)	Parses the XML from given input source
	Parses the XML from given input stream.
parse(Reader)	Parses the XML from given input stream.
parse(String)	Parses the XML from the URL indicated
parse(URL)	Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.
parseDTD(InputSource, String)	Parses the XML External DTD from given input source
parseDTD(InputStream, String)	Parses the XML External DTD from given input stream.
	Parses the XML External DTD from given input stream.
parseDTD(String, String)	Parses the XML External DTD from the URL indicated
parseDTD(URL, String)	Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.
removeDOMBuilderErrorListener(DOMBuilderErrorListener)	Remove DOMBuilderErrorListener
removeDOMBuilderListener(DOMBuilderListener)	Remove DOMBuilderListener
run()	This method runs in a thread
setBaseURL(URL)	Set the base URL for loading external enitites and DTDs.
setDebugMode(boolean)	Sets a flag to turn on debug information in the document
setDoctype(DTD)	Set the DTD
setErrorStream(OutputStream)	Creates an output stream for the output of errors and warnings.
setErrorStream(OutputStream, String)	Creates an output stream for the output of errors and warnings.
setErrorStream(PrintWriter)	Creates an output stream for the output of errors and warnings.
setNodeFactory(NodeFactory)	Set the node factory.
setPreserveWhitespace(boolean)	Set the white space preserving mode
setValidationMode(boolean)	Set the validation mode
showWarnings(boolean)	Switch to determine whether to print warnings

Inherited Member Summary

Fields inherited from interface oracle.xml.async.DOMBuilderConstants

DOM_GEN_ENDED, DOM_GEN_ERROR, DOM_GEN_STARTED, DOM_METHOD1, DOM_METHOD10, DOM_METHOD2, DOM_METHOD3, DOM_METHOD4, DOM_METHOD5, DOM_METHOD6, DOM_METHOD7, DOM_METHOD8, DOM_METHOD9

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

DOMBuilder()

`public  DOMBuilder()`
Creates a new parser object.

DOMBuilder(int)

`public  DOMBuilder(int id)`
Creates a new parser object with a given id.

Methods

addDOMBuilderErrorListener(DOMBuilderErrorListener)

`public void addDOMBuilderErrorListener(DOMBuilderErrorListener p0)`
Adds DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to add

addDOMBuilderListener(DOMBuilderListener)

`public void addDOMBuilderListener(DOMBuilderListener p0)`
Adds DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to add

getDoctype()

`public synchronized oracle.xml.async.DTD getDoctype()`

Get the DTD

Returns:

The `DTD`

getDocument()

`public synchronized oracle.xml.async.XMLDocument getDocument()`
Gets the document

Returns:

The document being parsed

getId()

`public int getId()`
Returns the parser object id.

getReleaseVersion()

`public synchronized java.lang.String getReleaseVersion()`
Returns the release version of the Oracle XML Parser

Returns:

the release version string

getResult()

`public synchronized org.w3c.dom.Document getResult()`
Gets the document

Returns:

The document being parsed

getValidationMode()

`public synchronized boolean getValidationMode()`
Returns the validation mode

Returns:

`true` if the XML parser is validating `false` if not

parse(InputSource)

`public final synchronized void parse(org.xml.sax.InputSource in)`

Parses the XML from given input source

Parameters:

`in` - the `org.xml.sax.InputSouce` to parse

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(InputStream)

`public final synchronized void parse(java.io.InputStream in)`

Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(Reader)

`public final synchronized void parse(java.io.Reader r)`

Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(String)

`public final synchronized void parse(java.lang.String in)`
Parses the XML from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(URL)

`public final synchronized void parse(java.net.URL url)`
Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputSource, String)

`public final synchronized void parseDTD(org.xml.sax.InputSource in, java.lang.String rootName)`

Parses the XML External DTD from given input source

Parameters:

`in` - the `org.xml.sax.InputSouce` to parse

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputStream, String)

`public final synchronized void parseDTD(java.io.InputStream in, java.lang.String rootName)`

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(Reader, String)

`public final synchronized void parseDTD(java.io.Reader r, java.lang.String rootName)`

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(String, String)

`public final synchronized void parseDTD(java.lang.String in, java.lang.String rootName)`

Parses the XML External DTD from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(URL, String)

`public final synchronized void parseDTD(java.net.URL url, java.lang.String rootName)`

Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

removeDOMBuilderErrorListener(DOMBuilderErrorListener)

```
public synchronized void removeDOMBuilderErrorListener(DOMBuilderErrorListener
p0)
```

Remove DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to remove

removeDOMBuilderListener(DOMBuilderListener)

```
public synchronized void removeDOMBuilderListener(DOMBuilderListener p0)
```

Remove DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to remove

run()

```
public void run()
```

This method runs in a thread

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setBaseURL(URL)

```
public synchronized void setBaseURL(java.net.URL url)
```

Set the base URL for loading external enitites and DTDs. This method should to be called if the parse(InputStream) is used to parse the XML Document

Parameters:

`url` - The base URL

setDebugMode(boolean)

`public void setDebugMode(boolean flag)`
Sets a flag to turn on debug information in the document

Parameters:

`flag` - determines whether debug info is stored

setDoctype(DTD)

`public synchronized void setDoctype(oracle.xml.async.DTD dtd)`
Set the DTD

Parameters:

`dtd` - `DTD` to set and used while parsing

setErrorStream(OutputStream)

`public final synchronized void setErrorStream(java.io.OutputStream out)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The output stream to use for errors and warnings

setErrorStream(OutputStream, String)

```
public final synchronized void setErrorStream(java.io.OutputStream out,
java.lang.String enc)
```
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings. Additionally, an .exception is thrown if the encoding specified is unsupported.

Parameters:

`out` - The output stream to use for errors and warnings

`enc` - the encoding to use

Throws:

`IOException` - if an unsupported encoding is specified

setErrorStream(PrintWriter)

`public final synchronized void setErrorStream(java.io.PrintWriter out)`
Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The `PrintWriter` to use for errors and warnings

setNodeFactory(NodeFactory)

`public synchronized void setNodeFactory(oracle.xml.async.NodeFactory factory)`
Set the node factory. Applications can extend the NodeFactory and register it through this method. The parser will then use the user supplied NodeFactory to create nodes of the DOM tree.

Parameters:

`factory` - The `NodeFactory` to set

Throws:

`XMLParseException` - if an invalid factory is set

See Also:

`NodeFactory`

setPreserveWhitespace(boolean)

`public synchronized void setPreserveWhitespace(boolean flag)`
Set the white space preserving mode

Parameters:

`flag` - preserving mode

setValidationMode(boolean)

`public synchronized void setValidationMode(boolean yes)`
Set the validation mode

Parameters:

`yes` - determines whether the XML parser should be validating

showWarnings(boolean)

```
public synchronized void showWarnings(boolean yes)
```

Switch to determine whether to print warnings

Parameters:

`yes` - determines whether warnings should be shown

oracle.xml.async
DOMBuilderBeanInfo

Syntax

```
public class DOMBuilderBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.DOMBuilderBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

DOMBuilderBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

DOMBuilderBeanInfo()

```
public  DOMBuilderBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async DOMBuilderErrorEvent

Syntax

```
public class DOMBuilderErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

This class defines the error event which is sent when parse exception occurs.

Member Summary

Constructors	
DOMBuilderErrorEvent(Object, Exception)	Constructor for DOMBuilderErrorEvent.
Methods	
getException()	Gets the Exception
getMessage()	Returns the error message generated by the parser

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderErrorEvent(Object, Exception)

```
public  DOMBuilderErrorEvent(java.lang.Object p0, java.lang.Exception e)
```

Constructor for DOMBuilderErrorEvent.

Parameters:

`p0` - the `Object` that create this object

`e` - the `Exception` raised.

Methods

getException()

```
public java.lang.Exception getException()
```

Gets the Exception

Returns:

The Exception beind raised

getMessage()

```
public java.lang.String getMessage()
```

Returns the error message generated by the parser

Returns:

the error message string

oracle.xml.async DOMBuilderErrorListener

Syntax

`public interface DOMBuilderErrorListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications when error is found during parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderErrorListener method.

Member Summary
Methods
domBuilderErrorCalled(DOMBuilderErrorEvent)

Methods

domBuilderErrorCalled(DOMBuilderErrorEvent)

`public void domBuilderErrorCalled(DOMBuilderErrorEvent p0)`

Parameters:

`p0` - - The DOMBuilderErrorEvent object produced by the DOMBuilder as result of parsing error

oracle.xml.async
DOMBuilderEvent

Syntax

```
public class DOMBuilderEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that DOMBuilder uses to notify all registered listeners about parse events.

Member Summary	
Constructors	
DOMBuilderEvent(Object, int)	
Methods	
getID()	Returns the DOMBuilder object unique id.

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderEvent(Object, int)

```
public  DOMBuilderEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the DOMBuilder object unique id. Can be used to identify which instance of the DOMBuilder generated this event in case when multiple instances of DOMBuilder work in the background.

Returns:

The event id

oracle.xml.async DOMBuilderListener

Syntax

`public interface DOMBuilderListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about events during the asyncronous parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderListener method.

Member Summary

Methods	
domBuilderError(DOMBuilderEvent)	This method is called when parse error occur.
domBuilderOver(DOMBuilderEvent)	This method is called when the parse is complete
domBuilderStarted(DOMBuilderEvent)	This method is called when parse starts

Methods

domBuilderError(DOMBuilderEvent)

`public void domBuilderError(DOMBuilderEvent p0)`

This method is called when parse error occur.

Parameters:

`p0` - - The DOMBuilderEvent object produced by the DOMBuilder

domBuilderOver(DOMBuilderEvent)

`public void domBuilderOver(DOMBuilderEvent p0)`

This method is called when the parse is complete

Parameters:

`p0` -- The DOMBuilderEvent object produced by the DOMBuilder

domBuilderStarted(DOMBuilderEvent)

```
public void domBuilderStarted(DOMBuilderEvent p0)
```

This method is called when parse starts

Parameters:

`p0` -- The DOMBuilderEvent object produced by the DOMBuilder

oracle.xml.async ResourceManager

Syntax

```
public class ResourceManager extends java.lang.Object

java.lang.Object
  |
 +--oracle.xml.async.ResourceManager
```

Description

Simple semaphore that maintains access to fixed number of logical resources.

Member Summary	
Constructors	
ResourceManager(int)	The ResourceManager constructor
Methods	
activeFound()	Returns how many resources are avialable.
getResource()	If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns.
releaseResource()	Releases a resource.
sleep(int)	Let's you use Thread.sleep() without try/catch

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ResourceManager(int)

```
public  ResourceManager(int i)
```

The ResourceManager constructor

Parameters:

`i` -- the number of resources to manage

Methods

activeFound()

`public boolean activeFound()`
Returns how many resources are avialable.

getResource()

`public synchronized void getResource()`
If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns. Othervise, waits until resource is released.

releaseResource()

`public void releaseResource()`
Releases a resource. When this method is called, the number of resources avialable is increased by one.

sleep(int)

`public void sleep(int i)`
Let's you use Thread.sleep() without try/catch

oracle.xml.async
XSLTransformer

Syntax

```
public class XSLTransformer extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.XSLTransformerConstants,
java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.XSLTransformer
```

All Implemented Interfaces:

```
java.lang.Runnable, java.io.Serializable,
oracle.xml.async.XSLTransformerConstants
```

Description

Applies XSL transformation in a background thread.

Member Summary

Constructors	
XSLTransformer()	XSLTransformet constructor
XSLTransformer(int)	XSLTransformer constructor
Methods	
addXSLTransformerErrorListener(XSLTransformerErrorListener)	Adds an error event listener
addXSLTransformerListener(XSLTransformerListener)	Adds a listener
	Returns the unique XSLTransformer id
getResult()	Returns the document fragment for the resulting document.
processXSL(XSLStylesheet, InputStream, URL)	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, Reader, URL)	Initiate XSL Transformation in the background.

Member Summary	
processXSL(XSLStylesheet, URL, URL)	Initiate XSL Transformation in the background.
	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, XMLDocument, OutputStream)	Initiate XSL Transformation in the background.
removeDOMTransformerErrorListener (XSLTransformerErrorListener)	Renmoves an error event listener
removeXSLTransformerListener(XSLTransformerListener)	Removes a listener
run()	
	Sets the error stream used by the XSL processor
showWarnings(boolean)	Sets the showWarnings flage used by the XSL processor

Inherited Member Summary
Fields inherited from interface oracle.xml.async.XSLTransformerConstants
XSL_GEN_ENDED, XSL_GEN_ERROR, XSL_GEN_STARTED
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSLTransformer()

`public XSLTransformer()`

XSLTransformet constructor

XSLTransformer(int)

`public XSLTransformer(int id)`

XSLTransformer constructor

Parameters:

`id` -- A unique integer that can be used to identify the XSLTransformer instance during event processing

Methods

addXSLTransformerErrorListener(XSLTransformerErrorListener)

`public void addXSLTransformerErrorListener(XSLTransformerErrorListener p0)`

Adds an error event listener

Parameters:

`p0` -- XSLTransformerErrorListener

addXSLTransformerListener(XSLTransformerListener)

`public void addXSLTransformerListener(XSLTransformerListener p0)`

Adds a listener

Parameters:

`p0` -- XSLTransformerListener

getId()

`public int getId()`

Returns the unique XSLTransformer id

Returns:

The id

getResult()

`public synchronized oracle.xml.async.DocumentFragment getResult()`

Returns the document fragment for the resulting document. Call this method only after receiving notification that the transformation is complete. Since the transformation occurs in background and asyncronously, calling this method immediately after processXSL will result in holding the control until tre result is avialable.

processXSL(XSLStylesheet, InputStream, URL)

```
public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.InputStream
xml, java.net.URL ref)
```

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, Reader, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.Reader xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, URL, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.net.URL xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument, OutputStream)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml, java.io.OutputStream os)`

Initiate XSL Transformation in the background. The control is returned immediately.

removeDOMTransformerErrorListener(XSLTransformerErrorListener)

`public synchronized void removeDOMTransformerErrorListener(XSLTransformerErrorListener p0)`

Renmoves an error event listener

Parameters:

`p0` - - XSLTransformerErrorListener

removeXSLTransformerListener(XSLTransformerListener)

`public synchronized void removeXSLTransformerListener(XSLTransformerListener p0)`

Removes a listener

Parameters:

`p0` - - XSLTransformerListener

run()

`public void run()`

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setErrorStream(OutputStream)

```
public final void setErrorStream(java.io.OutputStream out)
```

Sets the error stream used by the XSL processor

showWarnings(boolean)

```
public final void showWarnings(boolean yes)
```

Sets the showWarnings flage used by the XSL processor

oracle.xml.async XSLTransformerBeanInfo

Syntax

```
public class XSLTransformerBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.XSLTransformerBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XSLTransformerBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XSLTransformerBeanInfo()

```
public  XSLTransformerBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async XSLTransformerErrorEvent

Syntax

```
public class XSLTransformerErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.

Member Summary	
Constructors	
XSLTransformerErrorEvent(Object, Exception)	
Methods	
getException()	Returns the exception that XSLTransformer encountered object unique id.
getMessage()	Returns the error message that describes the error that XSLTransformer encountered

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerErrorEvent(Object, Exception)

```
public  XSLTransformerErrorEvent(java.lang.Object p0, java.lang.Exception e)
```

Methods

getException()

```
public java.lang.Exception getException()
```

Returns the exception that XSLTransformer encountered object unique id. Can be used to

Returns:

The transformation exception

getMessage()

```
public java.lang.String getMessage()
```

Returns the error message that describes the error that XSLTransformer encountered

Returns:

The error message

oracle.xml.async XSLTransformerErrorListener

Syntax

`public interface XSLTransformerErrorListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about error events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet error event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerErrorCalled(XSLTransformerErrorEvent)	This method is called when parse or transformation error occur.

Methods

xslTransformerErrorCalled(XSLTransformerErrorEvent)

`public void xslTransformerErrorCalled(XSLTransformerErrorEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` -- The XSLTransformerErrorEvent object produced by the XSLTransformer

oracle.xml.async XSLTransformerEvent

Syntax

```
public class XSLTransformerEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that XSLTransformer uses to notify all registered listeners about transformation events.

Member Summary

Constructors	
XSLTransformerEvent(Object, int)	
Methods	
getID()	Returns the XSLTransformer object unique id.

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerEvent(Object, int)

```
public  XSLTransformerEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the XSLTransformer object unique id. Can be used to identify which instance of the XSLTransformer generated this event in case when multiple instances of XSLTransformer work in the background.

Returns:

The event id

oracle.xml.async XSLTransformerListener

Syntax

`public interface XSLTransformerListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerError(XSLTransformerEvent)	This method is called when parse or transformation error occur.
xslTransformerOver(XSLTransformerEvent)	This method is called when the transformation is complete
xslTransformerStarted(XSLTransformerEvent)	This method is called when the transformation starts

Methods

xslTransformerError(XSLTransformerEvent)

`public void xslTransformerError(XSLTransformerEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` -- The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerOver(XSLTransformerEvent)

`public void xslTransformerOver(XSLTransformerEvent p0)`

This method is called when the transformation is complete

Parameters:

`p0` - - The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerStarted(XSLTransformerEvent)

`public void xslTransformerStarted(XSLTransformerEvent p0)`

This method is called when the transformation starts

Parameters:

`p0` - - The XSLTransformerEvent object produced by the XSLTransformer

oracle.xml.transviewer Package

Description

Class Summary	
Classes	
DBAccess	Maintains CLOB tables that can hold multiple XML and text documents.
XMLTransformPanel	XMLTransformPanel visual bean.
XMLTransformPanelBeanInfo	
XMLTransViewer	Simple application that uses XMLTransformPanel.

oracle.xml.transviewer
DBAccess

Syntax

```
public class DBAccess extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.transviewer.DBAccess
```

Description

Maintains CLOB tables that can hold multiple XML and text documents. Each table is created using the statement: CREATE TABLE tablename FILENAME CHAR(16) UNIQUE, FILEDATA CLOB) LOB(FILEDATA) STORE AS (DISABLE STORAGE IN ROW). Each XML (or text) document is stored as a row in the table and the FILENAME field holds a unique string that is used as a key to retrieve, update or delete the row. The document text is stored in the FILEDATA field that is a CLOB object. This CLOB tables are automatically maintained by the transviewer bean. The CLOB tables maintained by this class can be later used by the transviewer bean. The class creates and deletes CLOB tables, list a CLOB table content and also add, replace or delete text documents in this CLOB tables.

Member Summary

Constructors	
DBAccess()	
Methods	
createXMLTable(Connection, String)	Create XML table
deleteXMLName(Connection, String, String)	Delete text file from XML table
dropXMLTable(Connection, String)	Delete XML table
getNameSize()	Returns the size of the field where the filename is kept.
getXMLData(Connection, String, String)	Retrieve text file from XML table
getXMLNames(Connection, String)	Returns all file names in XML table

Member Summary	
getXMLTableNames(Connection, String)	Gets all XML tables with names starting with a given string
insertXMLData(Connection, String, String, String)	Inserts text file as a row in XML table
isXMLTable(Connection, String)	Check if the table is XML table.
replaceXMLData(Connection, String, String, String)	Replace text file as a row in XML table
xmlTableExists(Connection, String)	Checks if the XML table exists

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

DBAccess()

`public DBAccess()`

Methods

createXMLTable(Connection, String)

`public boolean createXMLTable(java.sql.Connection con, java.lang.String tablename)`

Create XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

true if successfull

deleteXMLName(Connection, String, String)

`public boolean deleteXMLName(java.sql.Connection con, java.lang.String tablename, java.lang.String xmlname)`

Delete text file from XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

Returns:

true if successfull

dropXMLTable(Connection, String)

`public boolean dropXMLTable(java.sql.Connection con, java.lang.String tablename)`

Delete XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

true if successfull

getNameSize()

`public int getNameSize()`

Returns the size of the field where the filename is kept.

Returns:

filename size

getXMLData(Connection, String, String)

`public java.lang.String getXMLData(java.sql.Connection con, java.lang.String tablename, java.lang.String xmlname)`

Retrieve text file from XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

Returns:

file as a string

getXMLNames(Connection, String)

```
public java.lang.String[] getXMLNames(java.sql.Connection con, java.lang.String
tablename)
```

Returns all file names in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

String array with all file names in this table

getXMLTableNames(Connection, String)

```
public java.lang.String[] getXMLTableNames(java.sql.Connection con,
java.lang.String tablePrefix)
```

Gets all XML tables with names starting with a given string

Parameters:

`con` -- the Connection object

`tablePrefix` -- table prefix string

Returns:

array of all XML tables that begin with tablePrefix

insertXMLData(Connection, String, String, String)

```
public boolean insertXMLData(java.sql.Connection con, java.lang.String
tablename, java.lang.String xmlname, java.lang.String xmldata)
```

Inserts text file as a row in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

`xmldata` -- string with the file data

Returns:

true if successfull

isXMLTable(Connection, String)

```
public boolean isXMLTable(java.sql.Connection con, java.lang.String tablename)
```

Check if the table is XML table.

Parameters:

`con` -- the Connection object

`tableName` -- the table name to test

Returns:

true if this is XML table

replaceXMLData(Connection, String, String, String)

```
public boolean replaceXMLData(java.sql.Connection con, java.lang.String
tablename, java.lang.String xmlname, java.lang.String xmldata)
```

Replace text file as a row in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

`xmldata` -- string with the file data

Returns:

true if successfull

xmlTableExists(Connection, String)

```
public boolean xmlTableExists(java.sql.Connection con, java.lang.String
tablename)
```

Checks if the XML table exists

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

true if the table exists

oracle.xml.transviewer XMLTransformPanel

Syntax

```
public class XMLTransformPanel extends javax.swing.JPanel

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.transviewer.XMLTransformPanel
```

All Implemented Interfaces:

javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable

Description

XMLTransformPanel visual bean. Applies XSL transformations on XML documents. Visualizes the result. Allows editing of input XML and XSL documents/files.

Member Summary	
Constructors	
XMLTransformPanel()	The class constructor.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

Inherited Member Summary

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBackground, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLTransformPanel()

```
public  XMLTransformPanel()
```

The class constructor. Creates an object of type `XMLTransformPanel`.

oracle.xml.transviewer XMLTransformPanelBeanInfo

Syntax

```
public class XMLTransformPanelBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.transviewer.XMLTransformPanelBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLTransformPanelBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLTransformPanelBeanInfo()

```
public  XMLTransformPanelBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.transviewer
XMLTransViewer

Syntax

```
public class XMLTransViewer extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.transviewer.XMLTransViewer
```

Description

Simple application that uses XMLTransformPanel. Can be used from the command line to edit and parse XML files, edit and apply XSL transformations and retrieve and save XML, XSL and result files in the file system or in the Oracle 8i database

Member Summary

Constructors

XMLTransViewer()

Methods

main(String)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XMLTransViewer()

```
public  XMLTransViewer()
```

Methods

main(String)

```
public static void main(java.lang.String[] args)
```

oracle.xml.treeviewer Package

Description

Class Summary

Classes	
XMLTreeView	Shows an XML document as a tree.
XMLTreeViewBeanInfo	

oracle.xml.treeviewer XMLTreeView

Syntax

```
public class XMLTreeView extends javax.swing.JPanel

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.treeviewer.XMLTreeView
```

All Implemented Interfaces:

javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable

Description

Shows an XML document as a tree. Recognizes the following XML DOM nodes: `Tag, Attribute Name, Attribute Value, Comment, CDATA, PCDATA, PI Data, PI Name and NOTATION Symbol`. Takes as input an `org.w3c.dom.Document` object.

Member Summary

Constructors	
XMLTreeView()	The class constructor.
Methods	
getPreferredSize()	Returns the XMLTreeView preffered size.
setXMLDocument(Document)	Associates the XMLTreeViewer with a XML document.
updateUI()	Forces the XMLTreeView to update/refresh UI.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

Inherited Member Summary

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBackground, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLTreeView()

```
public  XMLTreeView()
```

The class constructor. Creates an object of type `XMLTreeView`.

Methods

getPreferredSize()

```
public java.awt.Dimension getPreferredSize()
```

Returns the XMLTreeView preffered size.

Overrides:

javax.swing.JComponent.getPreferredSize() in class javax.swing.JComponent

Returns:

The `Dimension` object containing the XMLTreeView prefered size.

setXMLDocument(Document)

```
public void setXMLDocument(org.w3c.dom.Document document)
```

Associates the XMLTreeViewer with a XML document.

Parameters:

`doc` - The `Document` document to display.

updateUI()

```
public void updateUI()
```

Forces the XMLTreeView to update/refresh UI.

Overrides:

javax.swing.JPanel.updateUI() in class javax.swing.JPanel

oracle.xml.treeviewer XMLTreeViewBeanInfo

Syntax

```
public class XMLTreeViewBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.treeviewer.XMLTreeViewBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLTreeViewBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLTreeViewBeanInfo()

```
public  XMLTreeViewBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.srcviewer Package

Description

Class Summary

Classes

oracle.xml.srcviewer.XMLSourceView	Shows an XML document.
XMLSourceViewBeanInfo	

oracle.xml.srcviewer.XMLSourceView

Syntax

```
public class XMLSourceView extends javax.swing.JPanel implements
java.io.Serializable

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.srcviewer.XMLSourceView
```

All Implemented Interfaces:

`javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable`

Description

Shows an XML document. Recognizes the following XML token types: `Tag, Attribute Name, Attribute Value, Comment, CDATA, PCDATA, PI Data, PI Name and NOTATION Symbol`. Each token type has a foreground color and font. The default color/font settings can be changed by the user. Takes as input an org.w3c.dom.Document object.

Member Summary

Constructors	
XMLSourceView()	The class constructor.
Methods	
fontGet(AttributeSet)	Extracts and returns the font from a given attributeset.
fontSet(MutableAttributeSet, Font)	Sets the mutableattributeset font.
getAttributeNameFont()	Returns the Attribute Value font.
getAttributeNameForeground()	Returns the Attribute Name foreground color.

Member Summary	
getAttributeValueFont()	Returns the Attribute Value font.
getAttributeValueForeground()	Returns the Attribute Value foreground color.
getBackground()	Returns the background color.
getCDATAFont()	Returns the CDATA font.
getCDATAForeground()	Returns the CDATA foreground color.
getCommentDataFont()	Returns the Comment Data font.
getCommentDataForeground()	Returns the Comment Data foreground color.
getEditedText()	Returns the edited text.
getJTextPane()	Returns the viewer `JTextPane` component.
getMinimumSize()	Returns the XMLSourceView minimal size.
getNodeAtOffset(int)	Returns the XML node at a given offset.
getPCDATAFont()	Returns the PCDATA font.
getPCDATAForeground()	Returns the PCDATA foreground color.
getPIDataFont()	Returns the PI Data font.
getPIDataForeground()	Returns the PI Data foreground color.
getPINameFont()	Returns the PI Name font.
getPINameForeground()	Returns the PI Data foreground color.
getSymbolFont()	Returns the NOTATION Symbol font.
getSymbolForeground()	Returns the NOTATION Symbol foreground color.
getTagFont()	Returns the Tag font.
getTagForeground()	Returns the Tag foreground color.
getText()	Returns the XML document as a String.
isEditable()	Returns boolean to indicate whether this object is editable.
selectNodeAt(int)	Moves the cursor to XML Node at offset `i`.
setAttributeNameFont(Font)	Sets the Attribute Name font.
setAttributeNameForeground(Color)	Sets the Attribute Name foreground color.
setAttributeValueFont(Font)	Sets the Attribute Value font.

Member Summary	
setAttributeValueForeground(Color)	Sets the Attribute Value foreground color.
setBackground(Color)	Sets the background color.
setCDATAFont(Font)	Sets the CDATA font.
setCDATAForeground(Color)	Sets the CDATA foreground color.
setCommentDataFont(Font)	Sets the Comment font.
setCommentDataForeground(Color)	Sets the Comment foreground color.
setEditable(boolean)	Sets the specified boolean to indicate whether this object should be editable.
setPCDATAFont(Font)	Sets the PCDATA font.
setPCDATAForeground(Color)	Sets the PCDATA foreground color.
setPIDataFont(Font)	Sets the PI Data font.
setPIDataForeground(Color)	Sets the PI Data foreground color.
setPINameFont(Font)	Sets the PI Name font.
setPINameForeground(Color)	Sets the PI Name foreground color.
setSelectedNode(Node)	Sets the cursor position at the selected XML node.
setSymbolFont(Font)	Sets the NOTATION Symbol font.
setSymbolForeground(Color)	Sets the NOTATION Symbol foreground color.
setTagFont(Font)	Sets the Tag font.
setTagForeground(Color)	Sets the Tag foreground color.
setXMLDocument(Document)	Associates the XMLviewer with a XML document.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

Inherited Member Summary

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getNextFocusableComponent, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Inherited Member Summary

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLSourceView()

`public  XMLSourceView()`
The class constructor. Creates an object of type `XMLSourceView`.

Methods

fontGet(AttributeSet)

`public static java.awt.Font fontGet(javax.swing.text.AttributeSet attributeset)`
Extracts and returns the font from a given attributeset.

Parameters:

`attributeset` - The source `Attributeset`.

Returns:

The extracted `Font`.

fontSet(MutableAttributeSet, Font)

`public static void fontSet(javax.swing.text.MutableAttributeSet mutableattributeset, java.awt.Font font)`
Sets the mutableattributeset font.

Parameters:

`mutableattributeset` - The `mutableattributeset` to update.

`font` - The new `Font` for the mutableattributeset.

getAttributeNameFont()

`public java.awt.Font getAttributeNameFont()`
Returns the Attribute Value font.

Returns:

The `Font` object.

getAttributeNameForeground()

`public java.awt.Color getAttributeNameForeground()`
Returns the Attribute Name foreground color.

Returns:

The `Color` object.

getAttributeValueFont()

`public java.awt.Font getAttributeValueFont()`
Returns the Attribute Value font.

Returns:

The `Font` object.

getAttributeValueForeground()

`public java.awt.Color getAttributeValueForeground()`
Returns the Attribute Value foreground color.

Returns:

The `Color` object.

getBackground()

`public java.awt.Color getBackground()`
Returns the background color.

Overrides:

java.awt.Component.getBackground() in class java.awt.Component

Returns:

The `Color` object.

getCDATAFont()

`public java.awt.Font getCDATAFont()`

Returns the CDATA font.

Returns:

The `Font` object.

getCDATAForeground()

`public java.awt.Color getCDATAForeground()`

Returns the CDATA foreground color.

Returns:

The `Color` object.

getCommentDataFont()

`public java.awt.Font getCommentDataFont()`

Returns the Comment Data font.

Returns:

The `Font` object.

getCommentDataForeground()

`public java.awt.Color getCommentDataForeground()`

Returns the Comment Data foreground color.

Returns:

The `Color` object.

getEditedText()

`public java.lang.String getEditedText()`

Returns the edited text.

Returns:

The `String` object containing the edited text.

getJTextPane()

`public javax.swing.JTextPane getJTextPane()`
Returns the viewer `JTextPane` component.

Returns:

The `JTextPane` object used by XMLSourceViewer

getMinimumSize()

`public java.awt.Dimension getMinimumSize()`
Returns the XMLSourceView minimal size.

Overrides:

javax.swing.JComponent.getMinimumSize() in class javax.swing.JComponent

Returns:

The `Dimension` object containing the XMLSourceView minimum size.

getNodeAtOffset(int)

`public org.w3c.dom.Node getNodeAtOffset(int i)`
Returns the XML node at a given offset.

Parameters:

`i` - The node offset.

Returns:

The `Node` object from offset `i`.

getPCDATAFont()

`public java.awt.Font getPCDATAFont()`
Returns the PCDATA font.

Returns:

The `Font` object.

getPCDATAForeground()

`public java.awt.Color getPCDATAForeground()`
Returns the PCDATA foreground color.

Returns:

The `Color` object.

getPIDataFont()

`public java.awt.Font getPIDataFont()`
Returns the PI Data font.

Returns:

The `Font` object

getPIDataForeground()

`public java.awt.Color getPIDataForeground()`
Returns the PI Data foreground color.

Returns:

The `Color` object.

getPINameFont()

`public java.awt.Font getPINameFont()`
Returns the PI Name font.

Returns:

The `Font` object.

getPINameForeground()

`public java.awt.Color getPINameForeground()`
Returns the PI Data foreground color.

Returns:

The `Color` object.

getSymbolFont()

`public java.awt.Font getSymbolFont()`
Returns the NOTATION Symbol font.

Returns:

The `Font` object.

getSymbolForeground()

`public java.awt.Color getSymbolForeground()`
Returns the NOTATION Symbol foreground color.

Returns:

The `Color` object.

getTagFont()

`public java.awt.Font getTagFont()`
Returns the Tag font.

Returns:

The `Font` object.

getTagForeground()

`public java.awt.Color getTagForeground()`
Returns the Tag foreground color.

Returns:

The `Color` object.

getText()

`public java.lang.String getText()`
Returns the XML document as a String.

Returns:

The `String` object containing the XML document.

isEditable()

`public boolean isEditable()`

Returns boolean to indicate whether this object is editable.

selectNodeAt(int)

`public void selectNodeAt(int i)`

Moves the cursor to XML Node at offset `i`.

Parameters:

`i` - The node offset.

setAttributeNameFont(Font)

`public void setAttributeNameFont(java.awt.Font font)`

Sets the Attribute Name font.

Parameters:

`font` - The new `Font` for Attribute Name.

setAttributeNameForeground(Color)

`public void setAttributeNameForeground(java.awt.Color color)`

Sets the Attribute Name foreground color.

Parameters:

`color` - The new `Color` for Attribute Name.

setAttributeValueFont(Font)

`public void setAttributeValueFont(java.awt.Font font)`

Sets the Attribute Value font.

Parameters:

`font` - The new `Font` for Attribute Value.

setAttributeValueForeground(Color)

`public void setAttributeValueForeground(java.awt.Color color)`

Sets the Attribute Value foreground color.

Parameters:

`color` - The new `Color` for Attribute Value.

setBackground(Color)

`public void setBackground(java.awt.Color color)`
Sets the background color.

Overrides:

javax.swing.JComponent.setBackground(java.awt.Color) in class javax.swing.JComponent

Parameters:

`color` - The new background `Color`.

setCDATAFont(Font)

`public void setCDATAFont(java.awt.Font font)`
Sets the CDATA font.

Parameters:

`font` - The new `Font` for CDATA.

setCDATAForeground(Color)

`public void setCDATAForeground(java.awt.Color color)`
Sets the CDATA foreground color.

Parameters:

`color` - The new `Color` for CDATA.

setCommentDataFont(Font)

`public void setCommentDataFont(java.awt.Font font)`
Sets the Comment font.

Parameters:

`font` - The new `Font` for the XML Comments.

setCommentDataForeground(Color)

`public void setCommentDataForeground(java.awt.Color color)`
Sets the Comment foreground color.

Parameters:

`color` - The new `Color` for Comment.

setEditable(boolean)

`public void setEditable(boolean edit)`
Sets the specified boolean to indicate whether this object should be editable.

Parameters:

`doc` - The new `boolean` value.

setPCDATAFont(Font)

`public void setPCDATAFont(java.awt.Font font)`
Sets the PCDATA font.

Parameters:

`font` - The new `Font` for PCDATA.

setPCDATAForeground(Color)

`public void setPCDATAForeground(java.awt.Color color)`
Sets the PCDATA foreground color.

Parameters:

`color` - The new `Color` for PCDATA.

setPIDataFont(Font)

`public void setPIDataFont(java.awt.Font font)`
Sets the PI Data font.

Parameters:

`font` - The new `Font` for PI Data.

setPIDataForeground(Color)

`public void setPIDataForeground(java.awt.Color color)`
Sets the PI Data foreground color.

Parameters:

`color` - The new `Color` for PI Data.

setPINameFont(Font)

`public void setPINameFont(java.awt.Font font)`
Sets the PI Name font.

Parameters:

`font` - The new `Font` for the PI Names.

setPINameForeground(Color)

`public void setPINameForeground(java.awt.Color color)`
Sets the PI Name foreground color.

Parameters:

`color` - The new `Color` for PI Name.

setSelectedNode(Node)

`public void setSelectedNode(org.w3c.dom.Node node)`
Sets the cursor position at the selected XML node.

Parameters:

`node` - The selected node.

setSymbolFont(Font)

`public void setSymbolFont(java.awt.Font font)`
Sets the NOTATION Symbol font.

Parameters:

`color` - The new `Font` for NOTATION Symbol.

setSymbolForeground(Color)

`public void setSymbolForeground(java.awt.Color color)`
Sets the NOTATION Symbol foreground color.

Parameters:

`color` - The new `Color` for NOTATION Symbol.

setTagFont(Font)

`public void setTagFont(java.awt.Font font)`
Sets the Tag font.

Parameters:

`font` - The new `Font` for the XML Tags.

setTagForeground(Color)

`public void setTagForeground(java.awt.Color color)`
Sets the Tag foreground color.

Parameters:

`color` - The new `Color` for the XML Tags.

setXMLDocument(Document)

`public void setXMLDocument(org.w3c.dom.Document document)`
Associates the XMLviewer with a XML document.

Parameters:

`doc` - The `Document` document to display.

See Also:

getText()

oracle.xml.srcviewer XMLSourceViewBeanInfo

Syntax

```
public class XMLSourceViewBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.srcviewer.XMLSourceViewBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLSourceViewBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLSourceViewBeanInfo()

```
public  XMLSourceViewBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async Package

Description

Class Summary	
Interfaces	
DOMBuilderErrorListener	This interface must be implemented in order to receive notifications when error is found during parsing.
DOMBuilderListener	This interface must be implemented in order to receive notifications about events during the asyncronous parsing.
XSLTransformerErrorListener	This interface must be implemented in order to receive notifications about error events during the asyncronous transformation.
XSLTransformerListener	This interface must be implemented in order to receive notifications about events during the asyncronous transformation.
Classes	
DOMBuilder	This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation.
DOMBuilderBeanInfo	
DOMBuilderErrorEvent	This class defines the error event which is sent when parse exception occurs.
DOMBuilderEvent	The event object that DOMBuilder uses to notify all registered listeners about parse events.
ResourceManager	Simple semaphore that maintains access to fixed number of logical resources.
XSLTransformer	Applies XSL transformation in a background thread.
XSLTransformerBeanInfo	
XSLTransformerErrorEvent	The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.
XSLTransformerEvent	The event object that XSLTransformer uses to notify all registered listeners about transformation events.

oracle.xml.async
DOMBuilder

Syntax

```
public class DOMBuilder extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.DOMBuilderConstants, java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.DOMBuilder
```

All Implemented Interfaces:

`oracle.xml.async.DOMBuilderConstants, java.lang.Runnable, java.io.Serializable`

Description

This class implements an eXtensible Markup Language (XML) 1.0 parser according to the World Wide Web Consortium (W3C) recommendation. to parse a XML document and build a DOM tree. The parsing is done in a separate thread and DOMBuilderListener interface must be used for notification when the tree is built.

Member Summary

Constructors	
DOMBuilder()	Creates a new parser object.
Creates a new parser object.	Creates a new parser object with a given id.
Methods	
addDOMBuilderErrorListener(DOMBuilderErrorListener)	Adds DOMBuilderErrorListener
addDOMBuilderListener(DOMBuilderListener)	Adds DOMBuilderListener
	Get the DTD
getDocument()	Gets the document
getId()	Returns the parser object id.
getReleaseVersion()	Returns the release version of the Oracle XML Parser

Member Summary

	Gets the document
getValidationMode()	Returns the validation mode
parse(InputSource)	Parses the XML from given input source
	Parses the XML from given input stream.
parse(Reader)	Parses the XML from given input stream.
parse(String)	Parses the XML from the URL indicated
parse(URL)	Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.
parseDTD(InputSource, String)	Parses the XML External DTD from given input source
parseDTD(InputStream, String)	Parses the XML External DTD from given input stream.
	Parses the XML External DTD from given input stream.
parseDTD(String, String)	Parses the XML External DTD from the URL indicated
parseDTD(URL, String)	Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.
removeDOMBuilderErrorListener(DOMBuilderErrorListener)	Remove DOMBuilderErrorListener
removeDOMBuilderListener(DOMBuilderListener)	Remove DOMBuilderListener
run()	This method runs in a thread
setBaseURL(URL)	Set the base URL for loading external enitites and DTDs.
setDebugMode(boolean)	Sets a flag to turn on debug information in the document
setDoctype(DTD)	Set the DTD
setErrorStream(OutputStream)	Creates an output stream for the output of errors and warnings.
setErrorStream(OutputStream, String)	Creates an output stream for the output of errors and warnings.
setErrorStream(PrintWriter)	Creates an output stream for the output of errors and warnings.
setNodeFactory(NodeFactory)	Set the node factory.
setPreserveWhitespace(boolean)	Set the white space preserving mode
setValidationMode(boolean)	Set the validation mode
showWarnings(boolean)	Switch to determine whether to print warnings

Inherited Member Summary

Fields inherited from interface oracle.xml.async.DOMBuilderConstants

DOM_GEN_ENDED, DOM_GEN_ERROR, DOM_GEN_STARTED, DOM_METHOD1, DOM_METHOD10, DOM_METHOD2, DOM_METHOD3, DOM_METHOD4, DOM_METHOD5, DOM_METHOD6, DOM_METHOD7, DOM_METHOD8, DOM_METHOD9

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

DOMBuilder()

`public  DOMBuilder()`
Creates a new parser object.

DOMBuilder(int)

`public  DOMBuilder(int id)`
Creates a new parser object with a given id.

Methods

addDOMBuilderErrorListener(DOMBuilderErrorListener)

`public void addDOMBuilderErrorListener(DOMBuilderErrorListener p0)`
Adds DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to add

addDOMBuilderListener(DOMBuilderListener)

`public void addDOMBuilderListener(DOMBuilderListener p0)`
Adds DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to add

getDoctype()

`public synchronized oracle.xml.async.DTD getDoctype()`
Get the DTD

Returns:

The `DTD`

getDocument()

`public synchronized oracle.xml.async.XMLDocument getDocument()`
Gets the document

Returns:

The document being parsed

getId()

`public int getId()`
Returns the parser object id.

getReleaseVersion()

`public synchronized java.lang.String getReleaseVersion()`
Returns the release version of the Oracle XML Parser

Returns:

the release version string

getResult()

`public synchronized org.w3c.dom.Document getResult()`
Gets the document

Returns:

The document being parsed

getValidationMode()

`public synchronized boolean getValidationMode()`
Returns the validation mode

Returns:

`true` if the XML parser is validating `false` if not

parse(InputSource)

`public final synchronized void parse(org.xml.sax.InputSource in)`
Parses the XML from given input source

Parameters:

`in` - the `org.xml.sax.InputSouce` to parse

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(InputStream)

`public final synchronized void parse(java.io.InputStream in)`
Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(Reader)

`public final synchronized void parse(java.io.Reader r)`
Parses the XML from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parse(String)

`public final synchronized void parse(java.lang.String in)`
Parses the XML from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parse(URL)

`public final synchronized void parse(java.net.URL url)`
Parses the XML document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputSource, String)

`public final synchronized void parseDTD(org.xml.sax.InputSource in, java.lang.String rootName)`

Parses the XML External DTD from given input source

Parameters:

`in` - the `org.xml.sax.InputSouce` to parse

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(InputStream, String)

`public final synchronized void parseDTD(java.io.InputStream in, java.lang.String rootName)`

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`in` - the `InputStream` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(Reader, String)

`public final synchronized void parseDTD(java.io.Reader r, java.lang.String rootName)`

Parses the XML External DTD from given input stream. The base URL should be set for resolving external entities and DTD.

Parameters:

`r` - the `Reader` containing XML data to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

See Also:

`oracle.xml.parser.v2.XMLParser`

parseDTD(String, String)

`public final synchronized void parseDTD(java.lang.String in, java.lang.String rootName)`

Parses the XML External DTD from the URL indicated

Parameters:

`in` - the `String` containing the URL to parse from

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

parseDTD(URL, String)

`public final synchronized void parseDTD(java.net.URL url, java.lang.String rootName)`

Parses the XML External DTD document pointed to by the given URL and creates the corresponding XML document hierarchy.

Parameters:

`url` - the url points to the XML document to parse.

`rootName` - the element to be used as root Element

Throws:

`XMLParseException` - if syntax or other error encountered.

`SAXException` - Any SAX exception, possibly wrapping another exception.

`IOException` - IO Error.

removeDOMBuilderErrorListener(DOMBuilderErrorListener)

```
public synchronized void removeDOMBuilderErrorListener(DOMBuilderErrorListener
p0)
```

Remove DOMBuilderErrorListener

Parameters:

`p1` - The `DOMBuilderErrorListener` to remove

removeDOMBuilderListener(DOMBuilderListener)

```
public synchronized void removeDOMBuilderListener(DOMBuilderListener p0)
```

Remove DOMBuilderListener

Parameters:

`p1` - The `DOMBuilderListener` to remove

run()

```
public void run()
```

This method runs in a thread

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setBaseURL(URL)

```
public synchronized void setBaseURL(java.net.URL url)
```

Set the base URL for loading external enitites and DTDs. This method should to be called if the parse(InputStream) is used to parse the XML Document

Parameters:

`url` - The base URL

setDebugMode(boolean)

```
public void setDebugMode(boolean flag)
```

Sets a flag to turn on debug information in the document

Parameters:

`flag` - determines whether debug info is stored

setDoctype(DTD)

```
public synchronized void setDoctype(oracle.xml.async.DTD dtd)
```

Set the DTD

Parameters:

`dtd` - `DTD` to set and used while parsing

setErrorStream(OutputStream)

```
public final synchronized void setErrorStream(java.io.OutputStream out)
```

Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The output stream to use for errors and warnings

setErrorStream(OutputStream, String)

```
public final synchronized void setErrorStream(java.io.OutputStream out,
java.lang.String enc)
```

Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings. Additionally, an .exception is thrown if the encoding specified is unsupported.

Parameters:

`out` - The output stream to use for errors and warnings

`enc` - the encoding to use

Throws:

`IOException` - if an unsupported encoding is specified

setErrorStream(PrintWriter)

```
public final synchronized void setErrorStream(java.io.PrintWriter out)
```

Creates an output stream for the output of errors and warnings. If an output stream for errors is not specified, the parser will use the standard error output stream `System.err` for outputting errors and warnings.

Parameters:

`out` - The `PrintWriter` to use for errors and warnings

setNodeFactory(NodeFactory)

```
public synchronized void setNodeFactory(oracle.xml.async.NodeFactory factory)
```

Set the node factory. Applications can extend the NodeFactory and register it through this method. The parser will then use the user supplied NodeFactory to create nodes of the DOM tree.

Parameters:

`factory` - The `NodeFactory` to set

Throws:

`XMLParseException` - if an invalid factory is set

See Also:

`NodeFactory`

setPreserveWhitespace(boolean)

```
public synchronized void setPreserveWhitespace(boolean flag)
```

Set the white space preserving mode

Parameters:

`flag` - preserving mode

setValidationMode(boolean)

```
public synchronized void setValidationMode(boolean yes)
```

Set the validation mode

Parameters:

`yes` - determines whether the XML parser should be validating

showWarnings(boolean)

```
public synchronized void showWarnings(boolean yes)
```

Switch to determine whether to print warnings

Parameters:

`yes` - determines whether warnings should be shown

oracle.xml.async
DOMBuilderBeanInfo

Syntax

```
public class DOMBuilderBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.DOMBuilderBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

DOMBuilderBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

Inherited Member Summary

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

DOMBuilderBeanInfo()

```
public  DOMBuilderBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async DOMBuilderErrorEvent

Syntax

```
public class DOMBuilderErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

This class defines the error event which is sent when parse exception occurs.

Member Summary

Constructors	
DOMBuilderErrorEvent(Object, Exception)	Constructor for DOMBuilderErrorEvent.
Methods	
getException()	Gets the Exception
getMessage()	Returns the error message generated by the parser

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderErrorEvent(Object, Exception)

`public  DOMBuilderErrorEvent(java.lang.Object p0, java.lang.Exception e)`
Constructor for DOMBuilderErrorEvent.

Parameters:

`p0` - the `Object` that create this object

`e` - the `Exception` raised.

Methods

getException()

`public java.lang.Exception getException()`
Gets the Exception

Returns:

The Exception beind raised

getMessage()

`public java.lang.String getMessage()`
Returns the error message generated by the parser

Returns:

the error message string

oracle.xml.async
DOMBuilderErrorListener

Syntax

```
public interface DOMBuilderErrorListener extends java.util.EventListener
```

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications when error is found during parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderErrorListener method.

Member Summary
Methods
domBuilderErrorCalled(DOMBuilderErrorEvent)

Methods

domBuilderErrorCalled(DOMBuilderErrorEvent)

```
public void domBuilderErrorCalled(DOMBuilderErrorEvent p0)
```

Parameters:

`p0` -- The DOMBuilderErrorEvent object produced by the DOMBuilder as result of parsing error

oracle.xml.async DOMBuilderEvent

Syntax

```
public class DOMBuilderEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.DOMBuilderEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that DOMBuilder uses to notify all registered listeners about parse events.

Member Summary	
Constructors	
DOMBuilderEvent(Object, int)	
Methods	
getID()	Returns the DOMBuilder object unique id.

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

DOMBuilderEvent(Object, int)

```
public  DOMBuilderEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the DOMBuilder object unique id. Can be used to identify which instance of the DOMBuilder generated this event in case when multiple instances of DOMBuilder work in the background.

Returns:

The event id

oracle.xml.async DOMBuilderListener

Syntax

```
public interface DOMBuilderListener extends java.util.EventListener
```

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about events during the asyncronous parsing. The class implementing this interface must be added to the DOMBuilder using addDOMBuilderListener method.

Member Summary

Methods	
domBuilderError(DOMBuilderEvent)	This method is called when parse error occur.
domBuilderOver(DOMBuilderEvent)	This method is called when the parse is complete
domBuilderStarted(DOMBuilderEvent)	This method is called when parse starts

Methods

domBuilderError(DOMBuilderEvent)

```
public void domBuilderError(DOMBuilderEvent p0)
```

This method is called when parse error occur.

Parameters:

`p0` -- The DOMBuilderEvent object produced by the DOMBuilder

domBuilderOver(DOMBuilderEvent)

```
public void domBuilderOver(DOMBuilderEvent p0)
```

This method is called when the parse is complete

Parameters:

`p0` -- The DOMBuilderEvent object produced by the DOMBuilder

domBuilderStarted(DOMBuilderEvent)

```
public void domBuilderStarted(DOMBuilderEvent p0)
```
This method is called when parse starts

Parameters:

`p0` -- The DOMBuilderEvent object produced by the DOMBuilder

oracle.xml.async
ResourceManager

Syntax

```
public class ResourceManager extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.async.ResourceManager
```

Description

Simple semaphore that maintains access to fixed number of logical resources.

Member Summary	
Constructors	
ResourceManager(int)	The ResourceManager constructor
Methods	
activeFound()	Returns how many resources are avialable.
getResource()	If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns.
releaseResource()	Releases a resource.
sleep(int)	Let's you use Thread.sleep() without try/catch

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ResourceManager(int)

`public  ResourceManager(int i)`
The ResourceManager constructor

Parameters:

`i` -- the number of resources to manage

Methods

activeFound()

`public boolean activeFound()`
Returns how many resources are avialable.

getResource()

`public synchronized void getResource()`
If the number of resources avialable is nonzero, the method decreases the number of resources avialable by one and returns. Othervise, waits until resource is released.

releaseResource()

`public void releaseResource()`
Releases a resource. When this method is called, the number of resources avialable is increased by one.

sleep(int)

`public void sleep(int i)`
Let's you use Thread.sleep() without try/catch

oracle.xml.async
XSLTransformer

Syntax

```
public class XSLTransformer extends java.lang.Object implements
java.io.Serializable, oracle.xml.async.XSLTransformerConstants,
java.lang.Runnable

java.lang.Object
  |
  +--oracle.xml.async.XSLTransformer
```

All Implemented Interfaces:

`java.lang.Runnable, java.io.Serializable, oracle.xml.async.XSLTransformerConstants`

Description

Applies XSL transformation in a background thread.

Member Summary

Constructors	
XSLTransformer()	XSLTransformet constructor
XSLTransformer(int)	XSLTransformer constructor
Methods	
addXSLTransformerErrorListener(XSLTransformerErrorListener)	Adds an error event listener
addXSLTransformerListener(XSLTransformerListener)	Adds a listener
	Returns the unique XSLTransformer id
getResult()	Returns the document fragment for the resulting document.
processXSL(XSLStylesheet, InputStream, URL)	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, Reader, URL)	Initiate XSL Transformation in the background.

Member Summary	
processXSL(XSLStylesheet, URL, URL)	Initiate XSL Transformation in the background.
	Initiate XSL Transformation in the background.
processXSL(XSLStylesheet, XMLDocument, OutputStream)	Initiate XSL Transformation in the background.
removeDOMTransformerErrorListener (XSLTransformerErrorListener)	Renmoves an error event listener
removeXSLTransformerListener(XSLTransformerListener)	Removes a listener
run()	
	Sets the error stream used by the XSL processor
showWarnings(boolean)	Sets the showWarnings flage used by the XSL processor

Inherited Member Summary
Fields inherited from interface oracle.xml.async.XSLTransformerConstants
XSL_GEN_ENDED, XSL_GEN_ERROR, XSL_GEN_STARTED
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XSLTransformer()

`public  XSLTransformer()`
XSLTransformet constructor

XSLTransformer(int)

`public  XSLTransformer(int id)`
XSLTransformer constructor

Parameters:

`id` -- A unique integer that can be used to identify the XSLTransformer instance during event processing

Methods

addXSLTransformerErrorListener(XSLTransformerErrorListener)

`public void addXSLTransformerErrorListener(XSLTransformerErrorListener p0)`
Adds an error event listener

Parameters:

`p0` -- XSLTransformerErrorListener

addXSLTransformerListener(XSLTransformerListener)

`public void addXSLTransformerListener(XSLTransformerListener p0)`
Adds a listener

Parameters:

`p0` -- XSLTransformerListener

getId()

`public int getId()`
Returns the unique XSLTransformer id

Returns:

The id

getResult()

`public synchronized oracle.xml.async.DocumentFragment getResult()`
Returns the document fragment for the resulting document. Call this method only after receiving notification that the transformation is complete. Since the transformation occurs in background and asyncronously, calling this method immediately after processXSL will result in holding the control until tre result is avialable.

processXSL(XSLStylesheet, InputStream, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.InputStream xml, java.net.URL ref)`
Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, Reader, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.io.Reader xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, URL, URL)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, java.net.URL xml, java.net.URL ref)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml)`

Initiate XSL Transformation in the background. The control is returned immediately.

processXSL(XSLStylesheet, XMLDocument, OutputStream)

`public void processXSL(oracle.xml.async.XSLStylesheet xsl, oracle.xml.async.XMLDocument xml, java.io.OutputStream os)`

Initiate XSL Transformation in the background. The control is returned immediately.

removeDOMTransformerErrorListener(XSLTransformerErrorListener)

`public synchronized void removeDOMTransformerErrorListener(XSLTransformerErrorListener p0)`

Renmoves an error event listener

Parameters:

`p0` - - XSLTransformerErrorListener

removeXSLTransformerListener(XSLTransformerListener)

`public synchronized void removeXSLTransformerListener(XSLTransformerListener p0)`

Removes a listener

Parameters:

`p0` - - XSLTransformerListener

run()

`public void run()`

Specified By:

java.lang.Runnable.run() in interface java.lang.Runnable

setErrorStream(OutputStream)

```
public final void setErrorStream(java.io.OutputStream out)
```

Sets the error stream used by the XSL processor

showWarnings(boolean)

```
public final void showWarnings(boolean yes)
```

Sets the showWarnings flage used by the XSL processor

oracle.xml.async XSLTransformerBeanInfo

Syntax

```
public class XSLTransformerBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.async.XSLTransformerBeanInfo
```

All Implemented Interfaces:

java.beans.BeanInfo

Description

Member Summary

Constructors

XSLTransformerBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XSLTransformerBeanInfo()

```
public  XSLTransformerBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.async XSLTransformerErrorEvent

Syntax

```
public class XSLTransformerErrorEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerErrorEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The error event object that XSLTransformer uses to notify all registered listeners about transformation error events.

Member Summary	
Constructors	
XSLTransformerErrorEvent(Object, Exception)	
Methods	
getException()	Returns the exception that XSLTransformer encountered object unique id.
getMessage()	Returns the error message that describes the error that XSLTransformer encountered

Inherited Member Summary
Methods inherited from class java.util.EventObject
getSource, toString
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerErrorEvent(Object, Exception)

```
public  XSLTransformerErrorEvent(java.lang.Object p0, java.lang.Exception e)
```

Methods

getException()

```
public java.lang.Exception getException()
```

Returns the exception that XSLTransformer encountered object unique id. Can be used to

Returns:

The transformation exception

getMessage()

```
public java.lang.String getMessage()
```

Returns the error message that describes the error that XSLTransformer encountered

Returns:

The error message

oracle.xml.async XSLTransformerErrorListener

Syntax

`public interface XSLTransformerErrorListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about error events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet error event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerErrorCalled(XSLTransformerErrorEvent)	This method is called when parse or transformation error occur.

Methods

xslTransformerErrorCalled(XSLTransformerErrorEvent)

`public void xslTransformerErrorCalled(XSLTransformerErrorEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` - - The XSLTransformerErrorEvent object produced by the XSLTransformer

oracle.xml.async XSLTransformerEvent

Syntax

```
public class XSLTransformerEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--oracle.xml.async.XSLTransformerEvent
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The event object that XSLTransformer uses to notify all registered listeners about transformation events.

Member Summary

Constructors	
XSLTransformerEvent(Object, int)	
Methods	
getID()	Returns the XSLTransformer object unique id.

Inherited Member Summary

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

XSLTransformerEvent(Object, int)

```
public  XSLTransformerEvent(java.lang.Object p0, int p1)
```

Methods

getID()

```
public int getID()
```

Returns the XSLTransformer object unique id. Can be used to identify which instance of the XSLTransformer generated this event in case when multiple instances of XSLTransformer work in the background.

Returns:

The event id

oracle.xml.async XSLTransformerListener

Syntax

`public interface XSLTransformerListener extends java.util.EventListener`

All Superinterfaces:

`java.util.EventListener`

Description

This interface must be implemented in order to receive notifications about events during the asyncronous transformation. The class implementing this interface must be added to the XSLTransformet event queue using addXSLTransformerListener method.

Member Summary

Methods	
xslTransformerError(XSLTransformerEvent)	This method is called when parse or transformation error occur.
xslTransformerOver(XSLTransformerEvent)	This method is called when the transformation is complete
xslTransformerStarted(XSLTransformerEvent)	This method is called when the transformation starts

Methods

xslTransformerError(XSLTransformerEvent)

`public void xslTransformerError(XSLTransformerEvent p0)`

This method is called when parse or transformation error occur.

Parameters:

`p0` -- The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerOver(XSLTransformerEvent)

`public void xslTransformerOver(XSLTransformerEvent p0)`
This method is called when the transformation is complete

Parameters:

`p0` - - The XSLTransformerEvent object produced by the XSLTransformer

xslTransformerStarted(XSLTransformerEvent)

`public void xslTransformerStarted(XSLTransformerEvent p0)`
This method is called when the transformation starts

Parameters:

`p0` - - The XSLTransformerEvent object produced by the XSLTransformer

oracle.xml.transviewer Package

Description

Class Summary	
Classes	
DBAccess	Maintains CLOB tables that can hold multiple XML and text documents.
XMLTransformPanel	XMLTransformPanel visual bean.
XMLTransformPanelBeanInfo	
XMLTransViewer	Simple application that uses XMLTransformPanel.

oracle.xml.transviewer DBAccess

Syntax

```
public class DBAccess extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.transviewer.DBAccess
```

Description

Maintains CLOB tables that can hold multiple XML and text documents. Each table is created using the statement: CREATE TABLE tablename FILENAME CHAR(16) UNIQUE, FILEDATA CLOB) LOB(FILEDATA) STORE AS (DISABLE STORAGE IN ROW). Each XML (or text) document is stored as a row in the table and the FILENAME field holds a unique string that is used as a key to retrieve, update or delete the row. The document text is stored in the FILEDATA field that is a CLOB object. This CLOB tables are automatically maintained by the transviewer bean. The CLOB tables maintained by this class can be later used by the transviewer bean. The class creates and deletes CLOB tables, list a CLOB table content and also add, replace or delete text documents in this CLOB tables.

Member Summary

Constructors	
DBAccess()	
Methods	
createXMLTable(Connection, String)	Create XML table
deleteXMLName(Connection, String, String)	Delete text file from XML table
dropXMLTable(Connection, String)	Delete XML table
getNameSize()	Returns the size of the field where the filename is kept.
getXMLData(Connection, String, String)	Retrieve text file from XML table
getXMLNames(Connection, String)	Returns all file names in XML table

Member Summary	
getXMLTableNames(Connection, String)	Gets all XML tables with names starting with a given string
insertXMLData(Connection, String, String, String)	Inserts text file as a row in XML table
isXMLTable(Connection, String)	Check if the table is XML table.
replaceXMLData(Connection, String, String, String)	Replace text file as a row in XML table
xmlTableExists(Connection, String)	Checks if the XML table exists

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

DBAccess()

```
public  DBAccess()
```

Methods

createXMLTable(Connection, String)

```
public boolean createXMLTable(java.sql.Connection con, java.lang.String
tablename)
```

Create XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

true if successfull

deleteXMLName(Connection, String, String)

`public boolean deleteXMLName(java.sql.Connection con, java.lang.String tablename, java.lang.String xmlname)`

Delete text file from XML table

Parameters:

`con` - - the Connection object

`tablename` - - the table name

`xmlname` - - the file name

Returns:

true if successfull

dropXMLTable(Connection, String)

`public boolean dropXMLTable(java.sql.Connection con, java.lang.String tablename)`

Delete XML table

Parameters:

`con` - - the Connection object

`tablename` - - the table name

Returns:

true if successfull

getNameSize()

`public int getNameSize()`

Returns the size of the field where the filename is kept.

Returns:

filename size

getXMLData(Connection, String, String)

`public java.lang.String getXMLData(java.sql.Connection con, java.lang.String tablename, java.lang.String xmlname)`

Retrieve text file from XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

Returns:

file as a string

getXMLNames(Connection, String)

`public java.lang.String[] getXMLNames(java.sql.Connection con, java.lang.String tablename)`

Returns all file names in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

String array with all file names in this table

getXMLTableNames(Connection, String)

`public java.lang.String[] getXMLTableNames(java.sql.Connection con, java.lang.String tablePrefix)`

Gets all XML tables with names starting with a given string

Parameters:

`con` -- the Connection object

`tablePrefix` -- table prefix string

Returns:

array of all XML tables that begin with tablePrefix

insertXMLData(Connection, String, String, String)

`public boolean insertXMLData(java.sql.Connection con, java.lang.String tablename, java.lang.String xmlname, java.lang.String xmldata)`

Inserts text file as a row in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

`xmldata` -- string with the file data

Returns:

true if successfull

isXMLTable(Connection, String)

```
public boolean isXMLTable(java.sql.Connection con, java.lang.String tablename)
```

Check if the table is XML table.

Parameters:

`con` -- the Connection object

`tableName` -- the table name to test

Returns:

true if this is XML table

replaceXMLData(Connection, String, String, String)

```
public boolean replaceXMLData(java.sql.Connection con, java.lang.String
tablename, java.lang.String xmlname, java.lang.String xmldata)
```

Replace text file as a row in XML table

Parameters:

`con` -- the Connection object

`tablename` -- the table name

`xmlname` -- the file name

`xmldata` -- string with the file data

Returns:

true if successfull

xmlTableExists(Connection, String)

`public boolean xmlTableExists(java.sql.Connection con, java.lang.String tablename)`

Checks if the XML table exists

Parameters:

`con` -- the Connection object

`tablename` -- the table name

Returns:

true if the table exists

oracle.xml.transviewer XMLTransformPanel

Syntax

```
public class XMLTransformPanel extends javax.swing.JPanel

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.transviewer.XMLTransformPanel
```

All Implemented Interfaces:

`javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable`

Description

XMLTransformPanel visual bean. Applies XSL transformations on XML documents. Visualizes the result. Allows editing of input XML and XSL documents/files.

Member Summary	
Constructors	
XMLTransformPanel()	The class constructor.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Inherited Member Summary

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

Inherited Member Summary

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBackground, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLTransformPanel()

`public XMLTransformPanel()`

The class constructor. Creates an object of type `XMLTransformPanel`.

oracle.xml.transviewer XMLTransformPanelBeanInfo

Syntax

```
public class XMLTransformPanelBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.transviewer.XMLTransformPanelBeanInfo
```

All Implemented Interfaces:

java.beans.BeanInfo

Description

Member Summary

Constructors

XMLTransformPanelBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLTransformPanelBeanInfo()

```
public  XMLTransformPanelBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.transviewer
XMLTransViewer

Syntax

```
public class XMLTransViewer extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.transviewer.XMLTransViewer
```

Description

Simple application that uses XMLTransformPanel. Can be used from the command line to edit and parse XML files, edit and apply XSL transformations and retrieve and save XML, XSL and result files in the file system or in the Oracle 8i database

Member Summary

Constructors

XMLTransViewer()

Methods

main(String)

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XMLTransViewer()

```
public  XMLTransViewer()
```

Methods

main(String[])

```
public static void main(java.lang.String[] args)
```

oracle.xml.treeviewer Package

Description

Class Summary

Classes	
XMLTreeView	Shows an XML document as a tree.
XMLTreeViewBeanInfo	

oracle.xml.treeviewer XMLTreeView

Syntax

```
public class XMLTreeView extends javax.swing.JPanel

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.treeviewer.XMLTreeView
```

All Implemented Interfaces:

javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable

Description

Shows an XML document as a tree. Recognizes the following XML DOM nodes: `Tag, Attribute Name, Attribute Value, Comment, CDATA, PCDATA, PI Data, PI Name and NOTATION Symbol`. Takes as input an `org.w3c.dom.Document` object.

Member Summary	
Constructors	
XMLTreeView()	The class constructor.
Methods	
getPreferredSize()	Returns the XMLTreeView preffered size.
setXMLDocument(Document)	Associates the XMLTreeViewer with a XML document.
updateUI()	Forces the XMLTreeView to update/refresh UI.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

Inherited Member Summary

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBackground, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLTreeView()

```
public  XMLTreeView()
```

The class constructor. Creates an object of type `XMLTreeView`.

Methods

getPreferredSize()

```
public java.awt.Dimension getPreferredSize()
```

Returns the XMLTreeView preffered size.

Overrides:

javax.swing.JComponent.getPreferredSize() in class javax.swing.JComponent

Returns:

The `Dimension` object containing the XMLTreeView prefered size.

setXMLDocument(Document)

`public void setXMLDocument(org.w3c.dom.Document document)`
Associates the XMLTreeViewer with a XML document.

Parameters:

`doc` - The `Document` document to display.

updateUI()

`public void updateUI()`
Forces the XMLTreeView to update/refresh UI.

Overrides:

javax.swing.JPanel.updateUI() in class javax.swing.JPanel

oracle.xml.treeviewer XMLTreeViewBeanInfo

Syntax

```
public class XMLTreeViewBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.treeviewer.XMLTreeViewBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLTreeViewBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLTreeViewBeanInfo()

```
public  XMLTreeViewBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

oracle.xml.treeviewer Package

Description

Class Summary	
Classes	
XMLTreeView	Shows an XML document as a tree.
XMLTreeViewBeanInfo	

oracle.xml.treeviewer XMLTreeView

Syntax

```
public class XMLTreeView extends javax.swing.JPanel

java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--oracle.xml.treeviewer.XMLTreeView
```

All Implemented Interfaces:

`javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable`

Description

Shows an XML document as a tree. Recognizes the following XML DOM nodes: `Tag, Attribute Name, Attribute Value, Comment, CDATA, PCDATA, PI Data, PI Name and NOTATION Symbol`. Takes as input an `org.w3c.dom.Document` object.

Member Summary	
Constructors	
XMLTreeView()	The class constructor.
Methods	
getPreferredSize()	Returns the XMLTreeView preffered size.
setXMLDocument(Document)	Associates the XMLTreeViewer with a XML document.
updateUI()	Forces the XMLTreeView to update/refresh UI.

Inherited Member Summary

Inner classes inherited from class javax.swing.JComponent

javax.swing.JComponent.AccessibleJComponent

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUIClassID

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getActionForKeyStroke, getAlignmentX, getAlignmentY, getAutoscrolls, getBorder, getBounds, getClientProperty, getConditionForKeyStroke, getDebugGraphicsOptions, getGraphics, getHeight, getInsets, getInsets, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getVisibleRect, getWidth, getX, getY, grabFocus, hasFocus, isDoubleBuffered, isFocusCycleRoot, isFocusTraversable, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setDebugGraphicsOptions, setDoubleBuffered, setEnabled, setFont, setForeground, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getLayout, insets, invalidate, isAncestorOf, layout, list, list, locate, minimumSize, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, setCursor, setLayout, validate

Methods inherited from class java.awt.Component

Inherited Member Summary

action, add, addComponentListener, addFocusListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, bounds, checkImage, checkImage, contains, createImage, createImage, disable, dispatchEvent, enable, enable, enableInputMethods, getBackground, getBounds, getColorModel, getComponentOrientation, getCursor, getDropTarget, getFont, getFontMetrics, getForeground, getInputContext, getInputMethodRequests, getLocale, getLocation, getLocationOnScreen, getName, getParent, getPeer, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hide, imageUpdate, inside, isDisplayable, isEnabled, isLightweight, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, printAll, remove, removeComponentListener, removeFocusListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setDropTarget, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface javax.accessibility.Accessible

getAccessibleContext

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Constructors

XMLTreeView()

`public  XMLTreeView()`

The class constructor. Creates an object of type `XMLTreeView`.

Methods

getPreferredSize()

`public java.awt.Dimension getPreferredSize()`

Returns the XMLTreeView preffered size.

Overrides:

javax.swing.JComponent.getPreferredSize() in class javax.swing.JComponent

Returns:

The `Dimension` object containing the XMLTreeView prefered size.

setXMLDocument(Document)

`public void setXMLDocument(org.w3c.dom.Document document)`

Associates the XMLTreeViewer with a XML document.

Parameters:

`doc` - The `Document` document to display.

updateUI()

`public void updateUI()`

Forces the XMLTreeView to update/refresh UI.

Overrides:

javax.swing.JPanel.updateUI() in class javax.swing.JPanel

oracle.xml.treeviewer XMLTreeViewBeanInfo

Syntax

```
public class XMLTreeViewBeanInfo extends java.beans.SimpleBeanInfo

java.lang.Object
  |
  +--java.beans.SimpleBeanInfo
        |
        +--oracle.xml.treeviewer.XMLTreeViewBeanInfo
```

All Implemented Interfaces:

`java.beans.BeanInfo`

Description

Member Summary

Constructors

XMLTreeViewBeanInfo()

Methods

getIcon(int)

getPropertyDescriptors()

Inherited Member Summary

Fields inherited from interface java.beans.BeanInfo

ICON_COLOR_16x16, ICON_COLOR_32x32, ICON_MONO_16x16, ICON_MONO_32x32

Methods inherited from class java.beans.SimpleBeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors, loadImage

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Inherited Member Summary

Methods inherited from interface java.beans.BeanInfo

getAdditionalBeanInfo, getBeanDescriptor, getDefaultEventIndex, getDefaultPropertyIndex, getEventSetDescriptors, getMethodDescriptors

Constructors

XMLTreeViewBeanInfo()

```
public  XMLTreeViewBeanInfo()
```

Methods

getIcon(int)

```
public java.awt.Image getIcon(int iconKind)
```

Overrides:

java.beans.SimpleBeanInfo.getIcon(int) in class java.beans.SimpleBeanInfo

getPropertyDescriptors()

```
public java.beans.PropertyDescriptor[] getPropertyDescriptors()
```

Overrides:

java.beans.SimpleBeanInfo.getPropertyDescriptors() in class java.beans.SimpleBeanInfo

Part I

XDK for PL/SQL Packages

This section contains Chapter 5, "XML Parser for PL/SQL"

5 XML Parser for PL/SQL

This chapter describes the Extensible Markup Language (XML) Parser for PL/SQL. It has three main sections:

- PL/SQL Parser APIs
- Extensible Stylesheet Language (XSL) Package Processor APIs
- W3C Document Object Model (DOM) APIs

PL/SQL Parser APIs

The Extensible Markup Language (XML) describes a class of data objects called XML documents. It partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of the Standard Generalized Markup Language (SGML) By construction, XML documents are conforming SGML documents.

XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure.

A software module called an XML processor is used to read XML documents and provide access to their content and structure. It is assumed that an XML processor is doing its work on behalf of another module, called the application. This PL/SQL implementation of the XML processor (or parser) followed the W3C XML specification (rev. REC-xml-19980210) and included the required behavior of an XML processor in terms of how it must read XML data and the information it must provide to the application.

The following is the default behavior for this PL/SQL XML parser:

- A parse tree which can be accessed by DOM APIs is built
- The parser is validating if a DTD is found, otherwise it is non-validating
- Errors are not recorded unless an error log is specified; however, an application error will be raised if parsing fails

The types and methods described in this document are made available by the PL/SQL package `xmlparser`.

Types and Functions

parse(VARCHAR2)

Parses xml stored in the given url/file and returns the built DOM Document

newParser

Returns a new parser instance

parse(Parser, VARCHAR2)

Parses xml stored in the given url/file

parseBuffer(Parser, VARCHAR2)

Parses xml stored in the given buffer

parseClob(Parser, CLOB)

Parses xml stored in the given clob

parseDTD(Parser, VARCHAR2, VARCHAR2)

Parses xml stored in the given url/file

parseDTDBuffer(Parser, VARCHAR2, VARCHAR2)

Parses xml stored in the given buffer

parseDTDClob(Parser, CLOB, VARCHAR2)

Parses xml stored in the given clob

setBaseDir(Parser, VARCHAR2)

Sets base directory used to resolve relative urls

showWarnings(Parser, BOOLEAN)

Turn warnings on or off

setErrorLog(Parser, VARCHAR2)

Sets errors to be sent to the specified file

setPreserveWhitespace(Parser, BOOLEAN)

Sets white space preserve mode

setValidationMode(Parser, BOOLEAN)

Sets validation mode

getValidationMode(Parser)

Gets validation mode

setDoctype(Parser, DOMDocumentType)

Sets DTD

getDoctype(Parser)

Gets DTD Parser

getDocument(Parser)

Gets DOM document

freeParser(Parser)

Free a Parser object

Parser Interface Type Description

TYPE Parser IS RECORD (ID VARCHAR2(5));

Function Prototypes

parse

PURPOSE

Parses xml stored in the given url/file and returns the built DOM Document.

SYNTAX

```
FUNCTION parse(url VARCHAR2) RETURN DOMDocument;
```

PARAMETERS

```
url      (IN)-  complete path of the url/file to be parsed
```

RETURNS

Nothing

COMMENTS

This is meant to be used when the default parser behavior is acceptable and just a url/file needs to be parsed.

An application error is raised if parsing failed, for some reason.

newParser

PURPOSE

Returns a new parser instance

SYNTAX

```
FUNCTION newParser RETURN Parser;
```

PARAMETERS

None

RETURNS

A new parser instance

COMMENTS

This function must be called before the default behavior of Parser can be changed and if other parse methods need to be used.

parse

PURPOSE

Parses xml stored in the given url/file

SYNTAX

```
PROCEDURE parse(p Parser, url VARCHAR2);
```

PARAMETERS

```
p        (IN)-  parser instance
url      (IN)-  complete path of the url/file to be parsed
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

parseBuffer

PURPOSE

Parses xml stored in the given buffer

SYNTAX

```
PROCEDURE parseBuffer(p Parser, doc VARCHAR2);
```

PARAMETERS

```
p       (IN)-  parser instance
doc     (IN)-  xml document buffer to parse
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

parseClob

PURPOSE

Parses xml stored in the given clob

SYNTAX

```
PROCEDURE parseClob(p Parser, doc CLOB);
```

PARAMETERS

```
p       (IN)-  parser instance
doc     (IN)-  xml document clob to parse
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

parseDTD

PURPOSE

Parses the DTD stored in the given url/file

SYNTAX

```
PROCEDURE parseDTD(p Parser, url VARCHAR2, root VARCHAR2);
```

PARAMETERS

```
p        (IN)-  parser instance
url      (IN)-  complete path of the url/file to be parsed
root     (IN)-  name of the root element
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

parseDTDBuffer

PURPOSE

Parses the DTD stored in the given buffer

SYNTAX

```
PROCEDURE parseDTDBuffer(p Parser, dtd VARCHAR2, root VARCHAR2);
```

PARAMETERS

```
p        (IN)-  parser instance
dtd      (IN)-  dtd buffer to parse
root     (IN)-  name of the root element
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

parseDTDClob

PURPOSE

Parses the DTD stored in the given clob

SYNTAX

```
PROCEDURE parseDTDClob(p Parser, dtd CLOB, root VARCHAR2);
```

PARAMETERS

```
p        (IN)-  parser instance
dtd      (IN)-  dtd clob to parse
root     (IN)-  name of the root element
```

RETURNS

Nothing

COMMENTS

Any changes to the default parser behavior should be effected before calling this procedure.

An application error is raised if parsing failed, for some reason.

setBaseDir

PURPOSE

Sets base directory used to resolve relative urls

SYNTAX

```
PROCEDURE setBaseDir(p Parser, dir VARCHAR2);
```

PARAMETERS

```
p        (IN)-  parser instance
dir      (IN)-  directory to use as base directory
```

RETURNS

Nothing

COMMENTS
An application error is raised if parsing failed, for some reason.

showWarnings

PURPOSE
Turn warnings on or off

SYNTAX
```
PROCEDURE showWarnings(p Parser, yes BOOLEAN);
```

PARAMETERS
```
p        (IN)-  parser instance
yes      (IN)-  mode to set: TRUE - show warnings, FALSE - don't show warnings
```

RETURNS
Nothing

setErrorLog

PURPOSE
Sets errors to be sent to the specified file

SYNTAX
```
PROCEDURE setErrorLog(p Parser, fileName VARCHAR2);
```

PARAMETERS
```
p        (IN)-  parser instance
fileName (IN)-  complete path of the file to use as the error log
```

RETURNS
Nothing

setPreserveWhitespace

PURPOSE
Sets whitespace preserving mode

SYNTAX

```
PROCEDURE setPreserveWhitespace(p Parser, yes BOOLEAN);
```

PARAMETERS

```
p       (IN)-  parser instance
yes     (IN)-  mode to set: TRUE - preserve, FALSE - don't preserve
```

RETURNS

Nothing

setValidationMode

PURPOSE

Sets validation mode

SYNTAX

```
PROCEDURE setValidationMode(p Parser, yes BOOLEAN);
```

PARAMETERS

```
p       (IN)-  parser instance
yes     (IN)-  mode to set: TRUE - validating, FALSE - non valid
```

RETURNS

Nothing

getValidationMode

PURPOSE

Gets validation mode

SYNTAX

```
FUNCTION getValidationMode(p Parser) RETURN BOOLEAN;
```

PARAMETERS

```
p       (IN)-  parser instance
```

RETURNS

The validation mode: TRUE - validating, FALSE - non valid

setDoctype

PURPOSE

Sets a DTD to be used by the parser for validation

SYNTAX

```
PROCEDURE setDoctype(p Parser, dtd DOMDocumentType);
```

PARAMETERS

```
p        (IN)-  parser instance
dtd      (IN)-  DTD to set
```

RETURNS

Nothing

getDoctype

PURPOSE

Gets DTD - MUST be called only after a DTD is parsed

SYNTAX

```
FUNCTION getDoctype(p Parser) RETURN DOMDocumentType;
```

PARAMETERS

```
p        (IN)-  parser instance
```

RETURNS

The parsed DTD

getDocument

PURPOSE

Gets DOM Document built by the parser - MUST be called only after a document is parsed

SYNTAX

```
FUNCTION getDocument(p Parser) RETURN DOMDocument;
```

PARAMETERS

```
p       (IN)-  parser instance
```

RETURNS

The root of the DOM tree

freeParser

PURPOSE

Free a parser object

SYNTAX

```
PROCEDURE freeParser(p Parser)
```

PARAMETERS

```
p       (IN)-  parser instance
```

Extensible Stylesheet Language (XSL) Package Processor APIs

The Extensible Stylesheet Language Transformation (XSLT), describes rules for transforming a source tree into a result tree. A transformation expressed in XSLT is called a stylesheet. The transformation specified is achieved by associating patterns with templates defined in the stylesheet. A template is instantiated to create part of the result tree. This PL/SQL implementation of the XSL processor followed the W3C XSLT working draft (rev WD-xslt-19990813) and included the required behavior of an XSL processor in terms of how it must read XSLT stylesheets and the transformation it must effect.

The following is the default behavior for this PL/SQL XML parser:

- A result tree which can be accessed by DOM APIs is built
- Errors are not recorded unless an error log is specified; however, an application error will be raised if parsing fails

The types and methods described in this document are made available by the PL/SQL package `xslprocessor`.

Functions

newProcessor

PURPOSE
Returns a new processor instance

SYNTAX
```
FUNCTION newProcessor RETURN Processor;
```

PARAMETERS
None

RETURNS
A new processor instance

COMMENTS
This function must be called before the default behavior of Processor can be changed and if other processor methods need to be used.

processXSL

PURPOSE
Transforms input XML document using given DOMDocument and stylesheet

SYNTAX
```
PROCEDURE processXSL(p Processor, ss Stylesheet, xmldoc DOMDocument);
```

PARAMETERS
```
p        (IN)-  processor instance
ss       (IN)-  stylesheet instance
xmldoc   (IN)-  xml document to be transformed
```

RETURNS
Nothing

COMMENTS

Any changes to the default processor behavior should be effected before calling this procedure.

An application error is raised if processing failed, for some reason.

processXSL

PURPOSE

Transforms input XML document using given DOMDocumentFragment and stylesheet

SYNTAX

```
PROCEDURE processXSL(p Processor, ss Stylesheet, xmldoc DOMDocumentFragment);
```

PARAMETERS

```
p        (IN)-  processor instance
ss       (IN)-  stylesheet instance
xmldoc   (IN)-  xml document fragment to be transformed
```

RETURNS

Nothing

COMMENTS

Any changes to the default processor behavior should be effected before calling this procedure.

An application error is raised if processing failed, for some reason.

showWarnings

PURPOSE

Turn warnings on or off

SYNTAX

```
PROCEDURE showWarnings(p Processor, yes BOOLEAN);
```

PARAMETERS

```
p        (IN)-  processor instance
yes      (IN)-  mode to set: TRUE - show warnings, FALSE - don't show warnings
```

RETURNS

Nothing

setErrorLog

PURPOSE

Sets errors to be sent to the specified file

SYNTAX

```
PROCEDURE setErrorLog(p Processor, fileName VARCHAR2);
```

PARAMETERS

```
p        (IN)-  processor instance
fileName (IN)-  complete path of the file to use as the error log
```

RETURNS

Nothing

newStylesheet

PURPOSE

Create a new stylesheet using the given input and reference URLs

SYNTAX

```
FUNCTION newStylesheet(inp VARCHAR2, ref VARCHAR2) RETURN Stylesheet;
```

PARAMETERS

```
inp      (IN)-  input url to use for construction
ref      (IN)-  reference url
```

RETURNS

```
A new stylesheet instance
```

transformNode

PURPOSE

Transforms a node in a DOM tree using the given stylesheet

SYNTAX

```
FUNCTION transformNode(n DOMNode, ss Stylesheet) RETURN DOMDocumentFragment;
```

PARAMETERS

```
n        (IN)-  DOM Node to transform
ss       (IN)-  stylesheet to use
```

RETURNS

Result of the transformation

selectNodes

PURPOSE

Selects nodes from a DOM tree which match the given pattern

SYNTAX

```
FUNCTION selectNodes(n DOMNode, pattern VARCHAR2) RETURN DOMNodeList;
```

PARAMETERS

```
n        (IN)-  DOM Node to transform
pattern  (IN)-  pattern to use
```

RETURNS

Result of the selection

selectSingleNodes

PURPOSE

Selects the first node from the tree that matches the given pattern

SYNTAX

```
FUNCTION selectSingleNodes(n DOMNode, pattern VARCHAR2) RETURN DOMNode;
```

PARAMETERS

```
n        (IN)-  DOM Node to transform
pattern  (IN)-  pattern to use
```

RETURNS

Result of the selection

valueOf

PURPOSE

Retrieves the value of the first node from the tree that matches the given pattern

SYNTAX

```
FUNCTION valueOf(n DOMNode, pattern VARCHAR2) RETURN VARCHAR2;
```

PARAMETERS

```
n        (IN)-  DOM Node to transform
pattern  (IN)-  pattern to use
```

RETURNS

Result of the selection

setParam

PURPOSE

Sets a top level paramter in the stylesheet

SYNTAX

```
PROCEDURE setParam(ss Stylesheet, name VARCHAR2, value VARCHAR2)
```

PARAMETERS

```
ss        (IN)-  stylesheet
name      (IN)-  name of the param
value      (IN)-  value of the param
```

removeParam

PURPOSE

Removes a top level stylesheet parameter

SYNTAX

```
PROCEDURE removeParam(ss Stylesheet, name VARCHAR2)
```

PARAMETERS

```
ss      (IN)-  Stylesheet
name  (IN)-  name of the parameter
```

resetParams

PURPOSE

Resets the top-level stylesheet parameters

SYNTAX

```
PROCEDURE resetParams(ss Stylesheet)
```

PARAMETERS

```
ss       (IN)-  Stylesheet
```

freeStylesheet

PURPOSE

Frees a Stylesheet object

SYNTAX

```
PROCEDURE freestylesheet(ss Stylesheet)
```

PARAMETERS

```
ss       (IN)-  Stylesheet
```

freeProcessor

PURPOSE

Frees a Processor object

SYNTAX

```
PROCEDURE freeProccessor(p Processor)
```

PARAMETERS

`p`	`(IN)-`	`Processor`

W3C Document Object Model (DOM) APIs

The Document Object Model (DOM) is an application programming interface (API) for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. In the DOM specification, the term "document" is used in the broad sense. XML is increasingly being used as a way of representing many different kinds of information that may be stored in diverse systems, and much of this would traditionally be seen as data rather than as documents. Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data.

With the Document Object Model, programmers can build documents, navigate their structure, and add, modify, or delete elements and content. Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the Document Object Model, with a few exceptions. In particular, the DOM interfaces for the XML internal and external subsets have not yet been specified.

One important objective of the W3C specification for the Document Object Model is to provide a standard programming interface that can be used in a wide variety of environments and applications. The DOM is designed to be used with any programming language. Since the DOM standard is object-oriented, for this PL/SQL adaptation, some changes had to be made:

- Various DOM interfaces such as Node, Element, etc. have equivalent PL/SQL types DOMNode, DOMElement, etc. respectively.
- Various DOMException codes such as WRONG_DOCUMENT_ERR, HIERARCHY_REQUEST_ERR, etc. have similarly named PL/SQL exceptions
- Various DOM Node type codes such as ELEMENT_NODE, ATTRIBUTE_NODE, etc. have similarly named PL/SQL constants
- Methods defined on a DOM type become functions or procedures that accept it as a parameter. For example, to perform appendChild on a DOM Node n, the following PL/SQL function is provided:

FUNCTION

```
appendChild(n DOMNode, newChild IN DOMNode)
```

RETURN

```
DOMNode;
```

and to perform setAttribute on a DOM Element elem, the following PL/SQL procedure is provided:

PROCEDURE

```
setAttribute(elem DOMElement, name IN VARCHAR2,
                              value IN VARCHAR2);
```

DOM defines an inheritance hierarchy. For example, Document, Element, and Attr are defined to be subtypes of Node. Thus, a method defined in the Node interface should be available in these as well. Since, such inheritance is not directly possible in PL/SQL, the makeNode functions need to be invoked on different DOM types to convert these into a DOMNode. The appropriate functions or procedures that accept DOMNodes can then be called to operate on these types. If, subsequently, type specific functionality is desired, the DOMNode can be converted back into the type by using the makeXXX functions, where DOMXXX is the DOM type desired

The implementation of this PL/SQL DOM interface followed the DOM standard of revision REC-DOM-Level-1-19981001. The types and methods described in this document are made available by the PL/SQL package `xmldom`.

Types

DOM Node Types

ELEMENT_NODE

ATTRIBUTE_NODE

TEXT_NODE

CDATA_SECTION_NODE

ENTITY_REFERENCE_NODE

ENTITY_NODE

PROCESSING_INSTRUCTION_NODE

COMMENT_NODE

DOCUMENT_NODE

DOCUMENT_TYPE_NODE

DOCUMENT_FRAGMENT_NODE

NOTATION_NODE

DOM Exception Types

INDEX_SIZE_ERR

DOMSTRING_SIZE_ERR

HIERARCHY_REQUEST_ERR

WRONG_DOCUMENT_ERR

INVALID_CHARACTER_ERR

NO_DATA_ALLOWED_ERR

NO_MODIFICATION_ALLOWED_ERR

NOT_FOUND_ERR

NOT_SUPPORTED_ERR

INUSE_ATTRIBUTE_ERR

DOM Interface Types

DOMNode

DOMNamedNodeMap

DOMNodeList

DOMAttr

DOMCDataSection

DOMCharacterData

DOMComment

DOMDocumentFragment

DOMElement

DOMEntity

DOMEntityReference

DOMNotation

DOMProcessingInstruction

DOMText

DOMImplementation

DOMDocumentType

DOMDocument

Methods

Node Methods

FUNCTION
isNull(n DOMNode)

RETURN
BOOLEAN;

FUNCTION
makeAttr(n DOMNode)

RETURN
DOMAttr;

FUNCTION
makeCDataSection(n DOMNode)

RETURN
DOMCDataSection;

FUNCTION
makeCharacterData(n DOMNode)

RETURN
DOMCharacterData;

FUNCTION
makeComment(n DOMNode)

RETURN
DOMComment;

FUNCTION

makeDocumentFragment(n DOMNode)

RETURN

DOMDocumentFragment;

FUNCTION

makeDocumentType(n DOMNode)

RETURN

DOMDocumentType;

FUNCTION

makeElement(n DOMNode)

RETURN

DOMElement;

FUNCTION

makeEntity(n DOMNode)

RETURN

DOMEntity;

FUNCTION

makeEntityReference(n DOMNode)

RETURN

DOMEntityReference;

FUNCTION

makeNotation(n DOMNode)

RETURN

DOMNotation;

FUNCTION
makeProcessingInstruction(n DOMNode)

RETURN
DOMProcessingInstruction;

makeText(n DOMNode) RETURN DOMText;

FUNCTION
makeDocument(n DOMNode)

RETURN
DOMDocument;

PROCEDURE
writeToFile(n DOMNode, fileName VARCHAR2);

PROCEDURE
writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2);

PROCEDURE
writeToClob(n DOMNode, cl IN OUT CLOB);

PROCEDURE
writeToFile(n DOMNode, fileName VARCHAR2, charset VARCHAR2);

PROCEDURE
writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2, charset VARCHAR2);

PROCEDURE
writeToClob(n DOMNode, cl IN OUT CLOB, charset VARCHAR2);

FUNCTION
getNodeName(n DOMNode)

RETURN
VARCHAR2;

FUNCTION

getNodeValue(n DOMNode)

RETURN

VARCHAR2;

PROCEDURE

setNodeValue(n DOMNode, nodeValue IN VARCHAR2);

FUNCTION

getNodeType(n DOMNode)

RETURN

NUMBER;

FUNCTION

getParentNode(n DOMNode)

RETURN

DOMNode;

FUNCTION

getChildNodes(n DOMNode)

RETURN

DOMNodeList;

FUNCTION

getFirstChild(n DOMNode)

RETURN

DOMNode;

FUNCTION

getLastChild(n DOMNode)

RETURN

DOMNode;

FUNCTION

getPreviousSibling(n DOMNode)

RETURN

DOMNode;

FUNCTION

getNextSibling(n DOMNode)

RETURN

DOMNode;

FUNCTION

getAttributes(n DOMNode)

RETURN

DOMNamedNodeMap;

FUNCTION

getOwnerDocument(n DOMNode)

RETURN

DOMDocument;

FUNCTION

insertBefore(n DOMNode, newChild IN DOMNode, refChild IN DOMNode)

RETURN

DOMNode;

FUNCTION

replaceChild(n DOMNode, newChild IN DOMNode, oldChild IN DOMNode)>

RETURN

DOMNode;

FUNCTION

removeChild(n DOMNode, oldChild IN DOMNode)

RETURN

DOMNode;

FUNCTION

appendChild(n DOMNode, newChild IN DOMNode)

RETURN

DOMNode;

FUNCTION

hasChildNodes(n DOMNode)

RETURN

BOOLEAN;

FUNCTION

cloneNode(n DOMNode, deep boolean)

RETURN

DOMNode;

Named Node Map Methods

FUNCTION
isNull(nnm DOMNamedNodeMap)

RETURN
BOOLEAN;

FUNCTION
getNamedItem(nnm DOMNamedNodeMap, name IN VARCHAR2)

RETURN
DOMNode;

FUNCTION
setNamedItem(nnm DOMNamedNodeMap, arg IN DOMNode)

RETURN
DOMNode;

FUNCTION
removeNamedItem(nnm DOMNamedNodeMap, name IN VARCHAR2)

RETURN
DOMNode;

FUNCTION
item(nnm DOMNamedNodeMap, idx IN NUMBER)

RETURN
DOMNode;

FUNCTION
getLength(nnm DOMNamedNodeMap)

RETURN

NUMBER;

Node List Methods

FUNCTION

isNull(nl DOMNodeList)

RETURN

BOOLEAN;

FUNCTION

item(nl DOMNodeList, idx IN NUMBER)

RETURN

DOMNode;

FUNCTION

getLength(nl DOMNodeList)

RETURN

NUMBER;

Attr Methods

FUNCTION
isNull(a DOMAttr)

RETURN
BOOLEAN;

FUNCTION
makeNode(a DOMAttr)

RETURN
DOMNode;

FUNCTION
getQualifiedName(a DOMAttr)

RETURN
VARCHAR2;

FUNCTION
getNamespace(a DOMAttr)

RETURN
VARCHAR2;

FUNCTION
getLocalName(a DOMAttr)

RETURN
VARCHAR2;

FUNCTION
getExpandedName(a DOMAttr)

RETURN
VARCHAR2;

FUNCTION
getName(a DOMAttr)

RETURN
VARCHAR2;

FUNCTION
getSpecified(a DOMAttr)

RETURN
BOOLEAN;

FUNCTION
getValue(a DOMAttr)

RETURN
VARCHAR2;

PROCEDURE
setValue(a DOMAttr, value IN VARCHAR2);

C Data Section Methods

FUNCTION

isNull(cds DOMCDataSection)

RETURN

BOOLEAN;

FUNCTION

makeNode(cds DOMCDataSection)

RETURN

DOMNode;

Character Data Methods

FUNCTION

isNull(cd DOMCharacterData)

RETURN

BOOLEAN;

FUNCTION

makeNode(cd DOMCharacterData)

RETURN

DOMNode;

FUNCTION

getData(cd DOMCharacterData)

RETURN

VARCHAR2;

PROCEDURE

setData(cd DOMCharacterData, data IN VARCHAR2);

FUNCTION

getLength(cd DOMCharacterData)

RETURN

NUMBER;

FUNCTION

substringData(cd DOMCharacterData, offset IN NUMBER, cnt IN NUMBER)

RETURN

VARCHAR2;

PROCEDURE

appendData(cd DOMCharacterData, arg IN VARCHAR2);

PROCEDURE

insertData(cd DOMCharacterData, offset IN NUMBER, arg IN VARCHAR2);

PROCEDURE

deleteData(cd DOMCharacterData, offset IN NUMBER, cnt IN NUMBER);

PROCEDURE

replaceData(cd DOMCharacterData, offset IN NUMBER, cnt IN NUMBER, arg IN VARCHAR2);

Comment Methods

FUNCTION

isNull(com DOMComment)

RETURN

BOOLEAN;

FUNCTION

makeNode(com DOMComment)

RETURN

DOMNode;

DOM Implementation Methods

FUNCTION

isNull(di DOMImplementation)

RETURN

BOOLEAN;

FUNCTION

hasFeature(di DOMImplementation, feature IN VARCHAR2, version IN VARCHAR2)

RETURN

BOOLEAN;

Document Fragment Methods

FUNCTION

isNull(df DOMDocumentFragment)

RETURN

BOOLEAN;

FUNCTION

makeNode(df DOMDocumentFragment)

RETURN

DOMNode;

Document Type Methods

FUNCTION

isNull(dt DOMDocumentType)

RETURN

BOOLEAN;

FUNCTION

makeNode(dt DOMDocumentType)

RETURN

DOMNode;

FUNCTION

findEntity(dt DOMDocumentType, name VARCHAR2, par BOOLEAN)

RETURN

DOMEntity;

FUNCTION

findNotation(dt DOMDocumentType, name VARCHAR2)

RETURN

DOMNotation;

FUNCTION

getPublicId(dt DOMDocumentType)

RETURN

VARCHAR2;

FUNCTION

getSystemId(dt DOMDocumentType)

RETURN

VARCHAR2;

PROCEDURE

writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2);

PROCEDURE

writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2);

PROCEDURE

writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB);

PROCEDURE

writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2, charset VARCHAR2);

PROCEDURE

writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2, charset VARCHAR2);

PROCEDURE

writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB, charset VARCHAR2);

FUNCTION

getName(dt DOMDocumentType)

RETURN

VARCHAR2;

FUNCTION

getEntities(dt DOMDocumentType)

RETURN

DOMNamedNodeMap;

FUNCTION

getNotations(dt DOMDocumentType)

RETURN

DOMNamedNodeMap;

Element Methods

FUNCTION

isNull(elem DOMElement)

RETURN

BOOLEAN;

FUNCTION

makeNode(elem DOMElement)

RETURN

DOMNode;

FUNCTION

getQualifiedName(elem DOMElement)

RETURN

VARCHAR2;

FUNCTION

getNamespace(elem DOMElement)

RETURN

VARCHAR2;

FUNCTION

getLocalName(elem DOMElement)

RETURN

VARCHAR2;

FUNCTION

getExpandedName(elem DOMElement)

RETURN

VARCHAR2;

FUNCTION

getChildrenByTagName(elem DOMElement, name IN VARCHAR2)

RETURN

DOMNodeList;

FUNCTION

getElementsByTagName(elem DOMElement, name IN VARCHAR2, ns VARCHAR2)

RETURN

DOMNodeList;

FUNCTION

getChildrenByTagName(elem DOMElement, name IN VARCHAR2, ns VARCHAR2)

RETURN

DOMNodeList;

FUNCTION

resolveNamespacePrefix(elem DOMElement, prefix VARCHAR2)

RETURN

VARCHAR2;

FUNCTION

getTagName(elem DOMElement)

RETURN

VARCHAR2;

FUNCTION

getAttribute(elem DOMElement, name IN VARCHAR2)

RETURN

VARCHAR2;

PROCEDURE

setAttribute(elem DOMElement, name IN VARCHAR2, value IN VARCHAR2);

PROCEDURE

removeAttribute(elem DOMElement, name IN VARCHAR2);

FUNCTION

getAttributeNode(elem DOMElement, name IN VARCHAR2)

RETURN

DOMAttr;

FUNCTION

setAttributeNode(elem DOMElement, newAttr IN DOMAttr)

RETURN

DOMAttr;

FUNCTION

removeAttributeNode(elem DOMElement, oldAttr IN DOMAttr)

RETURN

DOMAttr;

FUNCTION

getElementsByTagName(elem DOMElement, name IN VARCHAR2)

RETURN

DOMNodeList;

PROCEDURE

normalize(elem DOMElement);

Entity Methods

FUNCTION
isNull(ent DOMEntity)

RETURN
BOOLEAN;

FUNCTION
makeNode(ent DOMEntity)

RETURN
DOMNode;

FUNCTION
getPublicId(ent DOMEntity)

RETURN
VARCHAR2;

FUNCTION
getSystemId(ent DOMEntity)

RETURN
VARCHAR2;

FUNCTION
getNotationName(ent DOMEntity)

RETURN
VARCHAR2;

Entity Reference Methods

FUNCTION

isNull(eref DOMEntityReference)

RETURN

BOOLEAN;

FUNCTION

makeNode(eref DOMEntityReference)

RETURN

DOMNode;

Notation Methods

FUNCTION
isNull(n DOMNotation)

RETURN
BOOLEAN;

FUNCTION
makeNode(n DOMNotation)

RETURN
DOMNode;

FUNCTION
getPublicId(n DOMNotation)

RETURN
VARCHAR2;

FUNCTION
getSystemId(n DOMNotation)

RETURN
VARCHAR2;

Processing Instruction Methods

FUNCTION
isNull(pi DOMProcessingInstruction)

RETURN
BOOLEAN;

FUNCTION
makeNode(pi DOMProcessingInstruction)

RETURN
DOMNode;

FUNCTION
getData(pi DOMProcessingInstruction)

RETURN
VARCHAR2;

FUNCTION
getTarget(pi DOMProcessingInstruction)

RETURN
VARCHAR2;

PROCEDURE
setData(pi DOMProcessingInstruction, data IN VARCHAR2);

Text Methods

FUNCTION
isNull(t DOMText)

RETURN

BOOLEAN;

FUNCTION

makeNode(t DOMText)

RETURN

DOMNode;

FUNCTION

splitText(t DOMText, offset IN NUMBER)

RETURN

DOMText;

Document Methods

FUNCTION
isNull(doc DOMDocument)

RETURN
BOOLEAN;

FUNCTION
makeNode(doc DOMDocument)

RETURN
DOMNode;

FUNCTION
newDOMDocument

RETURN
DOMDocument;

FUNCTION
getVersion(doc DOMDocument)

RETURN
VARCHAR2;

PROCEDURE
setVersion(doc DOMDocument, version VARCHAR2);

FUNCTION
getCharset(doc DOMDocument)

RETURN
VARCHAR2;

PROCEDURE

setCharset(doc DOMDocument, charset VARCHAR2);

FUNCTION

getStandalone(doc DOMDocument)

RETURN

VARCHAR2;

PROCEDURE

setStandalone(doc DOMDocument, value VARCHAR2);

PROCEDURE

writeToFile(doc DOMDocument, fileName VARCHAR2);

PROCEDURE

writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2);

PROCEDURE

writeToClob(doc DOMDocument, cl IN OUT CLOB);

PROCEDURE

writeToFile(doc DOMDocument, fileName VARCHAR2, charset VARCHAR2);

PROCEDURE

writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2, charset VARCHAR2);

PROCEDURE

writeToClob(doc DOMDocument, cl IN OUT CLOB, charset VARCHAR2);

PROCEDURE

writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2);

PROCEDURE

writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2);

PROCEDURE

writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB);

PROCEDURE

writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2, charset VARCHAR2);

PROCEDURE

writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2, charset VARCHAR2);

PROCEDURE

writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB, charset VARCHAR2);

FUNCTION

getDoctype(doc DOMDocument)

RETURN

DOMDocumentType;

FUNCTION

getImplementation(doc DOMDocument)

RETURN

DOMImplementation;

FUNCTION

getDocumentElement(doc DOMDocument)

RETURN

DOMElement;

FUNCTION

createElement(doc DOMDocument, tagName IN VARCHAR2)

RETURN

DOMElement;

FUNCTION

createDocumentFragment(doc DOMDocument)

RETURN

DOMDocumentFragment;

FUNCTION

createTextNode(doc DOMDocument, data IN VARCHAR2)

RETURN

DOMText;

FUNCTION

createComment(doc DOMDocument, data IN VARCHAR2)

RETURN

DOMComment;

FUNCTION

createCDATASection(doc DOMDocument, data IN VARCHAR2)

RETURN

DOMCDATASection;

FUNCTION

createProcessingInstruction(doc DOMDocument, target IN VARCHAR2, data IN VARCHAR2)

RETURN

DOMProcessingInstruction;

FUNCTION

createAttribute(doc DOMDocument, name IN VARCHAR2)

RETURN

DOMAttr;

FUNCTION

createEntityReference(doc DOMDocument, name IN VARCHAR2)

RETURN

DOMEntityReference;

FUNCTION

getElementsByTagName(doc DOMDocument, tagname IN VARCHAR2)

RETURN

DOMNodeList;

Method Prototypes

Node Methods

FUNCTION

isNull(n DOMNode)

RETURN

BOOLEAN;

PURPOSE

Checks if the given DOMNode is null

SYNTAX

```
 FUNCTION isNull(n DOMNode) RETURN BOOLEAN;
```

PARAMETERS

```
 n      (IN)-  DOMNode to check
```

RETURNS

Whether given DOMNode is null: TRUE - is null, FALSE - is not null

FUNCTION

makeAttr(n DOMNode) RETURN DOMAttr;

PURPOSE

Casts given DOMNode to a DOMAttr

SYNTAX

```
 FUNCTION makeAttr(n DOMNode) RETURN DOMAttr;
```

PARAMETERS

```
 n      (IN)-  DOMNode to cast
```

RETURNS

The DOMAttr

FUNCTION

makeCDataSection(n DOMNode) RETURN DOMCDataSection;

PURPOSE

Casts given DOMNode to a DOMCDataSection

SYNTAX

```
FUNCTION makeCDataSection(n DOMNode) RETURN DOMCDataSection;
```

PARAMETERS

```
n      (IN)-  DOMNode to cast
```

RETURNS

The DOMCDataSection

FUNCTION

makeCharacterData(n DOMNode) RETURN DOMCharacterData;

PURPOSE

Casts given DOMNode to a DOMCharacterData

SYNTAX

```
FUNCTION makeCharacterData(n DOMNode) RETURN DOMCharacterData;
```

PARAMETERS

```
n      (IN)-  DOMNode to cast
```

RETURNS

The DOMCharacterData

FUNCTION

makeComment(n DOMNode) RETURN DOMComment;

PURPOSE

Casts given DOMNode to a DOMComment

SYNTAX

```
FUNCTION makeComment(n DOMNode) RETURN DOMComment;
```

PARAMETERS

```
n      (IN)-  DOMNode to cast
```

RETURNS

The DOMComment

FUNCTION

makeDocumentFragment(n DOMNode) RETURN DOMDocumentFragment;

PURPOSE

Casts given DOMNode to a DOMDocumentFragment

SYNTAX

```
FUNCTION makeDocumentFragment(n DOMNode) RETURN DOMDocumentFragment;
```

PARAMETERS

```
n      (IN)-  DOMNode to cast
```

RETURNS

The DOMDocumentFragment

```
FUNCTION makeDocumentType(n DOMNode) RETURN DOMDocumentType;
```

PURPOSE

Casts given DOMNode to a DOMDocumentType

SYNTAX

```
FUNCTION makeDocumentType(n DOMNode) RETURN DOMDocumentType;
```

PARAMETERS

```
n      (IN)-  DOMNode to cast
```

RETURNS

The DOMDocumentType

FUNCTION **makeElement(n DOMNode)**

RETURNS
DOMElement;

PURPOSE
Casts given DOMNode to a DOMElement

SYNTAX
```
 FUNCTION makeElement(n DOMNode) RETURN DOMElement;
```

PARAMETERS
```
 n      (IN)-  DOMNode to cast
```

RETURNS
The DOMElement

FUNCTION **makeEntity(n DOMNode)** RETURN DOMEntity;

PURPOSE
Casts given DOMNode to a DOMEntity

SYNTAX
```
 FUNCTION makeEntity(n DOMNode) RETURN DOMEntity;
```

PARAMETERS
```
 n      (IN)-  DOMNode to cast
```

RETURNS
The DOMEntity

FUNCTION **makeEntityReference(n DOMNode)** RETURN DOMEntityReference;

PURPOSE
Casts given DOMNode to a DOMEntityReference

SYNTAX
```
 FUNCTION makeEntityReference(n DOMNode) RETURN DOMEntityReference;
```

PARAMETERS

```
 n      (IN)-  DOMNode to cast
```

RETURNS

The DOMEntityReference

FUNCTION

`makeNotation(n DOMNode)`

RETURN

`DOMNotation;`

PURPOSE

Casts given DOMNode to a DOMNotation

SYNTAX

FUNCTION

`makeNotation(n DOMNode)`

RETURN

`DOMNotation;`

PARAMETERS

```
 n      (IN)-  DOMNode to cast
```

RETURNS

The DOMNotation

FUNCTION

makeProcessingInstruction(n DOMNode)

RETURN

DOMProcessingInstruction;

PURPOSE

Casts given DOMNode to a DOMProcessingInstruction

SYNTAX

FUNCTION
`makeProcessingInstruction(n DOMNode)`

RETURN
`DOMProcessingInstruction;`

PARAMETERS
```
 n      (IN)-  DOMNode to cast
```

RETURNS
The DOMProcessingInstruction

FUNCTION
makeText(n DOMNode)

RETURN
DOMText;

PURPOSE
Casts given DOMNode to a DOMText

SYNTAX

FUNCTION
`makeText(n DOMNode)`

RETURN
`DOMText;`

PARAMETERS
```
 n      (IN)-  DOMNode to cast
```

RETURNS
The DOMText

FUNCTION

makeDocument(n DOMNode)

RETURN

DOMDocument;

PURPOSE

Casts given DOMNode to a DOMDocument

SYNTAX

FUNCTION

```
makeDocument(n DOMNode)
```

RETURN

```
DOMDocument;
```

PARAMETERS

```
 n      (IN)-  DOMNode to cast
```

RETURNS

The DOMDocument

PROCEDURE

writeToFile(n DOMNode, fileName VARCHAR2);

PURPOSE

Writes XML node to specified file using the database character set

SYNTAX

PROCEDURE

```
writeToFile(n DOMNode, fileName VARCHAR2);
```

PARAMETERS

```
 n        (IN)-  DOMNode
 fileName (IN)-  File to write to
```

RETURNS
Nothing

PROCEDURE

writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2);

PURPOSE
Writes XML node to specified buffer using the database character set

SYNTAX

PROCEDURE
```
writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2);
```

PARAMETERS
```
 n        (IN)-  DOMNode
 buffer   (OUT)- buffer to write to
```

RETURNS
Nothing

PROCEDURE

writeToClob(n DOMNode, cl IN OUT CLOB);

PURPOSE
Writes XML node to specified clob using the database character set

SYNTAX

PROCEDURE
```
writeToClob(n DOMNode, cl IN OUT CLOB);
```

PARAMETERS
```
 n        (IN)-  DOMNode
 cl      (OUT)-  CLOB to write to
```

RETURNS

Nothing

PROCEDURE

writeToFile(n DOMNode, fileName VARCHAR2, charset VARCHAR2);

PURPOSE

Writes XML node to specified file using the given character set

SYNTAX

PROCEDURE

```
writeToFile(n DOMNode, fileName VARCHAR2, charset VARCHAR2);
```

PARAMETERS

```
 n        (IN)-  DOMNode
 fileName (IN)-  File to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE

writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2, charset VARCHAR2);

PURPOSE

Writes XML node to specified buffer using the given character set

SYNTAX

PROCEDURE

```
writeToBuffer(n DOMNode, buffer IN OUT VARCHAR2, charset VARCHAR2);
```

PARAMETERS

```
 n        (IN)-  DOMNode
 buffer   (OUT)- buffer to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE

writeToClob(n DOMNode, cl IN OUT CLOB, charset VARCHAR2);

PURPOSE

Writes XML node to specified clob using the given character set

SYNTAX

PROCEDURE

```
writeToClob(n DOMNode, cl IN OUT CLOB, charset VARCHAR2);
```

PARAMETERS

```
 n        (IN)-  DOMNode
 cl      (OUT)-  CLOB to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

Named Node Map Methods

FUNCTION

isNull(nnm DOMNamedNodeMap)

RETURN

BOOLEAN;

PURPOSE

Checks if the given DOM NamedNodeMap is null

SYNTAX

FUNCTION
`isNull(nnm DOMNamedNodeMap)`

RETURN
`BOOLEAN;`

PARAMETERS
```
 nnm      (IN)-  DOMNamedNodeMap to check
```

RETURNS
Whether given DOM NamedNodeMap is null: TRUE - is null, FALSE - is not null

Node List Methods

FUNCTION
isNull(nl DOMNodeList)

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM NodeList is null

SYNTAX
```
 FUNCTION isNull(nl DOMNodeList) RETURN BOOLEAN;
```

PARAMETERS
```
 nl      (IN)-  DOMNodeList to check
```

RETURNS
Whether given DOM NodeList is null: TRUE - is null, FALSE - is not null

Attr Methods

FUNCTION **isNull(a DOMAttr)** RETURN BOOLEAN;

PURPOSE
Checks if the given DOM Attr is null

SYNTAX
```
FUNCTION isNull(a DOMAttr) RETURN BOOLEAN;
```

PARAMETERS
```
a      (IN)-  DOMAttr to check
```

RETURNS
Whether given DOM Attr is null: TRUE - is null, FALSE - is not null

FUNCTION **makeNode(a DOMAttr)** RETURN DOMNode;

PURPOSE
Casts given DOMAttr to a DOMNode

SYNTAX
```
FUNCTION makeNode(a DOMAttr) RETURN DOMNode;
```

PARAMETERS
```
a      (IN)-  DOMNode to cast
```

RETURNS
The DOMNode

FUNCTION **getQualifiedName(a DOMAttr)** RETURN VARCHAR2;

PURPOSE
Returns the qualified name of the DOMAttr

SYNTAX
```
FUNCTION getQualifiedName(a DOMAttr) RETURN VARCHAR2;
```

PARAMETERS

```
 a      (IN)-  DOMAttr
```

RETURNS

The qualified name

FUNCTION **getNamespace(a DOMAttr)** RETURN VARCHAR2;

PURPOSE

Returns the namespace of the DOMAttr

SYNTAX

```
FUNCTION getNamespace(a DOMAttr) RETURN VARCHAR2;
```

PARAMETERS

```
 a      (IN)-  DOMAttr
```

RETURNS

The namespace

FUNCTION **getLocalName(a DOMAttr)** RETURN VARCHAR2;

PURPOSE

Returns the local name of the DOMAttr

SYNTAX

```
FUNCTION getLocalName(a DOMAttr) RETURN VARCHAR2;
```

PARAMETERS

```
 a      (IN)-  DOMAttr
```

RETURNS

The local name

FUNCTION **getExpandedName(a DOMAttr)** RETURN VARCHAR2;

PURPOSE

Returns the expanded name of the DOMAttr

SYNTAX

```
FUNCTION getExpandedName(a DOMAttr) RETURN VARCHAR2;
```

PARAMETERS

```
 a      (IN)-  DOMAttr
```

RETURNS

The expanded name

C Data Section Methods

FUNCTION

isNull(cds DOMCDataSection)

RETURN

BOOLEAN;

PURPOSE

Checks if the given DOM CDataSection is null

SYNTAX

```
FUNCTION isNull(cds DOMCDataSection) RETURN BOOLEAN;
```

PARAMETERS

```
cds      (IN)-  DOMCDataSection to check
```

RETURNS

Whether given DOM CDataSection is null: TRUE - is null, FALSE - is not null

FUNCTION

makeNode(cds DOMCDataSection)

RETURN

DOMNode;

PURPOSE
Casts given DOMCDataSection to a DOMNode

SYNTAX
```
 FUNCTION makeNode(cds DOMCDataSection) RETURN DOMNode;
```

PARAMETERS
```
 cds      (IN)-  DOMCDataSection to cast
```

RETURNS
The DOMNode

Character Data Methods

FUNCTION
`isNull(cd DOMCharacterData)`

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM CharacterData is null

SYNTAX
```
FUNCTION isNull(cd DOMCharacterData) RETURN BOOLEAN;
```

PARAMETERS
```
 cd      (IN)-  DOMCharacterData to check
```

RETURNS
Whether given DOM CharacterData is null : TRUE - is null, FALSE - is not null

FUNCTION
makeNode(cd DOMCharacterData)

RETURN
DOMNode;

PURPOSE
Casts given DOMCharacterData to a DOMNode

SYNTAX
```
FUNCTION makeNode(cd DOMCharacterData) RETURN DOMNode;
```

PARAMETERS
```
 cd      (IN)-  DOMCharacterData to cast
```

RETURNS
The DOMNode

Comment Methods

FUNCTION
`isNull(com DOMComment)`

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM Comment is null

SYNTAX
```
 FUNCTION isNull(com DOMComment) RETURN BOOLEAN;
```

PARAMETERS
```
 com      (IN)-  DOMComment to check
```

RETURNS
Whether given DOM Comment is null: TRUE - is null, FALSE - is not null

FUNCTION
`makeNode(com DOMComment)`

RETURN
DOMNode;

PURPOSE
Casts given DOMComment to a DOMNode

SYNTAX
```
FUNCTION makeNode(com DOMComment) RETURN DOMNode;
```

PARAMETERS
```
 com     (IN)-  DOMComment to ast
```

RETURNS
The DOMComment

DOM Implementation Methods

FUNCTION
`isNull(di DOMImplementation)`

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM Implementation is null

SYNTAX
```
FUNCTION isNull(di DOMImplementation) RETURN BOOLEAN;
```

PARAMETERS
```
 di      (IN)-  DOMImplementation to check
```

RETURNS
Whether given DOM Implementation is null: TRUE - is null, FALSE - is not null

Document Fragment Methods

FUNCTION
`isNull(df DOMDocumentFragment)`

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM DocumentFragment is null

SYNTAX
```
FUNCTION isNull(df DOMDocumentFragment) RETURN BOOLEAN;
```

PARAMETERS
```
 df      (IN)-  DOMDocumentFragment to check
```

RETURNS
Whether given DOM DocumentFragment is null: TRUE - is null, FALSE - is not null

FUNCTION
`makeNode(df DOMDocumentFragment)`

RETURN
DOMComment;

PURPOSE
Casts given DOMDocumentFragment to a DOMNode

SYNTAX
```
FUNCTION makeNode(df DOMDocumentFragment) RETURN DOMNode;
```

PARAMETERS

```
df      (IN)-  DOMDocumentFragment to cast
```

RETURNS

The DOMNode

Document Type Methods

FUNCTION **isNull(dt DOMDocumentType)** RETURN BOOLEAN;

PURPOSE

Checks if the given DOM DocumentType is null

SYNTAX

```
FUNCTION isNull(dt DOMDocumentType) RETURN BOOLEAN;
```

PARAMETERS

```
dt      (IN)-  DOMDocumentType to check
```

RETURNS

Whether given DOM DocumentType is null: TRUE **-** is null, FALSE **-** is not null

---

FUNCTION **makeNode(dt DOMDocumentType)** RETURN DOMNode;

PURPOSE

Casts given DOMDocumentType to a DOMNode

SYNTAX

```
FUNCTION makeNode(dt DOMDocumentType) RETURN DOMNode;
```

PARAMETERS

```
dt      (IN)-  DOMDocumentType to cast
```

RETURNS

The DOMNode

FUNCTION

findEntity(dt DOMDocumentType, name VARCHAR2, par BOOLEAN)

RETURN

DOMEntity;

PURPOSE

Finds an entity in the given DTD

SYNTAX

```
FUNCTION findEntity(dt DOMDocumentType, name VARCHAR2, par BOOLEAN) RETURN
DOMEntity;
```

PARAMETERS

```
 dt      (IN)-  DTD
 name    (IN)-  entity to find
 par     (IN)-  TRUE - parameter entity, FALSE - normal entity
```

RETURNS

The DOMEntity, if found.

FUNCTION

findNotation(dt DOMDocumentType, name VARCHAR2)

RETURN

DOMNotation;

PURPOSE

Finds a notation in the given DTD

SYNTAX

```
 FUNCTION findNotation(dt DOMDocumentType, name VARCHAR2) RETURN DOMNotation;
```

PARAMETERS

```
 dt      (IN)-  DTD
 name    (IN)-  notation to find
```

RETURNS
The DOMNotation, if found.

FUNCTION
getPublicId(dt DOMDocumentType)

RETURN
VARCHAR2;

PURPOSE
Return the public id of the given DTD

SYNTAX
```
FUNCTION getPublicId(dt DOMDocumentType) RETURN VARCHAR2;
```

PARAMETERS
```
 dt      (IN)-  DTD
```

RETURNS
The public id

FUNCTION
getSystemId(dt DOMDocumentType)

RETURN
VARCHAR2;

PURPOSE
Return the system id of the given DTD

SYNTAX
```
FUNCTION getSystemId(dt DOMDocumentType) RETURN VARCHAR2;
```

PARAMETERS
```
 dt      (IN)-  DTD
```

RETURNS

The system id

PROCEDURE

`writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2);`

PURPOSE

Writes DTD to specified file using the database character set

SYNTAX

```
PROCEDURE writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2);
```

PARAMETERS

```
 dt       (IN)-  DOMDocumentType
 fileName (IN)-  File to write to
```

RETURNS

Nothing

PROCEDURE

writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2);

PURPOSE

Writes DTD to specified buffer using the database character set

SYNTAX

PROCEDURE

```
writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2);
```

PARAMETERS

```
 dt       (IN)-  DOMDocumentType
 buffer   (OUT)- buffer to write to
```

RETURNS

Nothing

PROCEDURE

`writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB);`

PURPOSE

Writes DTD to specified clob using the database character set

SYNTAX

```
 PROCEDURE writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB);
```

PARAMETERS

```
 dt       (IN)-  DOMDocumentType
 cl      (OUT)-  CLOB to write to
```

RETURNS

Nothing

PROCEDURE

writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2, charset VARCHAR2);

PURPOSE

Writes DTD to specified file using the given character set

SYNTAX

PROCEDURE

`writeExternalDTDToFile(dt DOMDocumentType, fileName VARCHAR2, charset VARCHAR2);`

PARAMETERS

```
 dt       (IN)-  DOMDocumentType
 fileName (IN)-  File to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE

writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2, charset VARCHAR2);

PURPOSE
Writes DTD to specified buffer using the given character set

SYNTAX

PROCEDURE
```
writeExternalDTDToBuffer(dt DOMDocumentType, buffer IN OUT VARCHAR2, charset
VARCHAR2);
```

PARAMETERS
```
 dt       (IN)-  DOMDocumentType
 buffer   (OUT)- buffer to write to
 charset  (IN)-  Character set
```

RETURNS
Nothing

PROCEDURE

writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB, charset VARCHAR2);

PURPOSE
Writes DTD to specified clob using the given character set

SYNTAX
```
 PROCEDURE writeExternalDTDToClob(dt DOMDocumentType, cl IN OUT CLOB, charset
VARCHAR2);
```

PARAMETERS
```
 dt        (IN)-  DOMDocumentType
 cl      (OUT)-  CLOB to write to
 charset  (IN)-  Character set
```

RETURNS
Nothing

Element Methods

FUNCTION
isNull(elem DOMElement)

RETURN
BOOLEAN;

PURPOSE
Checks if the given DOM Element is null

SYNTAX
```
FUNCTION isNull(elem DOMElement) RETURN BOOLEAN;
```

PARAMETERS
```
 elem     (IN)-  DOMElement to check
```

RETURNS
Whether given DOM Element is null: TRUE - is null, FALSE - is not null

FUNCTION
makeNode(elem DOMElement)

RETURN
DOMNode;

PURPOSE
Casts given DOMElement to a DOMNode

SYNTAX
```
 FUNCTION makeNode(elem DOMElement) RETURN DOMNode;
```

PARAMETERS
```
 elem     (IN)-  DOMElement to cast
```

RETURNS
The DOMNode

FUNCTION
`getQualifiedName(elem DOMElement)`

RETURN
VARCHAR2;

PURPOSE
Returns the qualified name of the DOMElem

SYNTAX

FUNCTION
getQualifiedName(elem DOMElement)

RETURN
VARCHAR2;

PARAMETERS
```
 elem      (IN)-  DOMElement
```

RETURNS
The qualified name

FUNCTION
getNamespace(elem DOMElement)

RETURN
VARCHAR2;

PURPOSE
Returns the namespace of the DOMElem

SYNTAX

```
FUNCTION getNamespace(elem DOMElement) RETURN VARCHAR2;
```

PARAMETERS

```
elem     (IN)-  DOMElement
```

RETURNS

The namespace

FUNCTION

`getLocalName(elem DOMElement)`

RETURN

VARCHAR2;

PURPOSE

Returns the local name of the DOMElem

SYNTAX

```
FUNCTION getLocalName(elem DOMElement) RETURN VARCHAR2;
```

PARAMETERS

```
elem     (IN)-  DOMElement
```

RETURNS

The local name

FUNCTION **getExpandedName(elem DOMElement)** RETURN VARCHAR2;

PURPOSE

Returns the expanded name of the DOMElem

SYNTAX

```
FUNCTION getExpandedName(elem DOMElement) RETURN VARCHAR2;
```

PARAMETERS

```
elem     (IN)-  DOMElement
```

RETURNS

The expanded name

FUNCTION **getChildrenByTagName(elem DOMElement, name varchar2)** RETURN DOMNodeList;

PURPOSE

Returns the children of the DOMElem having the given tag name

SYNTAX

```
 FUNCTION getChildrenByTagName(elem DOMElement, name varchar2) RETURN
DOMNodeList;
```

PARAMETERS

```
 elem      (IN)-  DOMElement
 name      (IN)-  tag name (* matches any tag)
```

RETURNS

Children with the given tag name

FUNCTION **getElementsByTagName(elem DOMElement, name varchar2, ns VARCHAR2)** RETURN DOMNodeList;

PURPOSE

Returns the element children of the DOMElem having the given tag name and namespace

SYNTAX

```
 FUNCTION getElementsByTagName(elem DOMElement, name varchar2, ns VARCHAR2 )
RETURN DOMNodeList;
```

PARAMETERS

```
 elem      (IN)-  DOMElement
 name      (IN)-  tag name (* matches any tag)
 ns        (IN)-  namespace
```

RETURNS

Elements with the given tag name and namespace

FUNCTION **getChildrenByTagName(elem DOMElement, name varchar2, ns VARCHAR2)** RETURN DOMNodeList;

PURPOSE

Returns the children of the DOMElem having the given tag name and namespace

SYNTAX

```
 FUNCTION getChildrenByTagName(elem DOMElement, name varchar2, ns VARCHAR2 )
RETURN DOMNodeList;
```

PARAMETERS

```
 elem      (IN)-  DOMElement
 name      (IN)-  tag name (* matches any tag)
 ns        (IN)-  namespace
```

RETURNS

Children with the given tag name and namespace

FUNCTION **resolveNamespacePrefix(elem DOMElement, prefix VARCHAR2)** RETURN VARCHAR2;

PURPOSE

Resolves the given namespace prefix

SYNTAX

```
 FUNCTION resolveNamespacePrefix(elem DOMElement, prefix VARCHAR2) RETURN
VARCHAR2;
```

PARAMETERS

```
 elem      (IN)-  DOMElement
 prefix    (IN)-  namespace prefix
```

RETURNS

The resolved namespace

Entity Methods

FUNCTION **isNull(ent DOMEntity)** RETURN BOOLEAN;

PURPOSE
Checks if the given DOM Entity is null

SYNTAX
```
FUNCTION isNull(ent DOMEntity) RETURN BOOLEAN;
```

PARAMETERS
```
ent      (IN)-  DOMEntity to check
```

RETURNS
Whether given DOM Entity is null: TRUE - is null, FALSE - is not null

FUNCTION **makeNode(ent DOMEntity)** RETURN DOMNode;

PURPOSE
Casts given DOMEntity to a DOMNode

SYNTAX
```
FUNCTION makeNode(ent DOMEntity) RETURN DOMNode;
```

PARAMETERS
```
ent      (IN)-  DOMEntity to cast
```

RETURNS
The DOMNode

Entity Reference Methods

FUNCTION **isNull(eref DOMEntityReference)** RETURN BOOLEAN;

PURPOSE
Checks if the given DOM EntityReference is null

SYNTAX
```
FUNCTION isNull(eref DOMEntityReference) RETURN BOOLEAN;
```

PARAMETERS

```
eref      (IN)-  DOMEntityReference to check
```

RETURNS

Whether given DOM EntityReference is null : TRUE - is null, FALSE - is not null

FUNCTION **makeNode(eref DOMEntityReference)** RETURN DOMNode;

PURPOSE

Casts given DOMEntityReference to a DOMNode

SYNTAX

```
FUNCTION makeNode(eref DOMEntityReference) RETURN DOMNode;
```

PARAMETERS

```
eref      (IN)-  DOMEntityReference to cast
```

RETURNS

The DOMNode

Notation Methods

FUNCTION **isNull(n DOMNotation)** RETURN BOOLEAN;

PURPOSE

Checks if the given DOM Notation is null

SYNTAX

```
FUNCTION isNull(n DOMNotation) RETURN BOOLEAN;
```

PARAMETERS

```
n      (IN)-  DOMNotation to check
```

RETURNS

Whether given DOM Notation is null : TRUE - is null, FALSE - is not null

FUNCTION **makeNode(n DOMNotation)** RETURN DOMNode;

PURPOSE
Casts given DOMNotation to a DOMNode

SYNTAX
```
FUNCTION makeNode(n DOMNotation) RETURN DOMNode;
```

PARAMETERS
```
n      (IN)-  DOMNotation to cast
```

RETURNS
The DOMNode

Processing Instruction Methods

FUNCTION **isNull(pi DOMProcessingInstruction)** RETURN BOOLEAN;

PURPOSE
Checks if the given DOM ProcessingInstruction is null

SYNTAX
```
FUNCTION isNull(pi DOMProcessingInstruction) RETURN BOOLEAN;
```

PARAMETERS
```
pi      (IN)-  DOMProcessingInstruction to check
```

RETURNS
Whether given DOM ProcessingInstruction is null : TRUE - is null, FALSE - is not null

FUNCTION **makeNode(pi DOMProcessingInstruction)** RETURN DOMNode;

PURPOSE
Casts given DOMProcessingInstruction to a DOMNode

SYNTAX

```
FUNCTION makeNode(pi DOMProcessingInstruction) RETURN DOMNode;
```

PARAMETERS

```
pi      (IN)-  DOMProcessingInstruction to cast
```

RETURNS

The DOMNode

Text Methods

FUNCTION **isNull(t DOMText)** RETURN BOOLEAN;

PURPOSE

Checks if the given DOMText is null

SYNTAX

```
FUNCTION isNull(t DOMText) RETURN BOOLEAN;
```

PARAMETERS

```
t      (IN)-  DOMText to check
```

RETURNS

Whether given DOMText is null : TRUE **-** is null, FALSE **-** is not null

FUNCTION **makeNode(t DOMText)** RETURN DOMNode;

PURPOSE

Casts given DOMText to a DOMNode

SYNTAX

```
FUNCTION makeNode(t DOMText) RETURN DOMNode;
```

PARAMETERS

```
t      (IN)-  DOMText to cast
```

RETURNS

The DOMNode

Document Methods

FUNCTION **isNull(doc DOMDocument)** RETURN BOOLEAN;

PURPOSE

Checks if the given DOMDocument is null

SYNTAX

```
FUNCTION isNull(doc DOMDocument) RETURN BOOLEAN;
```

PARAMETERS

```
doc      (IN)-  DOMDocument to check
```

RETURNS

Whether given DOMDocument is null : TRUE - is null, FALSE - is not null

FUNCTION **makeNode(doc DOMDocument)** RETURN DOMNode;

PURPOSE

Casts given DOMDocument to a DOMNode

SYNTAX

```
FUNCTION makeNode(doc DOMDocument) RETURN DOMNode;
```

PARAMETERS

```
doc      (IN)-  DOMDocument to cast
```

RETURNS

The DOMNode

FUNCTION **newDOMDocument** RETURN DOMDocument;

PURPOSE
Returns a new DOM Document instance

SYNTAX
```
FUNCTION newDOMDocument RETURN DOMDocument;
```

PARAMETERS
None

RETURNS
A new DOMDocument instance

PROCEDURE **freeDocument(doc DOMDocument)**

PURPOSE

Frees Document object

SYNTAX
```
PROCEDURE freeDocument(doc DOMDocument)
```

PARAMETERS
```
doc      (IN)-  DOMDocument
```
FUNCTION **getVersion(doc DOMDocument)** RETURN VARCHAR2;

PURPOSE
Gets version information for the XML document

SYNTAX
```
FUNCTION getVersion(doc DOMDocument) RETURN VARCHAR2;
```

PARAMETERS
```
doc      (IN)-  DOMDocument
```

RETURNS
Version information

PROCEDURE **setVersion(doc DOMDocument, version VARCHAR2)**;

PURPOSE

Sets version information for the XML document

SYNTAX

```
PROCEDURE setVersion(doc DOMDocument, version VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
version  (IN)-  version information
```

RETURNS

Nothing

FUNCTION **getCharset(doc DOMDocument)** RETURN VARCHAR2;

PURPOSE

Gets character set of the XML document

SYNTAX

```
FUNCTION getCharset(doc DOMDocument) RETURN VARCHAR2;
```

PARAMETERS

```
doc      (IN)-  DOMDocument
```

RETURNS

Character set of the XML document

PROCEDURE **setCharset(doc DOMDocument, charset VARCHAR2);**

PURPOSE

Sets character set of the XML document

SYNTAX

```
PROCEDURE setCharset(doc DOMDocument, charset VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
charset  (IN)-  Character set
```

RETURNS

Nothing

FUNCTION **getStandalone(doc DOMDocument)** RETURN VARCHAR2;

PURPOSE

Gets standalone information for the XML document

SYNTAX

```
FUNCTION getStandalone(doc DOMDocument) RETURN VARCHAR2;
```

PARAMETERS

```
doc      (IN)-  DOMDocument
```

RETURNS

Standalone information

PROCEDURE **setStandalone(doc DOMDocument, value VARCHAR2);**

PURPOSE

Sets standalone information for the XML document

SYNTAX

```
PROCEDURE setStandalone(doc DOMDocument, value VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
value    (IN)-  standalone information
```

RETURNS

Nothing

PROCEDURE **writeToFile(doc DOMDocument, fileName VARCHAR2)**;

PURPOSE

Writes XML document to specified file using the database character set

SYNTAX

```
PROCEDURE writeToFile(doc DOMDocument, fileName VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
fileName (IN)-  File to write to
```

RETURNS

Nothing

PROCEDURE **writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2)**;

PURPOSE

Writes XML document to specified buffer using the database character set

SYNTAX

```
PROCEDURE writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
buffer   (OUT)- buffer to write to
```

RETURNS

Nothing

PROCEDURE **writeToClob(doc DOMDocument, cl IN OUT CLOB)**;

PURPOSE
Writes XML document to specified clob using the database character set

SYNTAX
```
PROCEDURE writeToClob(doc DOMDocument, cl IN OUT CLOB);
```

PARAMETERS
```
doc      (IN)-  DOMDocument
cl      (OUT)-  CLOB to write to
```

RETURNS
Nothing

PROCEDURE **writeToFile(doc DOMDocument, fileName VARCHAR2, charset VARCHAR2)**;

PURPOSE
Writes XML document to specified file using the given character set

SYNTAX
```
PROCEDURE writeToFile(doc DOMDocument, fileName VARCHAR2, charset VARCHAR2);
```

PARAMETERS
```
doc      (IN)-  DOMDocument
fileName (IN)-  File to write to
charset  (IN)-  Character set
```

RETURNS
Nothing

PROCEDURE **writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2, charset VARCHAR2)**;

PURPOSE
Writes XML document to specified buffer using the given character set

SYNTAX

```
 PROCEDURE writeToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2, charset
VARCHAR2);
```

PARAMETERS

```
 doc      (IN)-  DOMDocument
 buffer   (OUT)- buffer to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE **writeToClob(doc DOMDocument, cl IN OUT CLOB, charset VARCHAR2)**;

PURPOSE

Writes XML document to specified clob using the given character set

SYNTAX

```
 PROCEDURE writeToClob(doc DOMDocument, cl IN OUT CLOB, charset VARCHAR2);
```

PARAMETERS

```
 doc      (IN)-  DOMDocument
 cl      (OUT)-  CLOB to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2)**;

PURPOSE

Writes an external DTD to specified file using the database character set

SYNTAX

```
PROCEDURE writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
fileName (IN)-  File to write to
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2)**;

PURPOSE

Writes an external DTD to specified buffer using the database character set

SYNTAX

```
PROCEDURE writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
buffer   (OUT)- buffer to write to
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB)**;

PURPOSE

Writes an external DTD to specified clob using the database character set

SYNTAX

```
PROCEDURE writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
```

```
cl      (OUT)-  CLOB to write to
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2, charset VARCHAR2)**;

PURPOSE

Writes an external DTD to specified file using the given character set

SYNTAX

```
PROCEDURE writeExternalDTDToFile(doc DOMDocument, fileName VARCHAR2, charset
VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
fileName (IN)-  File to write to
charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2, charset VARCHAR2)**;

PURPOSE

Writes an external DTD to specified buffer using the given character set

SYNTAX

```
PROCEDURE writeExternalDTDToBuffer(doc DOMDocument, buffer IN OUT VARCHAR2,
charset VARCHAR2);
```

PARAMETERS

```
doc      (IN)-  DOMDocument
buffer   (OUT)- buffer to write to
charset  (IN)-  Character set
```

RETURNS

Nothing

PROCEDURE **writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB, charset VARCHAR2)**;

PURPOSE

Writes an external DTD to specified clob using the given character set

SYNTAX

```
 PROCEDURE writeExternalDTDToClob(doc DOMDocument, cl IN OUT CLOB, charset
VARCHAR2);
```

PARAMETERS

```
 doc      (IN)-  DOMDocument
 cl      (OUT)-  CLOB to write to
 charset  (IN)-  Character set
```

RETURNS

Nothing

Interface org.w3c.dom.Attr

Public interface `Attr` extends `Node.`

The `Attr` interface represents an attribute in an `Element` object. Typically the allowable values for the attribute are defined in a document type definition.

`Attr` objects inherit the `Node` interface, but since they are not actually child nodes of the element they describe, the DOM does not consider them part of the document tree. Thus, the `Node` attributes `parentNode`, `previousSibling`, and `nextSibling` have a null value for `Attr` objects. The DOM takes the view that attributes are properties of elements rather than having a separate identity from the elements they are associated with; this should make it more efficient to implement such features as default attributes associated with all elements of a given type. Furthermore, `Attr` nodes may not be immediate children of a `DocumentFragment`. However, they can be associated with `Element` nodes contained within a `DocumentFragment`. In short, users of DOM need to be aware that `Attr` nodes have some things in common with other objects inheriting the

`Node` interface, but they also are quite distinct. The attribute's effective value is determined as follows: if this attribute has been explicitly assigned any value, that value is the attribute's effective value; otherwise, if there is a declaration for this attribute, and that declaration includes a default value, then that default value is the attribute's effective value; otherwise, the attribute does not exist on this element in the structure model until it has been explicitly added. Note that the `nodeValue` attribute on the `Attr` instance can also be used to retrieve the string version of the attribute's value(s). In XML, where the value of an attribute can contain entity references, the child nodes of the `Attr` node provide a representation in which entity references are not expanded. These child nodes may be either `Text` or `EntityReference` nodes. Because the attribute type may be unknown, there are no tokenized attribute values.

getName

Returns the name of this attribute.

getSpecified

If this attribute was explicitly given a value in the original document, this is `true`; otherwise, it is `false`.

getValue

On retrieval, the value of the attribute is returned as a string.

setValue

Enter an appropriate value.

Abstracts

getName

```
public abstract String getName()
```

Returns the name of this attribute.

getSpecified

```
public abstract boolean getSpecified()
```

If this attribute was explicitly given a value in the original document, this is `true`; otherwise, it is `false`. Note that the implementation is in charge of this attribute, not the user. If the user changes the value of the attribute (even if it ends up having the same value as the default value) then the `specified` flag is automatically

flipped to `true`. To re-specify the attribute as the default value from the DTD, the user must delete the attribute. The implementation will then make a new attribute available with `specified` set to `false` and the default value (if one exists).
In summary: If the attribute has an assigned value in the document then `specified` is `true`, and the value is the assigned value. If the attribute has no assigned value in the document and has a default value in the DTD, then `specified` is `false`, and the value is the default value in the DTD. If the attribute has no assigned value in the document and has a value of #IMPLIED in the DTD, then the attribute does not appear in the structure model of the document.

getValue

```
public abstract String getValue()
```

On retrieval, the value of the attribute is returned as a string. Character and general entity references are replaced with their values.
On setting, this creates a `Text` node with the unparsed contents of the string.

setValue

```
public abstract void setValue(String value)
```

Interface org.w3c.dom.CDATASection

Public interface **CDATASection** extends Text.

CDATA sections are used to escape blocks of text containing characters that would otherwise be regarded as markup. The only delimiter that is recognized in a CDATA section is the "]]>" string that ends the CDATA section. CDATA sections can not be nested. The primary purpose is for including material such as XML fragments, without needing to escape all the delimiters.

The `DOMString` attribute of the `Text` node holds the text that is contained by the CDATA section. Note that this may contain characters that need to be escaped outside of CDATA sections and that, depending on the character encoding ("charset") chosen for serialization, it may be impossible to write out some characters as part of a CDATA section.

The `CDATASection` interface inherits the `CharacterData` interface through the `Text` interface. Adjacent `CDATASections` nodes are not merged by use of the Element.normalize() method.

Interface org.w3c.dom.CharacterData

Public interface **CharacterData** extends Node.

The `CharacterData` interface extends Node with a set of attributes and methods for accessing character data in the DOM. For clarity this set is defined here rather than on each object that uses these attributes and methods. No DOM objects correspond directly to `CharacterData`, though `Text` and others do inherit the interface from it. All `offsets` in this interface start from 0.

appendData(String)

Append the string to the end of the character data of the node.

DeleteData(int, int)

Remove a range of characters from the node.

getData()

The character data of the node that implements this interface.

getLength()

The number of characters that are available through `data` and the `substringData` method below.

insertData(int, String)

Insert a string at the specified character offset.

replaceData(int, int, String)

Replace the characters starting at the specified character offset with the specified string.

setData(String)

substringData(int, int)

Extracts a range of data from the node.

getData

```
public abstract String getData() throws DOMExecption
```

The character data of the node that implements this interface. The DOM implementation may not put arbitrary limits on the amount of data that may be stored in a `CharacterData` node. However, implementation limits may mean that the entirety of a node's data may not fit into a single `DOMString`. In such cases, the user may call `substringData` to retrieve the data in appropriately sized pieces.

THROWS

`DOMExecption`

NO_MODIFICATION_ALLOWED_ERR: Raised when the node is read only.

THROWS

`DOMExecption`

DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

Abstracts

setData

```
public abstract void setData(String data) throws DOMExecption
```

getLength

```
public abstract int getLength()
```

The number of characters that are available through `data` and the `substringData` method below. This may have the value zero, i.e., `CharacterData` nodes may be empty.

substringData

```
public abstract String substringData(int offset,
                                     int count) throws DOMExecption
```

Extracts a range of data from the node.

PARAMETERS

offset **-** Start offset of substring to extract.

count **-** The number of characters to extract.

RETURNS

The specified substring. If the sum of `offset` and `count` exceeds the `length`, then all characters to the end of the data are returned.

THROWS

`DOMExecption`

INDEX_SIZE_ERR: Raised if the specified offset is negative or greater than the number of characters in `data`, or if the specified `count` is negative.
DOMSTRING_SIZE_ERR: Raised if the specified range of text does not fit into a `DOMString`.

appendData

```
 public abstract void appendData(String arg) throws DOMExecption
```
Append the string to the end of the character data of the node. Upon success, `data` provides access to the concatenation of `data` and the `DOMString` specified.

PARAMETERS

arg - The `DOMString` to append.

THROWS

`DOMExecption`

NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

insertData

```
 public abstract void insertData(int offset,
                                 String arg) throws DOMExecption
```
Insert a string at the specified character offset.

PARAMETERS

offset - The character offset at which to insert.

arg - The `DOMString` to insert.

THROWS

`DOMExecption`

INDEX_SIZE_ERR: Raised if the specified offset is negative or greater than the number of characters in `data`.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

deleteData

```
 public abstract void deleteData(int offset,
                                 int count) throws DOMExecption
```

Remove a range of characters from the node. Upon success, `data` and `length` reflect the change.

PARAMETERS

offset - The offset from which to remove characters.

count - The number of characters to delete. If the sum of `offset` and `count` exceeds `length` then all characters from `offset` to the end of the data are deleted.

THROWS

`DOMExecption`

INDEX_SIZE_ERR: Raised if the specified offset is negative or greater than the number of characters in `data`, or if the specified `count` is negative.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

replaceData

```
 public abstract void replaceData(int offset,
                                  int count,
                                  String arg) throws DOMExecption
```

Replace the characters starting at the specified character offset with the specified string.

PARAMETERS

offset - The offset from which to start replacing.

count - The number of characters to replace. If the sum of `offset` and `count` exceeds `length` , then all characters to the end of the data are replaced (i.e., the effect is the same as a `remove` method call with the same range, followed by an `append` method invocation).

arg - The `DOMString` with which the range must be replaced.

THROWS

`DOMExecption`

INDEX_SIZE_ERR: Raised if the specified offset is negative or greater than the number of characters in `data`, or if the specified `count` is negative.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

Interface org.w3c.dom.Comment

Public interface **Comment** extends Node.

This represents the content of a comment, i.e., all the characters between the starting '`<!--`' and ending '`-->`'. Note that this is the definition of a comment in XML, and, in practice, HTML, although some HTML tools may implement the full SGML comment structure.

Interface org.w3c.dom.Document

Public interface **Document** extends Node.

The `Document` interface represents the entire HTML or XML document. Conceptually, it is the root of the document tree, and provides the primary access to the document's data.

Since elements, text nodes, comments, processing instructions, etc. cannot exist outside the context of a `Document`, the `Document` interface also contains the factory methods needed to create these objects. The `Node` objects created have a `ownerDocument` attribute which associates them with the `Document` within whose context they were created.

createAttribute(String)

Creates an `Attr` of the given name.

createCDATASection(String)

Creates a `CDATASection` node whose value is the specified string.

createComment(String)

Creates a `Comment` node given the specified string.

createDocumentFragment()

Creates an empty `DocumentFragment` object.

createElement(String)

Creates an element of the type specified.

createEntityReference(String)

Creates an EntityReference object.

createProcessingInstruction(String, String)

Creates a `ProcessingInstruction` node given the specified name and data strings.

createTextNode(String)

Creates a `Text` node given the specified string.

getDocType()

The Document Type Declaration (see `DocumentType`) associated with this document.

getDocumentElement()

This is a convenience attribute that allows direct access to the child node that is the root element of the document.

getElementsByTagName(String)

Returns a `NodeList` of all the `Element`s with a given tag name in the order in which they would be encountered in a preorder traversal of the `Document` tree.

getImplementation()

The `DOMImplementation` object that handles this document.

Abstracts

getDoctype

```
public abstract  getDoctype()
```

The Document Type Declaration (see `DocumentType`) associated with this document. For HTML documents as well as XML documents without a document type declaration this returns `null`. The DOM Level 1 does not support editing the Document Type Declaration, therefore `docType` cannot be altered in any way.

getImplementation

`public abstract  getImplementation()`

The `DOMImplementation` object that handles this document. A DOM application may use objects from multiple implementations.

getDocumentElement

`public abstract  getDocumentElement()`

This is a convenience attribute that allows direct access to the child node that is the root element of the document. For HTML documents, this is the element with the tagName "HTML".

createElement

`public abstract  createElement(String tagName) throws DOMExecption`

Creates an element of the type specified. Note that the instance returned implements the Element interface, so attributes can be specified directly on the returned object.

PARAMETERS

tagName - The name of the element type to instantiate. For XML, this is case-sensitive. For HTML, the `tagName` parameter may be provided in any case, but it must be mapped to the canonical uppercase form by the DOM implementation.

RETURNS

A new `Element` object.

THROWS

`DOMExecption`

INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.

Abstracts

createDocumentFragment

```
public abstract  createDocumentFragment()
```

Creates an empty `DocumentFragment` object.

RETURNS

A new `DocumentFragment`.

createTextNode

```
public abstract  createTextNode(String data)
```

Creates a `Text` node given the specified string.

PARAMETERS

data - The data for the node.

RETURNS

The new `Text` object.

createComment

```
public abstract  createComment(String data)
```

Creates a `Comment` node given the specified string.

PARAMETERS

data - The data for the node.

RETURNS

`The new Comment object.`

createCDATASection

```
public abstract  createCDATASection(String data) throws DOMExecption
```

Creates a `CDATASection` node whose value is the specified string.

PARAMETERS

data - The data for the `CDATASection` contents.

RETURNS

The new `CDATASection` object.

THROWS

`DOMExecption`

NOT_SUPPORTED_ERR: Raised if this document is an HTML document.

createProcessingInstruction

```
 public abstract  createProcessingInstruction(String target,String data) throws
DOMExecption
```

Creates a `ProcessingInstruction` node given the specified name and data strings.

PARAMETERS

target - The target part of the processing instruction.

data - The data for the node.

RETURNS

The new `ProcessingInstruction` object.

THROWS

`DOMExecption`

INVALID_CHARACTER_ERR: Raised if an invalid character is specified.
NOT_SUPPORTED_ERR: Raised if this document is an HTML document.

createAttribute

```
 public abstract  createAttribute(String name) throws DOMExecption
```

Creates an `Attr` of the given name. Note that the `Attr` instance can then be set on an `Element` using the `setAttribute` method.

PARAMETERS

name - The name of the attribute.

RETURNS

A new `Attr` object.

THROWS

`DOMExecption`

INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.

createEntityReference

`public abstract  createEntityReference(String name) throws DOMExecption`

Creates an EntityReference object.

PARAMETERS

name - The name of the entity to reference.

RETURNS

The new `EntityReference` object.

THROWS

INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.
NOT_SUPPORTED_ERR: Raised if this document is an HTML document.

getElementsByTagName

`public abstract  getElementsByTagName(String tagname)`

Returns a `NodeList` of all the `Element`s with a given tag name in the order in which they would be encountered in a preorder traversal of the `Document` tree.

PARAMETERS

tagname - The name of the tag to match on. The special value "*" matches all tags.

RETURNS

A new `NodeList` object containing all the matched `Element`s.

Interface org.w3c.dom.DocumentFragment

Public interface **DocumentFragment** extends Node.

DocumentFragment is a "lightweight" or "minimal" Document object. It is very common to want to be able to extract a portion of a document's tree or to create a new fragment of a document. Imagine implementing a user command like cut or rearranging a document by moving fragments around. It is desirable to have an object which can hold such fragments and it is quite natural to use a Node for this purpose. While it is true that a Document object could fulfil this role, a Document object can potentially be a heavyweight object, depending on the underlying implementation. What is really needed for this is a very lightweight object. DocumentFragment is such an object.

Furthermore, various operations -- such as inserting nodes as children of another Node -- may take DocumentFragment objects as arguments; this results in all the child nodes of the DocumentFragment being moved to the child list of this node.

The children of a DocumentFragment node are zero or more nodes representing the tops of any sub-trees defining the structure of the document. DocumentFragment nodes do not need to be well-formed XML documents (although they do need to follow the rules imposed upon well-formed XML parsed entities, which can have multiple top nodes). For example, a DocumentFragment might have only one child and that child node could be a Text node. Such a structure model represents neither an HTML document nor a well-formed XML document.

When a DocumentFragment is inserted into a Document (or indeed any other Node that may take children) the children of the DocumentFragment and not the DocumentFragment itself are inserted into the Node. This makes the DocumentFragment very useful when the user wishes to create nodes that are siblings; the DocumentFragment acts as the parent of these nodes so that the user can use the standard methods from the Node interface, such as insertBefore() and appendChild().

Interface org.w3c.dom.DocumentType

Public interface **DocumentType** extends Node.

Each Document has a doctype attribute whose value is either null or a DocumentType object. The DocumentType interface in the DOM Level 1 Core provides an interface to the list of entities that are defined for the document, and little else because the effect of namespaces and the various XML scheme efforts on DTD representation are not clearly understood as of this writing.

The DOM Level 1 doesn't support editing `DocumentType` nodes.

getEntities()

A `NamedNodeMap` containing the general entities, both external and internal, declared in the DTD.

getName()

The name of DTD; i.e., the name immediately following the `DOCTYPE` keyword.

getNotations()

A `NamedNodeMap` containing the notations declared in the DTD.

Abstracts

getName

```
public abstract String getName()
```

The name of DTD; i.e., the name immediately following the `DOCTYPE` keyword.

getEntities

```
public abstract  getEntities()
```

A `NamedNodeMap` containing the general entities, both external and internal, declared in the DTD. Duplicates are discarded. For example in:<!DOCTYPE ex SYSTEM "ex.dtd" [<!ENTITY foo "foo"> <!ENTITY bar "bar"> <!ENTITY % baz "baz">]> <ex/> the interface provides access to `foo` and `bar` but not `baz`. Every node in this map also implements the `Entity` interface.
The DOM Level 1 does not support editing entities, therefore `entities` cannot be altered in any way.

getNotations

```
public abstract  getNotations()
```

A `NamedNodeMap` containing the notations declared in the DTD. Duplicates are discarded. Every node in this map also implements the `Notation` interface.
The DOM Level 1 does not support editing notations, therefore `notations` cannot be altered in any way.

Class org.w3c.dom.DOMException

```
java.lang.Object
   |
   +---java.lang.Throwable
           |
           +---java.lang.Exception
                   |
                   +---java.lang.RuntimeException
                           |
                           +---org.w3c.dom.DOMException
```

Public abstract class **DOMException** extends RuntimeException.

DOM operations only raise exceptions in "exceptional" circumstances, i.e., when an operation is impossible to perform (either for logical reasons, because data is lost, or because the implementation has become unstable). In general, DOM methods return specific error values in ordinary processing situation, such as out-of-bound errors when using NodeList.

Implementations may raise other exceptions under other circumstances. For example, implementations may raise an implementation-dependent exception if a null argument is passed.

Some languages and object systems do not support the concept of exceptions. For such systems, error conditions may be indicated using native error reporting mechanisms. For some bindings, for example, methods may return error codes similar to those listed in the corresponding method descriptions.

INDEX_SIZE_ERR

public static final short INDEX_SIZE_ERR

DOMSTRING_SIZE_ERR

public static final short DOMSTRING_SIZE_ERR

HIERARCHY_REQUEST_ERR

public static final short HIERARCHY_REQUEST_ERR

WRONG_DOCUMENT_ERR

public static final short WRONG_DOCUMENT_ERR

INVALID_CHARACTER_ERR

public static final short INVALID_CHARACTER_ERR

NO_DATA_ALLOWED_ERR

public static final short NO_DATA_ALLOWED_ERR

NO_MODIFICATION_ALLOWED_ERR

public static final short NO_MODIFICATION_ALLOWED_ERR

NOT_FOUND_ERR

public static final short NOT_FOUND_ERR

NOT_SUPPORTED_ERR

public static final short NOT_SUPPORTED_ERR

INUSE_ATTRIBUTE_ERR

public static final short INUSE_ATTRIBUTE_ERR

DOMException

```
public DOMException(short code,
                    String message)
```

Interface org.w3c.dom.DOMImplementation

Public interface **DOMImplementation**

The `DOMImplementation` interface provides a number of methods for performing operations that are independent of any particular instance of the document object model.

The DOM Level 1 does not specify a way of creating a document instance, and hence document creation is an operation specific to an implementation. Future

Levels of the DOM specification are expected to provide methods for creating documents directly.

hasFeature(String, String)

Tests if the DOM implementation implements a specific feature.

Abstracts

hasFeature

```
 public abstract boolean hasFeature(String feature,
                                    String version)
```
Test if the DOM implementation implements a specific feature.

PARAMETERS

feature - The package name of the feature to test. In Level 1, the legal values are "HTML" and "XML" (case-insensitive).

version - This is the version number of the package name to test. In Level 1, this is the string "1.0". If the version is not specified, supporting any version of the feature will cause the method to return `true`.

RETURNS

`true` if the feature is implemented in the specified version, `false` otherwise.

Interface org.w3c.dom.Element

Public interface **Element** extends Node.

By far the vast majority of objects (apart from text) that authors encounter when traversing a document are `Element` nodes. Assume the following XML document:<elementExample id="demo"> <subelement1/> <subelement2><subsubelement/></subelement2> </elementExample>

When represented using DOM, the top node is an `Element` node for "elementExample", which contains two child `Element` nodes, one for "subelement1" and one for "subelement2". "subelement1" contains no child nodes.

Elements may have attributes associated with them; since the `Element` interface inherits from `Node`, the generic `Node` interface method `getAttributes` may be used to retrieve the set of all attributes for an element. There are methods on the

`Element` interface to retrieve either an `Attr` object by name or an attribute value by name. In XML, where an attribute value may contain entity references, an `Attr` object should be retrieved to examine the possibly fairly complex sub-tree representing the attribute value. On the other hand, in HTML, where all attributes have simple string values, methods to directly access an attribute value can safely be used as a convenience.

getAttribute(String)

Retrieves an attribute value by name.

getAttributeNode(String)

Retrieves an `Attr` node by name.

getElementsByTagName(String)

Returns a `NodeList` of all descendant elements with a given tag name, in the order in which they would be encountered in a preorder traversal of the `Element` tree.

getTagName()

The name of the element.

normalize()

Puts all `Text` nodes in the full depth of the sub-tree underneath this `Element` into a "normal" form where only markup (e.g., tags, comments, processing instructions, CDATA sections, and entity references) separates `Text` nodes, i.e., there are no adjacent `Text` nodes.

removeAttribute(String)

Removes an attribute by name.

removeAttributeNode(Attr)

Removes the specified attribute.

setAttribute(String, String)

Adds a new attribute.

setAttributeNode(Attr)

Adds a new attribute.

Abstracts

getTagName

```
public abstract String getTagName()
```

The name of the element. For example, in: <elementExample id="demo"> ... </elementExample> , `tagName` has the value `"elementExample"`. Note that this is case-preserving in XML, as are all of the operations of the DOM. The HTML DOM returns the `tagName` of an HTML element in the canonical uppercase form, regardless of the case in the source HTML document.

getAttribute

```
public abstract String getAttribute(String name)
```

Retrieves an attribute value by name.

PARAMETERS

name - The name of the attribute to retrieve.

RETURNS

The `Attr` value as a string, or the empty string if that attribute does not have a specified or default value.

setAttribute

```
public abstract void setAttribute(String name,
                                  String value) throws DOMException
```

Adds a new attribute. If an attribute with that name is already present in the element, its value is changed to be that of the value parameter. This value is a simple string, it is not parsed as it is being set. So any markup (such as syntax to be recognized as an entity reference) is treated as literal text, and needs to be appropriately escaped by the implementation when it is written out. In order to assign an attribute value that contains entity references, the user must create an `Attr` node plus any `Text` and `EntityReference` nodes, build the appropriate subtree, and use `setAttributeNode` to assign it as the value of an attribute.

PARAMETERS

name - The name of the attribute to create or alter.

value - Value to set in string form.

THROWS

`DOMException`
INVALID_CHARACTER_ERR: Raised if the specified name contains an invalid character.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

removeAttribute

```
public abstract void removeAttribute(String name) throws DOMException
```

Removes an attribute by name. If the removed attribute has a default value it is immediately replaced.

PARAMETERS

name - The name of the attribute to remove.

THROWS

`DOMException`
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

getAttributeNode

```
public abstract Attr getAttributeNode(String name)
```

Retrieves an `Attr` node by name.

PARAMETERS

name - The name of the attribute to retrieve.

RETURNS

The `Attr` node with the specified attribute name or `null` if there is no such attribute.

Abstracts

setAttributeNode

```
public abstract  Attr setAttributeNode(Attr newAttr) throws
```

Adds a new attribute. If an attribute with that name is already present in the element, it is replaced by the new one.

PARAMETERS

newAttr - The `Attr` node to add to the attribute list.

RETURNS

If the `newAttr` attribute replaces an existing attribute with the same name, the previously existing `Attr` node is returned, otherwise `null` is returned.

THROWS

`DOMException`
WRONG_DOCUMENT_ERR: Raised if `newAttr` was created from a different document than the one that created the element.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.
INUSE_ATTRIBUTE_ERR: Raised if `newAttr` is already an attribute of another `Element` object. The DOM user must explicitly clone `Attr` nodes to re-use them in other elements.

removeAttributeNode

```
public abstract Attr removeAttributeNode(Attr oldAttr) throws DOMException
```

Removes the specified attribute.

PARAMETERS

oldAttr - The `Attr` node to remove from the attribute list. If the removed `Attr` has a default value it is immediately replaced.

RETURNS

The `Attr` node that was removed.

THROWS

`DOMException`

NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.
NOT_FOUND_ERR: Raised if `oldAttr` is not an attribute of the element.

getElementsByTagName

```
public abstract  getElementsByTagName(String name)
```

Returns a `NodeList` of all descendant elements with a given tag name, in the order in which they would be encountered in a preorder traversal of the `Element` tree.

PARAMETERS

name - The name of the tag to match on. The special value "*" matches all tags.

RETURNS

A list of matching `Element` nodes.

normalize

```
public abstract void normalize()
```

Puts all `Text` nodes in the full depth of the sub-tree underneath this `Element` into a "normal" form where only markup (e.g., tags, comments, processing instructions, CDATA sections, and entity references) separates `Text` nodes, i.e., there are no adjacent `Text` nodes. This can be used to ensure that the DOM view of a document is the same as if it were saved and re-loaded, and is useful when operations (such as XPointer lookups) that depend on a particular document tree structure are to be used.

Interface org.w3c.dom.Entity

Public interface **Entity** extends Node.

This interface represents an entity, either parsed or unparsed, in an XML document. Note that this models the entity itself not the entity declaration. `Entity` declaration modeling has been left for a later Level of the DOM specification.

The `nodeName` attribute that is inherited from `Node` contains the name of the entity.

An XML processor may choose to completely expand entities before the structure model is passed to the DOM; in this case there will be no `EntityReference` nodes in the document tree.

XML does not mandate that a non-validating XML processor read and process entity declarations made in the external subset or declared in external parameter entities. This means that parsed entities declared in the external subset need not be expanded by some classes of applications, and that the replacement value of the entity may not be available. When the replacement value is available, the corresponding `Entity` node's child list represents the structure of that replacement text. Otherwise, the child list is empty.

The resolution of the children of the `Entity` (the replacement value) may be lazily evaluated; actions by the user (such as calling the `childNodes` method on the `Entity` Node) are assumed to trigger the evaluation.

The DOM Level 1 does not support editing `Entity` nodes; if a user wants to make changes to the contents of an `Entity`, every related `EntityReference` node has to be replaced in the structure model by a clone of the `Entity`'s contents, and then the desired changes must be made to each of those clones instead. All the descendants of an `Entity` node are read only.

An `Entity` node does not have any parent.

getNotationName()

For unparsed entities, the name of the notation for the entity.

getPublicId()

The public identifier associated with the entity, if specified.

getSystemId()

The system identifier associated with the entity, if specified.

Abstracts

getPublicId

```
public abstract String getPublicId()
```

The public identifier associated with the entity, if specified. If the public identifier was not specified, this is `null`.

getSystemId

```
public abstract String getSystemId()
```

The system identifier associated with the entity, if specified. If the system identifier was not specified, this is `null`.

getNotationName

```
public abstract String getNotationName()
```

For unparsed entities, the name of the notation for the entity. For parsed entities, this is `null`.

Interface org.w3c.dom.EntityReference

Public interface **EntityReference** extends Node.

`EntityReference` objects may be inserted into the structure model when an entity reference is in the source document, or when the user wishes to insert an entity reference. Note that character references and references to predefined entities are considered to be expanded by the HTML or XML processor so that characters are represented by their Unicode equivalent rather than by an entity reference. Moreover, the XML processor may completely expand references to entities while building the structure model, instead of providing `EntityReference` objects. If it does provide such objects, then for a given `EntityReference` node, it may be that there is no `Entity` node representing the referenced entity; but if such an `Entity` exists, then the child list of the `EntityReference` node is the same as that of the `Entity` node. As with the `Entity` node, all descendants of the `EntityReference` are read only.

The resolution of the children of the `EntityReference` (the replacement value of the referenced `Entity`) may be lazily evaluated; actions by the user (such as calling the `childNodes` method on the `EntityReference` node) are assumed to trigger the evaluation.

Interface org.w3c.dom.NamedNodeMap

Public interface **NamedNodeMap**

Objects implementing the `NamedNodeMap` interface are used to represent collections of nodes that can be accessed by name. Note that `NamedNodeMap` does not inherit from `NodeList`; `NamedNodeMaps` are not maintained in any particular order. Objects contained in an object implementing `NamedNodeMap` may also be accessed by an ordinal index, but this is simply to allow convenient enumeration of the contents of a `NamedNodeMap`, and does not imply that the DOM specifies an order to these Nodes.

getLength()

The number of nodes in the map.

getNamedItem(String)

Retrieves a node specified by name.

item(int)

Returns the `index`th item in the map.

removeNamedItem(String)

Removes a node specified by name.

setNamedItem(Node)

Adds a node using its `nodeName` attribute.

Abstracts

getNamedItem

```
public abstract  getNamedItem(String name)
```

Retrieves a node specified by name.

PARAMETERS

name - Name of a node to retrieve.

RETURNS

A `Node` (of any type) with the specified name, or `null` if the specified name did not identify any node in the map.

setNamedItem

```
public abstract Node setNamedItem(Node arg) throws DOMException
```

Adds a node using its `nodeName` attribute.
As the `nodeName` attribute is used to derive the name which the node must be stored under, multiple nodes of certain types (those that have a "special" string value) cannot be stored as the names would clash. This is seen as preferable to allowing nodes to be aliased.

PARAMETERS

arg - A node to store in a named node map. The node will later be accessible using the value of the `nodeName` attribute of the node. If a node with that name is already present in the map, it is replaced by the new one.

RETURNS

If the new `Node` replaces an existing node with the same name the previously existing `Node` is returned, otherwise `null` is returned.

THROWS

`DOMException`

WRONG_DOCUMENT_ERR: Raised if `arg` was created from a different document than the one that created the `NamedNodeMap`.
NO_MODIFICATION_ALLOWED_ERR: Raised if this `NamedNodeMap` is read only.
INUSE_ATTRIBUTE_ERR: Raised if `arg` is an `Attr` that is already an attribute of another `Element` object. The DOM user must explicitly clone `Attr` nodes to re-use them in other elements.

removeNamedItem

```
public abstract Node removeNamedItem(String name) throws DOMException
```

Removes a node specified by name. If the removed node is an `Attr` with a default value it is immediately replaced.

PARAMETERS

name - The name of a node to remove.

RETURNS

The node removed from the map or `null` if no node with such a name exists.

THROWS

`DOMException`

NOT_FOUND_ERR: Raised if there is no node named `name` in the map.

item

```
public abstract Node item(int index)
```

Returns the `index`th item in the map. If `index` is greater than or equal to the number of nodes in the map, this returns `null`.

PARAMETERS

index - Index into the map.

RETURNS

The node at the `index`th position in the `NamedNodeMap`, or `null` if that is not a valid index.

getLength

```
public abstract int getLength()
```

The number of nodes in the map. The range of valid child node indices is 0 to `length-1` inclusive.

Interface org.w3c.dom.Node

Public interface **Node**

The `Node` interface is the primary datatype for the entire Document Object Model. It represents a single node in the document tree. While all objects implementing the `Node` interface expose methods for dealing with children, not all objects implementing the `Node` interface may have children. For example, `Text` nodes may not have children, and adding children to such nodes results in a `DOMException` being raised.

The attributes `nodeName`, `nodeValue` and `attributes` are included as a mechanism to get at node information without casting down to the specific derived interface. In cases where there is no obvious mapping of these attributes for a specific `nodeType` (e.g., `nodeValue` for an Element or `attributes` for a Comment), this returns `null`. Note that the specialized interfaces may contain additional and more convenient mechanisms to get and set the relevant information.

appendChild(Node)

Adds the node `newChild` to the end of the list of children of this node.

cloneNode(boolean)

Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes.

getAttributes()

A `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.

getChildNodes()

A `NodeList` that contains all children of this node.

getFirstChild()

The first child of this node.

getLastChild()

The last child of this node.

getNextSibling()

The node immediately following this node.

getNodeName()

The name of this node, depending on its type; see the table above.

getNodeType()

A code representing the type of the underlying object, as defined above.

getNodeValue()

The value of this node, depending on its type; see the table above.

getOwnerDocument()

The `Document` object associated with this node.

getParentNode()

The parent of this node.

getPreviousSibling()

The node immediately preceding this node.

hasChildNodes()

This is a convenience method to allow easy determination of whether a node has any children.

insertBefore(Node, Node)

Inserts the node `newChild` before the existing child node `refChild`.

removeChild(Node)

Removes the child node indicated by `oldChild` from the list of children, and returns it.

replaceChild(Node, Node)

Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node.

setNodeValue(String)

ELEMENT_NODE

```
public static final short ELEMENT_NODE
```

ATTRIBUTE_NODE

```
public static final short ATTRIBUTE_NODE
```

TEXT_NODE

```
public static final short TEXT_NODE
```

CDATA_SECTION_NODE

```
public static final short CDATA_SECTION_NODE
```

ENTITY_REFERENCE_NODE

```
public static final short ENTITY_REFERENCE_NODE
```

ENTITY_NODE

```
public static final short ENTITY_NODE
```

PROCESSING_INSTRUCTION_NODE

```
public static final short PROCESSING_INSTRUCTION_NODE
```

COMMENT_NODE

```
public static final short COMMENT_NODE
```

DOCUMENT_NODE

```
public static final short DOCUMENT_NODE
```

DOCUMENT_TYPE_NODE

```
public static final short DOCUMENT_TYPE_NODE
```

DOCUMENT_FRAGMENT_NODE

```
public static final short DOCUMENT_FRAGMENT_NODE
```

NOTATION_NODE

```
public static final short NOTATION_NODE
```

getNodeName

```
public abstract String getNodeName()
```

The name of this node, depending on its type; see the table above.

getNodeValue

```
public abstract String getNodeValue() throws
```

The value of this node, depending on its type; see the table above.

THROWS

`DOMException`

NO_MODIFICATION_ALLOWED_ERR: Raised when the node is read only.

THROWS

DOMSTRING_SIZE_ERR: Raised when it would return more characters than fit in a `DOMString` variable on the implementation platform.

setNodeValue

```
public abstract void setNodeValue(String nodeValue) throws DOMException
```

getNodeType

```
public abstract short getNodeType()
```

A code representing the type of the underlying object, as defined above.

getParentNode

```
public abstract Node getParentNode()
```

The parent of this node. All nodes, except `Document`, `DocumentFragment`, and `Attr` may have a parent. However, if a node has just been created and not yet added to the tree, or if it has been removed from the tree, this is `null`.

getChildNodes

```
public abstract  NodeList getChildNodes()
```

A `NodeList` that contains all children of this node. If there are no children, this is a `NodeList` containing no nodes. The content of the returned `NodeList` is "live" in the sense that, for instance, changes to the children of the node object that it was created from are immediately reflected in the nodes returned by the `NodeList` accessors; it is not a static snapshot of the content of the node. This is true for every `NodeList`, including the ones returned by the `getElementsByTagName` method.

getFirstChild

```
public abstract Node getFirstChild()
```

The first child of this node. If there is no such node, this returns `null`.

getLastChild

```
public abstract Node getLastChild()
```

The last child of this node. If there is no such node, this returns `null`.

getPreviousSibling

```
public abstract Node getPreviousSibling()
```

The node immediately preceding this node. If there is no such node, this returns `null`.

getNextSibling

```
public abstract Node getNextSibling()
```

The node immediately following this node. If there is no such node, this returns `null`.

getAttributes

```
public abstract NamedNodeMap getAttributes()
```

A `NamedNodeMap` containing the attributes of this node (if it is an `Element`) or `null` otherwise.

getOwnerDocument

```
public abstract Document getOwnerDocument()
```

The `Document` object associated with this node. This is also the `Document` object used to create new nodes. When this node is a `Document` this is `null`.

insertBefore

```
public abstract Node insertBefore(Node newChild,
                                  Node refChild) throws DOMException
```

Inserts the node `newChild` before the existing child node `refChild`. If `refChild` is `null`, insert `newChild` at the end of the list of children.
If `newChild` is a `DocumentFragment` object, all of its children are inserted, in the same order, before `refChild`. If the `newChild` is already in the tree, it is first removed.

PARAMETERS

newChild - The node to insert.

refChild - The reference node, i.e., the node before which the new node must be inserted.

RETURNS

The node being inserted.

THROWS

`DOMException`

HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or if the node to insert is one of this node's ancestors.
WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node.

NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.
NOT_FOUND_ERR: Raised if `refChild` is not a child of this node.

replaceChild

```
public abstract Node replaceChild(Node newChild,
                                  Node oldChild) throws DOMException
```

Replaces the child node `oldChild` with `newChild` in the list of children, and returns the `oldChild` node. If the `newChild` is already in the tree, it is first removed.

PARAMETERS

newChild - The new node to put in the child list.

oldChild - The node being replaced in the list.

RETURNS

The node replaced.

THROWS

`DOMException`

HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or it the node to put in is one of this node's ancestors.
WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.
NOT_FOUND_ERR: Raised if `oldChild` is not a child of this node.

removeChild

```
public abstract Node removeChild(Node oldChild) throws DOMException
```

Removes the child node indicated by `oldChild` from the list of children, and returns it.

PARAMETERS

oldChild - The node being removed.

RETURNS

The node removed.

THROWS

`DOMException`

NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.
NOT_FOUND_ERR: Raised if `oldChild` is not a child of this node.

appendChild

```
public abstract  appendChild( newChild) throws DOMException
```

Adds the node `newChild` to the end of the list of children of this node. If the `newChild` is already in the tree, it is first removed.

PARAMETERSPARAMETERS

newChild - The node to add.If it is a `DocumentFragment` object, the entire contents of the document fragment are moved into the child list of this node

RETURNS

The node added.

THROWS

`DOMException`

HIERARCHY_REQUEST_ERR: Raised if this node is of a type that does not allow children of the type of the `newChild` node, or if the node to append is one of this node's ancestors.
WRONG_DOCUMENT_ERR: Raised if `newChild` was created from a different document than the one that created this node.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

hasChildNodes

```
public abstract boolean hasChildNodes()
```

This is a convenience method to allow easy determination of whether a node has any children.

RETURNS

`true` if the node has any children, `false` if the node has no children.

cloneNode

```
public abstract Node cloneNode(boolean deep)
```

Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `null`.).
Cloning an `Element` copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child `Text` node. Cloning any other type of node simply returns a copy of this node.

PARAMETERS

deep - If `true`, recursively clone the subtree under the specified node; if `false`, clone only the node itself (and its attributes, if it is an `Element`).

RETURNS

The duplicate node.

Interface org.w3c.dom.NodeList

Public interface **NodeList**

The `NodeList` interface provides the abstraction of an ordered collection of nodes, without defining or constraining how this collection is implemented.

The items in the `NodeList` are accessible via an integral index, starting from 0.

getLenth()

The number of nodes in the list.

item(int)

Returns the `index`th item in the collection.

Abstracts

item

```
public abstract Node item(int index)
```

Returns the `index`th item in the collection. If `index` is greater than or equal to the number of nodes in the list, this returns `null`.

PARAMETERS

index - Index into the collection.

RETURNS

The node at the `index`th position in the `NodeList`, or `null` if that is not a valid index.

getLength

```
public abstract int getLength()
```

The number of nodes in the list. The range of valid child node indices is 0 to `length-1` inclusive.

Interface org.w3c.dom.Notation

Public interface **Notation** extends Node.

This interface represents a notation declared in the DTD. A notation either declares, by name, the format of an unparsed entity (see section 4.7 of the XML 1.0 specification), or is used for formal declaration of Processing Instruction targets (see section 2.6 of the XML 1.0 specification). The `nodeName` attribute inherited from `Node` is set to the declared name of the notation.

The DOM Level 1 does not support editing `Notation` nodes; they are therefore read only.

A `Notation` node does not have any parent.

getPublicId()

The public identifier of this notation.

getSystemId()

The system identifier of this notation.

Abstracts

getPublicId

```
public abstract String getPublicId()
```

The public identifier of this notation. If the public identifier was not specified, this is `null`.

getSystemId

```
public abstract String getSystemId()
```

The system identifier of this notation. If the system identifier was not specified, this is `null`.

Interface org.w3c.dom.ProcessingInstruction

Public interface **ProcessingInstruction** extends Node.

The `ProcessingInstruction` interface represents a "processing instruction", used in XML as a way to keep processor-specific information in the text of the document.

getData()

The content of this processing instruction.

getTarget()

The target of this processing instruction.

setData(String)

Abstracts

getTarget

```
public abstract String getTarget()
```

The target of this processing instruction. XML defines this as being the first token following the markup that begins the processing instruction.

getData

```
public abstract String getData()
```

The content of this processing instruction. This is from the first non white space character after the target to the character immediately preceding the `?>`.

THROWS

`DOMException`
NO_MODIFICATION_ALLOWED_ERR: Raised when the node is read only.

setData

```
public abstract void setData(String data) throws
```

Interface org.w3c.dom.Text

Public interface **Text** extends CharacterData.

The `Text` interface represents the textual content (termed character data in XML) of an `Element` or `Attr`. If there is no markup inside an element's content, the text is contained in a single object implementing the `Text` interface that is the only child of the element. If there is markup, it is parsed into a list of elements and `Text` nodes that form the list of children of the element.

When a document is first made available via the DOM, there is only one `Text` node for each block of text. Users may create adjacent `Text` nodes that represent the contents of a given element without any intervening markup, but should be aware that there is no way to represent the separations between these nodes in XML or HTML, so they will not (in general) persist between DOM editing sessions. The `normalize()` method on `Element` merges any such adjacent `Text` objects into a single node for each block of text; this is recommended before employing operations that depend on a particular document structure, such as navigation with `XPointers.`

splitText(int)

Breaks this `Text` node into two Text nodes at the specified offset, keeping both in the tree as siblings.

Abstracts

splitText

```
public abstract Text splitText(int offset) throws DOMException
```

Breaks this `Text` node into two Text nodes at the specified offset, keeping both in the tree as siblings. This node then only contains all the content up to the `offset` point. And a new `Text` node, which is inserted as the next sibling of this node, contains all the content at and after the `offset` point.

PARAMETERS

offset - The offset at which to split, starting from 0.

RETURNS

The new `Text` node.

THROWS

`DOMException`

INDEX_SIZE_ERR: Raised if the specified offset is negative or greater than the number of characters in `data`.
NO_MODIFICATION_ALLOWED_ERR: Raised if this node is read only.

Part II

XDK for C Packages

This section contains Chapter 6, "XML Parser for C"

6 XML Parser for C

This chapter describes the following sections:

- Parser APIs
- W3 Sax APIs
- W3 DOM APIs
- Namespace APIs

Parser APIs

Extensible Markup Language (XML) describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language [ISO 8879]. By construction, XML documents are conforming SGML documents.

XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure.

A software module called an XML processor is used to read XML documents and provide access to their content and structure. It is assumed that an XML processor is doing its work on behalf of another module, called the application.

This C implementation of the XML processor (or parser) followed the W3C XML specification (rev REC-xml-19980210) and included the required behavior of an XML processor in terms of how it must read XML data and the information it must provide to the application.

The following is the default behavior of this parser:

1. The character set encoding is UTF-8. If all your documents are ASCII, you are encouraged to set the encoding to US-ASCII for better performance.
2. Messages are printed to stderr unless msghdlr is given.
3. A parse tree which can be accessed by DOM APIs is built unless saxcb is set to use the SAX callback APIs. Note that any of the SAX callback functions can be set to NULL if not needed.
4. The default behavior for the parser is to check that the input is well-formed but not to check whether it is valid. The flag XML_FLAG_VALIDATE can be set to validate the input. The default behavior for whitespace processing is to be fully conformant to the XML 1.0 spec, i.e. all whitespace is reported back to the application but it is indicated which whitespace is ignorable. However, some applications may prefer to set the XML_FLAG_DISCARD_WHITESPACE which will discard all whitespace between an end-element tag and the following start-element tag.

Calling Sequence

The sequence of calls to the parser can be:

`xmlinit() - xmlparse()` *or* `xmlparsebuf() - xmlterm()`

or

```
xmlinit() - xmlparse() or
xmlparsebuf() - xmlclean() - xmlparse() or
xmlparsebuf() - xmlclean() - ... - xmlterm()
```

or

```
xmlinit() - xmlparse() or
xmlparsebuf() - xmlparse() or
xmlparsebuf() - ... - xmlterm()
```

Memory

The memory callback functions memcb may be used if you wish to use your own memory allocation. If they are used, all of the functions should be specified.

The memory allocated for parameters passed to the SAX callbacks or for nodes and data stored with the DOM parse tree will not be freed until one of the following is done:

1. `xmlparse()` or `xmlparsebuf()` is called to parse another file or buffer.
2. `xmlclean()` is called.
3. `xmlterm()` is called.

Thread Safety

If threads are forked off somewhere in the midst of the init-parse-term sequence of calls, you will get unpredictable behavior and results.

Data Types Index

oratext	String pointer
xmlctx	Master XML context
xmlmemcb	Memory callback structure (optional)
xmlsaxcb	SAX callback structure (SAX only)
ub4	32-bit (or larger) unsigned integer
uword	Native unsigned integer

Function Index

xmlinit	Initialize XML parser
xmlclean	Clean up memory used during parse
xmlparse	Parse a file
xmlparsebuf	Parse a buffer
xmlterm	Shut down XML parser
createDocument	Create a new document
isStandalone	Return document's standalone flag

Data Structures and Types

oratext

```
typedef unsigned char oratext;
```

xmlctx

```
typedef struct xmlctx xmlctx;
```

Note: The contents of xmlctx are private and must not be accessed by users.

---

xmlmemcb

```
struct xmlmemcb
{
   void *(*alloc)(void *ctx, size_t size);
   void  (*free)(void *ctx, void *ptr);
   void *(*realloc)(void *ctx, void *ptr, size_t size);
};
typedef struct xmlmemcb xmlmemcb;
```

Note: This is the memory callback structure.

---

xmlsaxcb

```
struct xmlsaxcb
{
   sword (*startDocument)(void *ctx);
   sword (*endDocument)(void *ctx);
   sword (*startElement)(void *ctx, const oratext *name,
                              const struct xmlattrs *attrs);
   sword (*endElement)(void *ctx, const oratext *name);
   sword (*characters)(void *ctx, const oratext *ch, size_t len);
   sword (*ignorableWhitespace)(void *ctx, const oratext *ch,
                                          size_t len);
   sword (*processingInstruction)(void *ctx, const oratext *target,
                                       const oratext *data);
   sword (*notationDecl)(void *ctx, const oratext *name,
                              const oratext *publicId,
                              const oratext *systemId);
   sword (*unparsedEntityDecl)(void *ctx, const oratext *name,
                                    const oratext *publicId,
                                    const oratext *systemId,
                                    const oratext *notationName);
   sword (*nsStartElement)(void *ctx, const oratext *qname,
                                   const oratext *local,
                                   const oratext *namespace,
                                   const struct xmlattrs *attrs);
   /* The following 8 fields are reserved for future use. */
   void (*empty1)();
   void (*empty2)();
   void (*empty3)();
   void (*empty4)();
   void (*empty5)();
   void (*empty6)();
   void (*empty7)();
   void (*empty8)();
};
typedef struct xmlsaxcb xmlsaxcb;
```

Note: Callbacks for SAX-like API.

ub4

```
typedef unsigned int ub4;
```

uword

```
typedef unsigned int uword;
```

Functions

xmlinit

Purpose

Initializes the C XML parser. It must be called before any parsing can take place.

Syntax

```
xmlctx *xmlinit(uword *err, const oratext *encoding,
                void (*msghdlr)(void *msgctx, const oratext *msg, ub4 errcode),
                 void *msgctx, const xmlsaxcb *saxcb, void *saxcbctx,
                 const xmlmemcb *memcb, void *memcbctx, const oratext *lang);
```

Parameters

```
err      (OUT)- The error, if any
encoding (IN) - default character set encoding
msghdlr  (IN) - Error message handler function
msgctx   (IN) - Context for the error message handler
saxcb    (IN) - SAX callback structure filled with function pointers
saxcbctx (IN) - Context for SAX callbacks
memcb    (IN) - Memory function callbacks
memcbctx (IN) - Context for the memory function callbacks
lang     (IN) - Language for error messages
```

Comments

Do not call any other XML parser functions if this is not successful!

This function should only be called once before starting the processing of one or more XML files. xmlterm() should be called after all processing of XML files has completed.

Error codes: `XMLERR_LEH_INIT, XMLERR_BAD_ENCODING, XMLERR_NLS_INIT, XMLERR_NO_MEMORY, XMLERR_NULL_PTR`

All values may be NULL except for err.

By default, the character set encoding is UTF-8. If all your documents are ASCII, you are encouraged to set the encoding to US-ASCII for better performance.

By default, messages are printed to stderr unless msghdlr is given.

By default, a parse tree is built (accessible by DOM APIs) unless saxcb is set (in which case the SAX callback APIs are invoked). Note that any of the SAX callback functions can be set to NULL if not needed.

The memory callback functions memcb may be used if you wish to use your own memory allocation. If they are used, all of the functions should be specified.

The parameters msgctx, saxcbctx, and memcbctx are structures that you may define and use to pass information to your callback routines for the message handler, SAX functions, or memory functions, respectively. They should be set to NULL if your callback functions do not need any additional information passed in to them.

The lang parameter is not used currently and may be set to NULL. It will be used in future releases to determine the language of the error messages.

xmlclean

Purpose

Frees any memory used during the previous parse.

Syntax

```
void xmlclean(xmlctx *ctx);
```

Parameters

ctx (IN) - The XML parser context

Comments

This function is provided as a convenience for those who want to parse multiple files but would like to free the memory used for parses before the subsequent call to `xmlparse()` or `xmlparsebuf()`.

xmlparse

Purpose

Invokes the XML parser on an input file. The parser must have been initialized successfully with a call to `xmlinit()` first.

Syntax

```
uword xmlparse(xmlctx *ctx, const oratext *filename, const oratext *encoding,
ub4 flags);
```

Parameters

```
ctx      (IN/OUT) - The XML parser context
filename (IN) - path to XML document
encoding (IN) - default character set encoding
flags    (IN) - what options to use
```

Comments

Flag bits must be OR'd to override the default behavior of the parser. The following flag bits may be set:

- `XML_FLAG_VALIDATE` turns validation on.
- `XML_FLAG_DISCARD_WHITESPACE` will discard whitespace where it appears to be insignificant.

The default behavior is to not validate the input. The default behavior for whitespace processing is to be fully conformant to the XML 1.0 spec, i.e. all whitespace is reported back to the application but it is indicated which whitespace is ignorable. However, some applications may prefer to set the XML_FLAG_DISCARD_WHITESPACE which will discard all whitespace between an end-element tag and the following start-element tag.

The memory passed to the SAX callbacks or stored with the DOM parse tree will not be freed until one of the following is done:

1. `xmlparse()` or `xmlparsebuf()` is called to parse another file.
2. `xmlclean()` is called.
3. `xmlterm()` is called.

This function will free any memory used during the previous parse.

xmlparsebuf

Purpose

Invokes the XML parser on a buffer. The parser must have been initialized successfully with a call to `xmlinit()` first.

Syntax

```
uword xmlparsebuf(xmlctx *ctx, const oratext *buffer, size_t len, const oratext
*encoding, ub4 flags);
```

Parameters

```
ctx      (IN/OUT) - The XML parser context
buffer   (IN) - file to be parsed
len      (IN) - length of the buffer
encoding (IN) - default character set encoding
flags    (IN) - what options to use
```

Comments

This function is identical to xmlparse() except that input is taken from the user's buffer instead of from an external file.

xmlterm

Purpose

Terminates the XML parser. It should be called after `xmlinit()`, and before exiting the main program.

Syntax

```
uword xmlterm(xmlctx *ctx);
```

Parameters

```
ctx (IN) - the XML parser context
```

Comments

This function will free any memory used during the previous parse. No additional XML parser calls can be made until `xmlinit()` is called.

createDocument

Purpose

Creates a new document in memory.

Syntax

```
xmlnode* createDocument(xmlctx *ctx)
```

Parameters

```
ctx (IN) - the XML parser context
```

Comments

This function is used when constructing a new document in memory. An XML document is always rooted in a node of type `DOCUMENT_NODE`-- this function creates that root node and sets it in the context. There can be only one current document and hence only one document node; if one already exists, this function does nothing and returns NULL.

isStandalone

Purpose

Return value of document's *standalone* flag.

Syntax

```
boolean isStandalone(xmlctx *ctx)
```

Parameters

```
ctx (IN) - the XML parser context
```

Comments

This function returns the boolean value of the document's standalone flag, as specified in the `<?xml?>` processing instruction.

XSLT API

XSLT is a language for tranforming XML documents into other XML documents.

XSLT is designed for use as part of XSL, which is a stylesheet language for XML. In addition to XSLT, XSL includes an XML vocabulary for specifying formatting. XSL specifies the styling of an XML document by using XSLT to describe how the document is transformed into another XML document that uses the formatting vocabulary.

XSLT is also designed to be used indepently of XSL. However, XSLT is not intended as a completely general-purpose XML transformation language. Rather it is designed primarily for the kinds of transformation that are needed when XSLT is used as part of XSL.

A transformation expressed in XSLT describes rules for transforming a source tree into a result tree. The transformation is achieved by associating patterns with templates. A pattern is matched against elements in the source tree. A template is instantiated to create part of the result tree. The result tree is separate from the source tree. The structure of the result tree can be completely different from the structure of the source tree. In constructing the result tree, elements from the source tree can be filtered and reordered, and arbitrary structure can be added.

A transformation expressed in XSLT is called a stylesheet. This is because, in the case when XSLT is transforming into the XSL formatting vocabulary, the transformation functions as a stylesheet.

A stylesheet contains a set of template rules. A template rule has two parts: a pattern which is matched against nodes in the source tree and a template which can be instantiated to form part of the result tree. This allows a stylesheet to be applicable to a wide class of documents that have similar source tree structures.

A template is instantiated for a particular source element to create part of the result tree. A template can contain elements that specify literal result element structure. A template can also contain elements from the XSLT namespace that are instructions for creating result tree fragments. When a template is instantiated, each instruction is executed and replaced by the result tree fragment that it creates. Instructions can select and process descendant source elements. Processing a descendant element creates a result tree fragment by finding the applicable template rule and instantiating its template. Note that elements are only processed when they have been selected by the execution of an instruction. The result tree is constructed by finding the template rule for the root node and instantiating its template.

A software module called an XSL processor is used to read XML documents and transform them into other XML documents with different styles.

The C implementation of the XSL processor followed the XSL Transformations standard (version 1.0, November 16, 1999) and included the required behavior of an XSL processor as specified in the XSLT specification.

Data Structures and Types

```
uword
xmlctx
xmlnode
```

Functions

```
xmlprocess(xmlctx *docctx, xmlctx *xslctx, xmlctx *resctx, xmlnode **result)
```

Processes XSL Stylesheet with XML document source and returns success or an error code.

Data Structure and Type Description

uword

```
typedef unsigned int uword;
```

xmlctx

```
typedef struct xmlctx xmlctx;
```

Note: The contents of `xmlctx` are private and must not be accessed by users.

---

xmlnode

```
typedef struct xmlnode xmlnode;
```

Note: The contents of xmlnode are private and must not be accessed by users.

Function Prototypes

xslprocess

Purpose

This function processes an XSL Stylesheet with an XML document source.

Syntax

```
 uword xslprocess(xmlctx *docctx, xmlctx *xslctx, xmlctx *resctx, xmlnode
**result);
```

Parameters

xmlctx (IN / OUT) - The XML document context

xslctx (IN) - The XSL stylesheet context

resctx (IN) - The result document fragment context

result (IN / OUT) - The result document fragment node

W3C SAX APIs

SAX is a standard interface for event-based XML parsing, developed collaboratively by the members of the XML-DEV mailing list.

There are two major types of XML (or SGML) APIs:

1. tree-based APIs, and
2. event-based APIs.

A tree-based API compiles an XML document into an internal tree structure, then allows an application to navigate that tree using the Document Object Model (DOM), a standard tree-based API for XML and HTML documents.

An event-based API, on the other hand, reports parsing events (such as the start and end of elements) directly to the application through callbacks, and does not usually build an internal tree. The application implements handlers to deal with the different events, much like handling events in a graphical user interface.

Tree-based APIs are useful for a wide range of applications, but they often put a great strain on system resources, especially if the document is large (under very controlled circumstances, it is possible to construct the tree in a lazy fashion to avoid some of this problem). Furthermore, some applications need to build their own, different data trees, and it is very inefficient to build a tree of parse nodes, only to map it onto a new tree.

In both of these cases, an event-based API provides a simpler, lower-level access to an XML document: you can parse documents much larger than your available system memory, and you can construct your own data structures using your callback event handlers.

To use SAX, an xmlsaxcb structure is initialized with function pointers and passed to the *xmlinit()* call. A pointer to a user-defined context structure may also be included; that context pointer will be passed to each SAX function.

The SAX callback structure:

```
typedef struct
{
   sword (*startDocument)(void *ctx);
   sword (*endDocument)(void *ctx);
   sword (*startElement)(void *ctx, const oratext *name, const struct xmlarray
*attrs);
   sword (*endElement)(void *ctx, const oratext *name);
   sword (*characters)(void *ctx, const oratext *ch, size_t len);
   sword (*ignorableWhitespace)(void *ctx, const oratext *ch, size_t len);
   sword (*processingInstruction)(void *ctx, const oratext *target, const
oratext *data);
   sword (*notationDecl)(void *ctx, const oratext *name,
                         const oratext *publicId, const oratext *systemId);
   sword (*unparsedEntityDecl)(void *ctx, const oratext *name, const oratext
*publicId,
                               const oratext *systemId, const oratext
*notationName);
   sword (*nsStartElement)(void *ctx, const oratext *qname,
                           const oratext *local, const oratext *nsp,
                   const struct xmlnodes *attrs);
} xmlsaxcb;
```

Data Structures and Types

Callback Functions conforming to the SAX standard:

```
(void *ctx, const oratext *ch, size_t len)
```

Receive notification of character data inside an element.

```
(void *ctx)
```

Receive notification of the end of the document.

```
(void *ctx, const oratext *name)
```

Receive notification of the end of an element.

```
(void *ctx, const oratext *ch, size_t len)
```

Receive notification of ignorable whitespace in element content.

```
(void *ctx, const oratext *name, const oratext *publicId, const oratext
*systemId)
```

Receive notification of a notation declaration.

```
(void *ctx, const oratext *target, const oratext *data)
```

Receive notification of a processing instruction.

```
(void *ctx)
```
Receive notification of the beginning of the document.

```
(void *ctx, const oratext *name, const struct xmlattrs *attrs)
```

Receive notification of the start of an element.

```
(void *ctx, const oratext *name, const oratext *publicId, const oratext
*systemId,
const oratext *notationName)
```

Receive notification of an unparsed entity declaration.

Non-SAX Callback Functions

```
(void *ctx, const oratext *qname, const oratext *local, const oratext
*namespace, const struct xmlattrs *attrs)
```

Receive notification of the start of a namespace for an element.

Data Structure and Type Description

oratext

```
typedef unsigned char oratext;
```

sword

```
typedef signed int sword;
```

xmlattrs

```
typedef struct xmlattrs xmlattrs;
```

Note: The contents of xmlattrs are private and must not be accessed by users.

Function Prototypes

characters

Purpose
This callback function receives notification of character data inside an element.

Syntax
```
sword (*characters)(void *ctx, const oratext *ch, size_t len);
```

Parameters
ctx (IN) - client context pointer

ch (IN) - the characters

len (IN) - number of characters to use from the character pointer

Comments

endDocument

Purpose
This callback function receives notification of the end of the document.

Syntax
```
sword (*endDocument)(void *ctx);
```

Parameters
ctx (IN) - client context

Comments

endElement

Purpose
This callback function receives notification of the end of an element.

Syntax
```
sword (*endElement)(void *ctx, const oratext *name);
```

Parameters
ctx (IN) - client context

name (IN) - element type name

Comments

ignorableWhitespace

Purpose
This callback function receives notification of ignorable whitespace in element content.

Syntax
```
sword (*ignorableWhitespace)(void *ctx, const oratext *ch, size_t len);
```

Parameters
ctx (IN) - client context

ch (IN) - whitespace characters

len (IN) - number of characters to use from the character pointer

Comments

notationDecl

Purpose

This callback function receives notification of a notation declaration.

Syntax

```
sword (*notationDecl)(void *ctx, const oratext *name, const oratext *publicId,
const oratext *systemId);
```

Parameters

ctx (IN) - client context

name (IN) - notation name

publicId (IN) - notation public identifier, or null if not available

systemId (IN) - notation system identifier

Comments

processingInstruction

Purpose

This callback function receives notification of a processing instruction.

Syntax

```
sword (*processingInstruction)(void *ctx, const oratext *target, const oratext
*data);
```

Parameters

ctx (IN) - client context

target (IN) - processing instruction target

data (IN) - processing instruction data, or null if none is supplied

Comments

startDocument

Purpose
This callback function receives notification of the beginning of the document.

Syntax
```
sword (*startDocument)(void *ctx);
```

Parameters
ctx (IN) - client context

Comments

startElement

Purpose
This callback function receives notification of the beginning of an element.

Syntax
```
sword (*startElement)(void *ctx, const oratext *name, const struct xmlattrs
*attrs);
```

Parameters
ctx (IN) - client context

name (IN) - element type name

attrs (IN) - specified or defaulted attributes

Comments

unparsedEntityDecl

Purpose

This callback function receives notification of an unparsed entity declaration.

Syntax

```
sword (*unparsedEntityDecl)(void *ctx, const oratext *name, const oratext
*publicId, const oratext *systemId,
         const oratext *notationName);
```

Parameters

ctx (IN) - client context

name (IN) - entity name

publicId (IN) - entity public identifier, or null if not available

systemId (IN) - entity system identifier

notationName (IN) - name of the associated notation

Comments

nsStartElement

Purpose

This callback function receives notification of the start of a namespace for an element.

Syntax

```
sword (*nsStartElement)(void *ctx, const oratext *qname, const oratext *local,
const oratext *namespace,
         const struct xmlattrs *attrs));
```

Parameters

ctx (IN) - client context

qname (IN) - element fully qualified name local (IN) - element local name

namespace (IN) - element namespace (URI)

attrs (IN) - specified or defaulted attributes

Comments

W3C DOM APIs

The *Document Object Model* (**DOM**) is an application programming interface (API) for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. In the DOM specification, the term *document* is used in the broad sense -- increasingly, XML is being used as a way of representing many different kinds of information that may be stored in diverse systems, and much of this would traditionally be seen as data rather than as documents. Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data.

With the DOM, programmers can build documents, navigate their structure, and add, modify, or delete elements and content. Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the DOM, with a few exceptions -- in particular, the DOM interfaces for the XML internal and external subsets have not yet been specified.

One important objective of the W3C specification for the DOM is to provide a standard programming interface that can be used in a wide variety of environments and applications. The DOM is designed to be used with any programming language. Since the DOM standard is object-oriented, for this C adaptation, some changes had to be made:

- Reused function names had to be expanded, e.g. getValue() in the attribute class is given the unique name getAttrValue(), matching the pattern established by getNodeValue().
- Also, some functions were added to extend the DOM. For example, there is no function defined which returns the number of children of a node, so numChildNodes() was invented, etc.

The implementation of this C DOM interface follows REC-DOM-Level-1-19981001.

Data Structures and Types

boolean	Boolean value, TRUE or FALSE
oratext	String pointer
xmlcpmod	Content model node modifier
xmlctx	Master XML parser context
xmlnode	Document node
xmlnodes	Array of nodes
xmlntype	Node type enumeration

DOM Functions

appendChild	Append child node to current node
appendData	Append character data to end of node's current data
cloneNode	Create a new node identical to the current one
createAttribute	Create an new attribute for an element node
createCDATASection	Create a CDATA_SECTION node
createComment	Create a COMMENT node
createDocumentFragment	Create a DOCUMENT_FRAGMENT node
createElement	Create an ELEMENT node
createEntityReference	Create an ENTITY_REFERENCE node
createProcessingInstruction	Create a PROCESSING_INSTRUCTION (PI) node
createTextNode	Create a TEXT node
deleteData	Remove substring from a node's character data
getAttrName	Return an attribute's name
getAttrSpecified	Return value of attribute's *specified* flag [DOM getSpecified]
getAttrValue	Return an attribute's value (definition) [DOM getValue]
getAttribute	Return the value of an attribute
getAttributeIndex	Return an element's attribute given its index
getAttributeNode	Get an element's attribute node given its name [DOM getName]
getAttributes	Return array of element's attributes

getCharData	Return character data for a TEXT node [DOM getData]
getCharLength	Return length of TEXT node's character data [DOM getLength]
getChildNode	Return indexed node from array of nodes [DOM item]
getChildNodes	Return array of node's children
getContentModel	Returns the content model for an element from the DTD [DOM extension]
getDocument	Return top-level DOCUMENT node [DOM extension]
getDocumentElement	Return highest-level (root) ELEMENT node
getDocType	Return current DTD
getDocTypeEntities	Return array of DTD's general entities
getDocTypeName	Return name of DTD
getDocTypeNotations	Return array of DTD's notations
getElementsByTagName	Return list of elements with matching name
getEntityNotation	Return an entity's NDATA [DOM getNotation]
getEntityPubID	Return an entity's public ID [DOM getPublicId]
getEntitySysID	Return an entity's system ID [DOM getSystemId]
getFirstChild	Return the first child of a node
getImplementation	Return DOM-implementation structure (if defined)
getLastChild	Return the last child of a node
getModifier	Returns a content model node's '?', '*', or '+' modifier [DOM extension
getNextSibling	Return a node's next sibling
getNamedItem	Returns the named node from a list of nodes
getNodeMapLength	Returns number of entries in a NodeMap [DOM getLength]
getNodeName	Returns a node's name
getNodeType	Returns a node's type code (enumeration)
getNodeValue	Returns a node's "value", its character data
getNotationPubID	Returns a notation's public ID [DOM getPublicId]
getNotationSysID	Returns a notation's system ID [DOM getSystemId]
getOwnerDocument	Returns the DOCUMENT node containing the given node
getPIData	Returns a processing instruction's data [DOM getData]
getPITarget	Returns a processing instruction's target [DOM getTarget]

getParentNode	Returns a node's parent node
getPreviousSibling	Returns a node's "previous" sibling
getTagName	Returns a node's "tagname", same as name for now
hasAttributes	Determine if element node has attributes [DOM extension]
hasChildNodes	Determine if node has children
hasFeature	Determine if DOM implementation supports a specific feature
insertBefore	Inserts a new child node before the given reference node
insertData	Inserts new character data into a node's existing data
isStandalone	Determine if document is standalone [DOM extension]
nodeValid	Validate a node against the current DTD [DOM extension]
normalize	Normalize a node by merging adjacent TEXT nodes
numAttributes	Returns number of element node's attributes [DOM extension]
numChildNodes	Returns number of node's children [DOM extension]
removeAttribute	Removes an element's attribute given its names
removeAttributeNode	Removes an element's attribute given its pointer
removeChild	Removes a node from its parents list of children
removeNamedItem	Removes a node from a list of nodes given its name
replaceChild	Replace one node with another
replaceData	Replace a substring of a node's character data with another string
setAttribute	Sets (adds or replaces) a new attribute for an element node given attribute's name and value
setAttributeNode	Sets (adds or replaces) a new attribute for an element node given pointer to the new attribute
setNamedItem	Sets (adds or replaces) a new node in a parent's list of children
setNodeValue	Sets a node's "value" (character data)
setPIData	Sets a processing instruction's data [DOM setData]
splitText	Split a node's character data into two parts
substringData	Return a substring of a node's character data

Data Structures and Types

boolean

```
typedef int boolean;
```

oratext

```
typedef unsigned char oratext;
```

xmlcpmod

Content model node modifiers, see `getModifier`.

```
XMLCPMOD_NONE  = 0                /* no modifier */
XMLCPMOD_OPT   = 1                /* '?' optional */
XMLCPMOD_0MORE = 2                /* '*' zero or more */
XMLCPMOD_1MORE = 3                /* '+' one or more */
```

xmlctx

```
typedef struct xmlctx xmlctx;
```

Note: The contents of xmlctx are private and must not be accessed by users.

xmlnode

```
typedef struct xmlnode xmlnode;
```

Note: The contents of xmlnode are private and must not be accessed by users.

---

xmlnodes

```
typedef struct xmlnodes xmlnodes;
```

Note: The contents of xmlnodes are private and must not be accessed by users.

---

xmlntype

Parse tree node types, see `getNodeType`. Names and values match DOM specification.

```
ELEMENT_NODE                = 1    /* element */
ATTRIBUTE_NODE              = 2    /* attribute */
TEXT_NODE                   = 3    /* char data not escaped by CDATA */
CDATA_SECTION_NODE          = 4    /* char data escaped by CDATA */
ENTITY_REFERENCE_NODE       = 5    /* entity reference */
ENTITY_NODE                 = 6    /* entity */
PROCESSING_INSTRUCTION_NODE = 7    /* processing instruction */
COMMENT_NODE                = 8    /* comment */
DOCUMENT_NODE               = 9    /* document */
DOCUMENT_TYPE_NODE          = 10   /* DTD */
DOCUMENT_FRAGMENT_NODE      = 11   /* document fragment */
NOTATION_NODE               = 12   /* notation */
```

Function Prototypes

appendChild

Purpose

Adds new node to the end of the list of children for the given parent and returns the node added.

Syntax

```
xmlnode *appendChild(xmlctx *ctx, xmlnode *parent, xmlnode *newnode)
```

Parameters

ctx	(IN)	XML context
parent	(IN)	parent node
newnode	(IN)	new node to append

Example

```
xmlnode *node, *parent;
...

if (node = createElement(ctx, "node"))
    appendChild(ctx, parent, node);
```

appendData

Purpose

Append the given string to the character data of a TEXT or CDATA node.

Syntax

```
void appendData(xmlctx *ctx, xmlnode *node, const oratext *arg)
```

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node
arg	(IN)	new data to append

Example

```
xmlnode *node;
...
getNodeValue(node) -> "foo"
appendData(ctx, node, "bar");
getNodeValue(node) -> "foobar"
```

cloneNode

Purpose

Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (`parentNode` returns `NULL`).

Cloning an Element copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child Text node. Cloning any other type of node simply returns a copy of this node.

A *deep* clone differs in that the node's children are also recursively cloned instead of just pointed-to.

Syntax

```
xmlnode *cloneNode(xmlctx *ctx, const xmlnode *old, boolean deep)
```

Parameters

ctx	(IN)	XML context
old	(IN)	old node to clone
deep	(IN)	recursion flag

createAttribute

Purpose

Create a new ATTRIBUTE node with the given name and value. The new node is unattached and must be added to an element node with `setAttributeNode`.

Syntax

```
xmlnode *createAttribute(xmlctx *ctx, const oratext *name, const oratext *value)
```

Parameters

ctx	(IN)	XML context
name	(IN)	name of new attribute
value	(IN)	value of new attribute

Example

```
xmlnode *attr, *elem;
...
if (attr = createAttribute(ctx, "attr1", "value1"))
{
    setAttributeNode(ctx, elem, attr, NULL);
}
```

createCDATASection

Purpose

Create a new CDATA node.

Syntax

```
xmlnode *createCDATASection(xmlctx *ctx, const oratext *data)
```

Parameters

ctx	(IN)	XML context
data	(IN)	CDATA body

Example

```
xmlnode *node, *parent;
...
if (node = createCDATASection(ctx, "<greeting>H'o!</greeting>"))
    appendChild(ctx, parent, node);
```

createComment

Purpose

Create a new COMMENT node.

Syntax

```
xmlnode *createComment(xmlctx *ctx, const oratext *data)
```

Parameters

ctx	(IN)	XML context
data	(IN)	text of comment

Example

```
xmlnode *node, *parent;
...
if (node = createComment(ctx, "From here on this document is unfinished"))
    appendChild(ctx, parent, node);
```

createDocumentFragment

Purpose

Create a new DOCUMENT_FRAGMENT node. A document fragment is a lightweight document object that contains one or more children, but does not have the overhead of a full document. It can be used in some operations (inserting for example) in place of a simple node, in which case all the fragment's children are operated on instead of the fragment node itself.

Syntax

```
xmlnode *createDocumentFragment(xmlctx *ctx)
```

Parameters

ctx	(IN)	XML context

Example

```
xmlnode *frag, *fragelem, *fragtext;
...
if ((frag = createDocumentFragment(ctx)) &&
    (fragelem = createElement(ctx, (oratext *) "FragElem")) &&
    (fragtext = createTextNode(ctx, (oratext *) "FragText")))
{
    appendChild(ctx, frag, fragelem);
    appendChild(ctx, frag, fragtext);
}
```

createElement

Purpose

Create a new ELEMENT node.

Syntax

```
xmlnode *createElement(xmlctx *ctx, const oratext *elname)
```

Parameters

ctx	(IN)	XML context
elname	(IN)	name of new element

Example

```
xmlnode *node, *parent;
...
if (node = createElement(ctx, "BOOK"))
    appendChild(ctx, parent, node);
```

createEntityReference

Purpose

Create a new ENTITY_REFERENCE node.

Syntax

```
xmlnode *createEntityReference(xmlctx *ctx, const oratext *name)
```

Parameters

ctx	(IN)	XML context
name	(IN)	name of entity to reference

Example

```
xmlnode *node, *parent;
...
if (node = createEntityReference(ctx, "homephone"))
    appendChild(ctx, parent, node);
```

createProcessingInstruction

Purpose

Create a new PROCESSING_INSTRUCTION node with the given target and contents.

Syntax

```
xmlnode *createProcessingInstruction(xmlctx *ctx, const oratext *target, const
oratext *data)
```

Parameters

ctx	(IN)	XML context
target	(IN)	PI target
data	(IN)	PI definition

Example

```
xmlnode *node, *parent;
...
if (node = createProcessingInstruction(ctx, "target", "definition"))
    appendChild(ctx, parent, node);
```

createTextNode

Purpose

Create a new TEXT node with the given contents.

Syntax

```
xmlnode *createTextNode(xmlctx *ctx, const oratext *data)
```

Parameters

ctx	(IN)	XML context
data	(IN)	data for node

Example

```
xmlnode *node, *parent;
...
if (node = createTextNode(ctx, "riverrun, past Eve and Adam's..."))
    appendChild(ctx, parent, node);
```

deleteData

Purpose

Delete a substring from the node's character data.

Syntax

```
void deleteData(xmlctx *ctx, xmlnode *node, ub4 offset, ub4 count)
```

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node
offset	(IN)	offset of start of substring (0 is first char)
count	(IN)	length of substring

Example

```
xmlnode *node;
...
getNodeValue(node) -> "phoenix"
deleteData(ctx, node, 2, 1);
getNodeValue(node) -> "phenix"
```

getAttribute

Purpose

Returns one attribute from an array of attributes, given an index (starting at 0). Fetch the attribute name and/or value (with `getAttrName` and `getAttrValue`). On error, returns `NULL`.

Syntax

```
const oratext *getAttribute(const xmlnode *node, const oratext *name)
```

Parameters

node	(IN)	node whose attribtutes to scan
name	(IN)	name of the attribute

Example

```
xmlnode  *node, *attr;
xmlnodes *nodes;
const oratext *attrval;
...
if (nodes = getAttributes(node))
{
    attr = getAttributeIndex(nodes, 1);/* second attribute */
    attrval = getAttribute(attr, "foo");
    ...
}
```

getAttributeIndex

Purpose

Returns one attribute from an array of attributes, given an index (starting at 0). Fetch the attribute name and/or value (with `getAttrName` and `getAttrValue`). On error, returns `NULL`.

Syntax

```
xmlnode *getAttributeIndex(const xmlnodes *attrs, size_t index)
```

Parameters

attrs	(IN)	pointer to attribute nodes structure (as returned by getAttributes)
index	(IN)	zero-based attribute# to return

Example

```
xmlnode  *node, *attr;
xmlnodes *nodes;
...
if (nodes = getAttributes(node))
{
    attr = getAttributeIndex(nodes, 1);      /* second attribute */
    ...
}
```

getAttributeNode

Purpose

Returns a pointer to the element node's attribute of the given name. If no such thing exists, returns `NULL`.

Syntax

```
xmlnode *getAttributeNode(const xmlnode *elem, const oratext *name)
```

Parameters

elem	(IN)	pointer to element node
name	(IN)	name of attribute

Example

```
xmlnode *node, *attr;
...
if (attr = getAttributeNode(elem, "attr1"))
    ...
```

getAttributes

Purpose

Returns an array of all attributes of the given node. This pointer may then be passed to `getAttribute` to fetch individual attribute pointers, or to `numAttributes` to return the total number of attributes. If no attributes are defined, returns `NULL`.

Syntax

```
xmlnodes *getAttributes(const xmlnode *node)
```

Parameters

node	(IN)	node whose attributes to return

Example

```
xmlnode  *node;
xmlnodes *nodes;
...
if (nodes = getAttributes(node))
    ...
```

getAttrName

Purpose

Given a pointer to an attribute, returns the name of the attribute. Under the DOM spec, this is a method named `getName`.

Syntax

```
const oratext *getAttrName(const xmlnode *attr)
```

Parameters

attr	(IN)	pointer to attribute (see `getAttribute`)

Example

```
xmlnode *elem, *attr;
...
attr = setAttribute(ctx, elem, "x", "y");
getAttrName(attr) -> "x"
```

getAttrSpecified

Purpose

Return the 'specified' flag for the attribute: if this attribute was explicitly given a value in the original document or through the DOM, this is `TRUE`; otherwise, it is `FALSE`. If the node is not an attribute, returns `FALSE`. Under the DOM spec, this is a method named `getSpecified`.

Syntax

```
boolean getAttrSpecified(const xmlnode *attr)
```

Parameters

attr	(IN)	pointer to attribute (see `getAttribute`)

Example

```
xmlnode *elem, *attr;
...
attr = setAttribute(ctx, elem, "x", "y");
getAttrSpecified(attr) -> TRUE
```

getAttrValue

Purpose

Given a pointer to an attribute, returns the "value" (definition) of the attribute. Under the DOM spec, this is a method named `getValue`.

Syntax

```
const oratext *getAttrValue(const xmlnode *attr)
```

Parameters

attr	(IN)	pointer to attribute (see `getAttribute`)

Example

```
xmlnode *elem, *attr;
...
attr = setAttribute(ctx, elem, "x", "y");
getAttrValue(attr) -> "y"
```

getCharData

Purpose

Returns the character data of a TEXT or CDATA node. Under the DOM spec, this is a method named `getData`.

Syntax

```
const oratext *getCharData(const xmlnode *node)
```

Parameters

node	(IN)	pointer to text node

Example

```
xmlnode *node;
...
if (node = createTextNode(ctx, "riverrun"))
```

```
    getCharData(node) -> "riverrun"
```

getCharLength

Purpose

Returns the length of the character data of a TEXT or CDATA node. Under the DOM spec, this is a method named `getLength`.

Syntax

```
ub4 getCharLength(const xmlnode *node)
```

Parameters

node	(IN)	pointer to text node

Example

```
xmlnode *node;
...
if (node = createTextNode(ctx, "pruptly"))
    getCharLength(node) -> 8
```

getChildNode

Purpose

Returns the *n*th node in an array of nodes, or `NULL` if the numbered node does not exist. Invented function, not in DOM, but named to match the DOM pattern.

Syntax

```
xmlnode* getChildNode(const xmlnodes *nodes, size_t index)
```

Parameters

nodes	(IN)	array of nodes (see `getChildNodes`)
index	(IN)	zero-based child#

Example

```
xmlnode  *node, *child;
xmlnodes *nodes;
...
if (nodes = getChildNodes(node))
{
    child = getChildNode(nodes, 1);/* second child node */
    ...
}
```

getChildNodes

Purpose

Returns the array of children of the given node. This pointer may then be passed to `getChildNode` to fetch individual children.

Syntax

```
xmlnodes* getChildNodes(const xmlnode *node)
```

Parameters

node	(IN)	node whose children to return

Example

```
xmlnode  *node;
xmlnodes *nodes;
...
if (nodes = getChildNodes(node))
    ...
```

getContentModel

Purpose

Returns the content model for the named element from the current DTD. The content model is composed of xmlnodes, so may be traversed with the same functions as the parsed document. See also the `getModifier` function which returns the '?', '*', and '+' modifiers to content model nodes.

Syntax

```
xmlnode *LpxGetContentModel(xmldtd *dtd, oratext *name)
```

Parameters

dtd	(IN)	pointer to the DTD
name	(IN)	name of element

getDocType

Purpose

Returns a pointer to the (opaque) DTD for the current document.

Syntax

```
xmldtd* getDocType(xmlctx *ctx)
```

Parameters

ctx	(IN)	XML parser context

Example

```
xmlnodes *nodes;
...
nodes = getDocTypeEntities(getDocType(ctx));
```

getDocTypeEntities

Purpose

Returns an array of (general) entities defined for the given DTD.

Syntax

```
xmlnodes *getDocTypeEntities(xmldtd* dtd)
```

Parameters

dtd	(IN)	pointer to DTD

Example

```
xmldtd   *dtd;
xmlnodes *entities;
...
dtd = getDocType(ctx);
entities = getDocTypeEntities(dtd);
```

getDocTypeName

Purpose

Returns the given DTD's name.

Syntax

```
oratext *getDocTypeName(xmldtd* dtd)
```

Parameters

dtd	(IN)	pointer to DTD

getDocTypeNotations

Purpose

Returns an array of notations defined for the given DTD.

Syntax

```
xmlnodes *getDocTypeNotations(xmldtd* dtd)
```

Parameters

dtd	(IN)	pointer to DTD

Example

```
xmldtd   *dtd;
xmlnodes *notations;
...
dtd = getDocType(ctx);
notations = getDocTypeNotations(dtd);
```

getElementsByTagName

Purpose

Returns a list of all elements (within the tree rooted at the given node) with a given tag name in the order in which they would be encountered in a pre-order traversal of the tree. If root is `NULL`, the entire document is searched. The special value "*" matches all tags.

Syntax

```
xmlnodes *getElementsByTagName(xmlctx *ctx, xmlnode *root, const oratext *name)
```

Parameters

ctx	(IN)	XML parser context
root	(IN)	root node of tree
name	(IN)	element tag name

Example

```
xmlnodes *nodes;
...
nodes = getElementsByTagName(ctx, NULL, "ACT");/* find all ACT elements */
```

getDocument

Purpose

Returns the root node of the parsed document. The root node is always of type `DOCUMENT_NODE`. Compare to the `getDocumentElement` function, which returns the root *element* node, which is a child of the DOCUMENT node.

Syntax

```
xmlnode* getDocument(xmlctx *ctx)
```

Parameters

ctx	(IN)	XML parser context

getDocumentElement

Purpose

Returns the root element (node) of the parsed document. The entire document is rooted at this node. Compare to `getDocument` which returns the uppermost DOCUMENT node (the parent of the root element node).

Syntax

```
xmlnode* getDocumentElement(xmlctx *ctx)
```

Parameters

ctx	(IN)	XML parser context

getEntityNotation

Purpose

Returns an entity node's NDATA (notation). Under the DOM spec, this is a method named `getNotationName`.

Syntax

```
const oratext *getEntityNotation(const xmlnode *ent)
```

Parameters

ent	(IN)	pointer to entity

Example

```
<!NOTATION n SYSTEM "http://www.w3.org/">
<!ENTITY e SYSTEM "http://www.w3.org/" NDATA n>

xmlnode *ent;/* assume ent will be set to ENTITY node above */
...
getEntityNotation(ent) -> "n"
```

getEntityPubID

Purpose

Returns an entity node's public ID. Under the DOM spec, this is a method named `getPublicId`.

Syntax

```
const oratext *getEntityPubID(const xmlnode *ent)
```

Parameters

ent	(IN)	pointer to entity

Example

```
<!ENTITY e PUBLIC "PublicID" "nop.ent">

xmlnode *ent;/* assume ent will be set to ENTITY node above */
...
getEntityPubID(ent) -> "PublicID"
```

getEntitySysID

Purpose

Returns an entity node's system ID. Under the DOM spec, this is a method named `getSystemId`.

Syntax

`const oratext *getEntitySysID(const xmlnode *ent)`

Parameters

ent	(IN)	pointer to entity

Example

```
<!ENTITY e PUBLIC "PublicID" "nop.ent">

xmlnode *ent;/* assume ent will be set to ENTITY node above */
...
getEntitySysID(ent) -> "nop.ent"
```

getFirstChild

Purpose

Returns the first child of the given node, or `NULL` if the node has no children.

Syntax

```
xmlnode* getFirstChild(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode *elem;/* assume elem will point to element Thing */
...
getFirstChild(elem) -> element "A"
```

getImplementation

Purpose

This function returns a pointer to the DOMImplementation structure for this implementation, or `NULL` if no such information is available.

Syntax

```
xmldomimp* getImplementation(xmlctx *ctx)
```

Parameters

ctx	(IN)	XML context

getLastChild

Purpose

Returns the last child of the given node, or `NULL` if the node has no children.

Syntax

```
xmlnode* getLastChild(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode *elem;/* assume elem will point to element Thing */
...
getLastChild(elem) -> element "C"
```

getModifier

Purpose

Returns the modifier for a content model node. Possible values are `XMLCPMOD_NONE` (no modifier), `XMLCPMOD_OPT` ('?', optional), `XMLCPMOD_0MORE` ('*', zero or more), or `XMLCPMOD_1MORE` (one or more).

Syntax

```
xmlcpmod getModifier(xmlnode *node)
```

Parameters

node	(IN)	pointer to content model node

getNamedItem

Purpose

Returns the named node from an array nodes; sets the user's index (if provided) to the child# of the node (first node is zero).

Syntax

```
xmlnode *getNamedItem(const xmlnodes *nodes, const oratext *name, size_t *index)
```

Parameters

nodes	(IN)	array of nodes
name	(IN)	name of node to fetch
index	(OUT)	index of found node

Example

```
xmlnode  *node, *elem;
xmlnodes *nodes;
size_t    index;
...
if (nodes = getChildNodes(elem))
{
    node = getNamedItem(nodes, "FOO", &index);
    ...
}
```

getNextSibling

Purpose

This function returns a pointer to the next sibling of the given node, that is, the next child of the parent. For the last child, `NULL` is returned.

Syntax

```
xmlnode* getNextSibling(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode *node, *elem;/* assume elem will point to node Thing */
...
```

```
for (node = getFirstChild(elem); node; node = getNextSibling(node))
    ...node will be A then B then C...
```

getNodeMapLength

Purpose

Given an array of nodes (as returned by `getChildNodes`), returns the number of nodes in the map. Under the DOM spec, this is a member function named `getLength`.

Syntax

```
size_t getNodeMapLength(const xmlnodes *nodes)
```

Parameters

nodes	(IN)	array of nodes

Example

```
<Thing><A/><B/><C/></Thing>

xmlnodes *nodes;
xmlnode  *elem;/* assume elem will point to node Thing */
...
if (nodes = getChildNodes(elem))
    getNodeMapLength(nodes) -> 3
```

getNodeName

Purpose

Returns the name of the given node, or `NULL` if the node has no name. Note that "tagname" and "name" are currently synonymous.

Syntax

```
const oratext* getNodeName(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode  *elem;/* assume elem will point to node Thing */
...
getNodeName(elem) -> "Thing"
```

getNodeType

Purpose
Returns the type code for a node.

Syntax
```
xmlntype getNodeType(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode  *elem;/* assume elem will point to node Thing */
...
getNodeType(elem) -> ELEMENT_NODE
```

getNodeValue

Purpose
Returns the "value" (associated character data) for a node, or `NULL` if the node has no data.

Syntax

```
const oratext* getNodeValue(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<!--This is a comment-->

xmlnode *node;/* assume node will point to comment node above */
...
getNodeValue(node) -> "This is a comment"
```

getNotationPubID

Purpose

Return a notation node's public ID. Under the DOM spec, this is a method named `getPublicId`.

Syntax

```
const oratext *getNotationPubID(const xmlnode *note)
```

Parameters

note	(IN)	pointer to node

Example

```
<!NOTATION n PUBLIC "whatever">

xmlnode *note;/* assume note will point to notation node above */
...
getNotationPubID(note) -> "whatever"
```

getNotationSysID

Purpose

Return a notation node's system ID. Under the DOM spec, this is a method named `getSystemId`.

Syntax

```
const oratext *getNotationSysID(const xmlnode *note)
```

Parameters

note	(IN)	pointer to node

Example

```
<!NOTATION n SYSTEM "http://www.w3.org/">

xmlnode *note;/* assume note will point to notation node above */
...
getNotationSysID(note) -> "http://www.w3.org/"
```

getOwnerDocument

Purpose

Returns the document node which contains the given node. An XML document is always rooted in a node of type DOCUMENT_NODE. Calling `getOwnerDocument` on any node in the document returns that document node.

Syntax

```
xmlnode* getOwnerDocument(xmlnode *node)
```

Parameters

node	(IN)	pointer to node

getParentNode

Purpose

Returns the parent node of the given node. For the top-most node, `NULL` is returned.

Syntax

```
xmlnode* getParentNode(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode  *elem;/* assume elem will point to node A */
...
getParentNode(elem) -> node Thing
```

getPIData

Purpose

Returns a Processing Instruction's (PI) data string. Under the DOM spec, this is a method named `getData`.

Syntax

```
const oratext *getPIData(const xmlnode *pi)
```

Parameters

pi	(IN)	pointer to PI node

Example

```
<?PI Blither blather?>
```

```
xmlnode *pi;/* assume pi will point to PI node above */
...
getPIData(pi) -> "Blither blather"
```

getPITarget

Purpose

Returns a Processing Instruction's (PI) target string. Under the DOM spec, this is a method named `getTarget`.

Syntax

```
const oratext *getPITarget(const xmlnode *pi)
```

Parameters

pi	(IN)	pointer to PI node

Example

```
<?PI Blither blather?>

xmlnode *pi;/* assume pi will point to PI node above */
...
getPITarget(pi) -> "PI"
```

getPreviousSibling

Purpose

Returns the previous sibling of the given node. That is, the node at the same level which came before this one. For the first child of a node, `NULL` is returned.

Syntax

```
xmlnode* getPreviousSibling(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode *node, *elem;/* assume elem will point to node Thing */
...
for (node = getLastChild(elem); node; node = getPreviousSibling(node))
    ...node will be C then B then A...
```

getTagName

Purpose

Returns the "tagname" of a node, which is the same as its name for now, see `getNodeName`. The DOM says "...even though there is a generic nodeName attribute on the Node interface, there is still a tagName attribute on the Element interface; these two attributes must contain the same value, but the Working Group considers it worthwhile to support both, given the different constituencies the DOM API must satisfy.

Syntax

```
const oratext *getTagName(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

hasAttributes

Purpose

Determines if if the given node has any defined attributes, returning `TRUE` if so, `FALSE` if not. This is a DOM extension named after the pattern started by `hasChildNodes`.

Syntax

```
boolean hasAttributes(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

hasChildNodes

Purpose

Determines if the given node has children, returning `TRUE` if so, `FALSE` if not. The same result can be achieved by testing if `getChildNodes` returns a pointer (has children) or `NULL` (no children).

Syntax

```
boolean hasChildNodes(const xmlnode *node)
```

Parameters

node	(IN)	pointer to node

hasFeature

Purpose

Tests if the DOM implementation implements a specific feature and version. *feature* is the package name of the feature to test. In DOM Level 1, the legal values are "HTML" and "XML" (case-insensitive). *version* is the version number of the package name to test. In DOM Level 1, this is the string "1.0". If the version is not specified, supporting any version of the feature will cause the method to return `TRUE`.

Syntax

```
boolean hasFeature(xmlctx *ctx, const oratext *feature, const oratext *version)
```

Parameters

ctx	(IN)	XML context
feature	(IN)	the package name of the feature to test
version	(IN)	the version number of the package name to test

insertBefore

Purpose

Inserts a new node into the given parent node's list of children before the existing reference node. If the reference node is `NULL`, appends the new node at the end of the list. If the new node is a DocumentFragment, its children are inserted, in the same order, instead of the fragment itself. If the new node is already in the tree, it is first removed.

Syntax

```
xmlnode *insertBefore(xmlctx *ctx, xmlnode *parent, xmlnode *newChild, xmlnode
*refChild)
```

Parameters

ctx	(IN)	XML context
parent	(IN)	parent node to insert into
newChild	(IN)	new child node to insert
refChild	(IN)	reference node to insert before

Example

```
<Thing><A/><B/><C/></Thing>

xmlnode *elem, *new, *ref;    /* assume elem points to Thing, new is a new
                                 element "Z", and ref points to node B */
...
insertBefore(ctx, elem, new, ref);

<Thing><A/><Z/><B/><C/></Thing>
```

insertData

Purpose

Inserts a string into the node character data at the specified offset.

Syntax

`void insertData(xmlctx *ctx, xmlnode *node, ub4 offset, const oratext *arg)`

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node
offset	(IN)	insertion point (0 is first position)
refChild	(IN)	new string to insert

Example

```
xmlnode *node;
...
getNodeValue(node) -> "abcdefg"
insertData(ctx, node, 3, "ZZZ");
getNodeValue(node) -> "abcZZZdefg"
```

isStandalone

Purpose

Returns the value of the `standalone` flag as specified in the document's <?xml?> processing instruction. This is an invented function, not in DOM spec, but named to match the DOM pattern.

Syntax

`boolean isStandalone(xmlctx *ctx)`

Parameters

ctx	(IN)	XML parser context

nodeValid

Purpose

Validate a node against the DTD. Returns 0 on success, else a non-zero error code (which can be looked up in the message file). This function is provided for applications which construct their own documents via the API and/or Class Generator. Normally the parser will validate the document and the user need not call `nodeValid` explicitly.

Syntax

```
uword nodeValid(xmlctx *ctx, const xmlnode *node)
```

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node

normalize

Purpose

"*Normalizes*" an element, i.e. merges adjacent TEXT nodes. Adjacent TEXT nodes don't happen during a normal parse, only when extra nodes are inserted via the DOM.

Syntax

```
void normalize(xmlctx *ctx, xmlnode *elem)
```

Parameters

ctx	(IN)	XML context
elem	(IN)	pointer to element node

Example

```
xmlnode *node, *t1, *t2;
...
if ((node = createElement(ctx, "FOO")) &&
    (t1 = createTextNode(ctx, "one of ")) &&
    (t2 = createTextNode(ctx, "these days")) &&
    appendChild(ctx, node, t1) &&
    appendChild(ctx, node, t2))
{
    <FOO>"one of " "these days"</FOO>
    normalize(ctx, node);
    <FOO>"one of these days"</FOO>
}
```

numAttributes

Purpose

Returns the number of defined attributes in an attribute array (as returned by `getAttributes`). This is an invented function, not in the DOM spec, but named after the DOM pattern.

Syntax

```
size_t numAttributes(const xmlnodes *attrs)
```

Parameters

attrs	(IN)	array of attributes

Example

```
xmlnodes *nodes;
xmlnode  *node;
size_t    i;
...
if (nodes = getAttributes(node))
{
    for (i = 0; i < numAttributes(nodes); i++)
        ...
}
```

numChildNodes

Purpose

Returns the number of children in an array of nodes (as returned by `getChildNodes`). This is an invented function, not in the DOM spec, but named after the DOM pattern.

Syntax

```
size_t numChildNodes(const xmlnodes *nodes)
```

Parameters

nodes	(IN)	pointer to opaque nodes structure

Example

```
xmlnodes *nodes;
xmlnode  *elem;
size_t    i;
...
if (nodes = getChildNodes(elem))
{
    for (i = 0; i < numChildNodes(nodes); i++)
        ...
}
```

removeAttribute

Purpose

Removes the named attribute from an element node. If the removed attribute has a default value it is immediately replaced.

Syntax

```
void removeAttribute(xmlnode *elem, const oratext *name)
```

Parameters

elem	(IN)	pointer to element node
name	(IN)	name of attribute to remove

Example

```
<!ATTLIST FOO attr CDATA 'default'>
```

```
xmlnode *elem;/* assume elem point to a FOO node */
...
<FOO attr="snark"/>
removeAttribute(elem, "attr");
<FOO attr="default"/>
```

removeAttributeNode

Purpose

Removes an attribute from an element, given a pointer to the attribute. If successful, returns the attribute node back. On error, returns `NULL`.

Syntax

```
xmlnode *removeAttributeNode(xmlnode *elem, xmlnode *attr)
```

Parameters

elem	(IN)	pointer to element node
attr	(IN)	attribute node to remove

Example

```
xmlnode *elem, *attr;
...
if (attr = getAttributeNode(elem, "attr1"))
    removeAttributeNode(elem, attr);
```

removeChild

Purpose

Removes the given node from its parent and returns it.

Syntax

```
xmlnode *removeChild(xmlnode *node)
```

Parameters

node	(IN)	old node to remove

Example

```
xmlnodes *nodes;
xmlnode  *elem, *node;
...
if ((nodes = getChildNodes(elem)) &&
    (node = getNamedItem(nodes, "B", NULL))
{
    <Thing><A/><B/><C/></Thing>
    removeChild(node);
    <Thing><A/><C/></Thing>
}
```

removeNamedItem

Purpose

Removes the named node from an array of nodes.

Syntax

```
xmlnode *removeNamedItem(xmlnodes *nodes, const oratext *name)
```

Parameters

nodes	(IN)	list of nodes
name	(IN)	name of node to remove

Example

```
xmlnodes *nodes;
xmlnode  *elem;
...
if (nodes = getChildNodes(elem))
{
    <Thing><A/><B/><C/></Thing>
    removeNamedItem(nodes, "B");
    <Thing><A/><C/></Thing>
}
```

replaceChild

Purpose

Replaces an existing child node with a new node and returns the old node. If the new node is already in the tree, it is first removed.

Syntax

```
xmlnode *replaceChild(xmlctx *ctx, xmlnode *newChild, xmlnode *oldChild)
```

Parameters

ctx	(IN)	XML context
newChild	(IN)	new replacement node
oldChild	(IN)	old node being replaced

Example

```
xmlnodes *nodes;
xmlnode  *elem, *old, *new;
...
if ((nodes = getChildNodes(elem)) &&
    (old = getNamedItem(nodes, "B", NULL)) &&
    (new = createElement(ctx, "NEW")))
{
    <Thing><A/><B/><C/></Thing>
    replaceChild(ctx, new, old);
    <Thing><A/><NEW/><C/></Thing>
}
```

replaceData

Purpose

Replaces the substring at the given character offset and length with a replacement string.

Syntax

```
void replaceData(xmlctx *ctx, xmlnode *node, ub4 offset, ub4 count, oratext
*arg)
```

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node
offset	(IN)	start of substring to replace (0 is first character)
count	(IN)	length of old substring
arg	(IN)	replacement text

Example

```
xmlnode *node;
...
getNodeValue(node) -> "every dog has his day"
replaceData(ctx, node, 6, 3, "man");
getNodeValue(node) -> "every man has his day"
```

setAttribute

Purpose

Create a new attribute for an element. If the named attribute already exists, its value is simply replaced.

Syntax

```
xmlnode *setAttribute(xmlctx *ctx, xmlnode *elem, const oratext *name, const
oratext *value)
```

Parameters

ctx	(IN)	XML context
elem	(IN)	pointer to element node
name	(IN)	name of new attribute

value	(IN)	value of new attribute

Example

```
xmlnode *elem;
...
<Thing/>
setAttribute(ctx, elem, "attr", "value");
<Thing attr="value"/>
```

setAttributeNode

Purpose

Adds a new attribute to the given element. If the named attribute already exists, it is replaced and the user's old pointer (if provided) is set to the old attr. If the attribute is new, it is added and the old pointer is set to `NULL`. Returns a truth value indicating success.

Syntax

```
boolean setAttributeNode(xmlctx *ctx, xmlnode *elem,
                         xmlnode *newNode, xmlnode **oldNode)
```

Parameters

ctx	(IN)	XML context
elem	(IN)	pointer to element node
newNode	(IN)	pointer to new attribute
oldNode	(OUT)	return pointer for old attribute

Example

```
xmlnode *elem, *attr;
...
if (attr = createAttribute(ctx, "attr", "value"))
{
    <Thing/>
    setAttributeNode(ctx, elem, attr, NULL);
    <Thing attr="value"/>
}
```

setNamedItem

Purpose

Sets a new child node in a parent node's map; if an old node exists with same name, replaces the old node (and sets user's pointer, if provided, to it); if no such named node exists, appends node to map and sets pointer to NULL.

Syntax

```
boolean setNamedItem(xmlctx *ctx, xmlnode *parent, xmlnode *node, xmlnode **old)
```

Parameters

node	(IN)	pointer to node
parent	(IN)	parent to add node to
node	(IN)	new node to add
old	(IN)	pointer to replaced node

Example

```
xmlnode *elem, *new;
...
if ((new = createElement(ctx, "B")) &&
    setAttribute(ctx, new, "attr", "value"))
{
    <Thing><A/><B/><C/></Thing>
    setNamedItem(ctx, elem, new, NULL);
    <Thing><A/><B attr="value"/><C/></Thing>
}
```

setNodeValue

Purpose

Sets the *value* (character data) associated with a node.

Syntax

```
boolean setNodeValue(xmlnode *node, const oratext *data)
```

Parameters

node	(IN)	pointer to node
data	(IN)	new data for node

Example

```
xmlnode *node;
...
getNodeValue(node) -> "umbrella"
setNodeValue(node, "brolly");
getNodeValue(node) -> "brolly"
```

setPIData

Purpose

Sets a Processing Instruction's (PI) data (equivalent to `setNodeValue`). It is not permitted to set the data to `NULL`. Under the DOM spec, this is a method named `setData`.

Syntax

```
void setPIData(xmlnode *pi, const oratext *data)
```

Parameters

pi	(IN)	pointer to PI node
data	(IN)	new data for PI

Example

```
xmlnode *pi;
...
<?SKRINKLIT Monster Grendel's tastes are plainish?>
setPIData(pi, "Breakfast?  Just a couple Danish.");
```

```
<?SKRINKLIT Breakfast?  Just a couple Danish.?>
```

splitText

Purpose

Breaks a TEXT node into two TEXT nodes at the specified offset, keeping both in the tree as siblings. The original node then only contains all the content up to the offset point. And a new node, which is inserted as the next sibling of the original, contains all the old content starting at the offset point.

Syntax

```
xmlnode *splitText(xmlctx *ctx, xmlnode *old, uword offset)
```

Parameters

ctx	(IN)	XML context
old	(IN)	original node to split
offset	(IN)	offset of split point

Example

```
xmlnode *node;
...
<FOO>"one of these days"</FOO>
splitText(ctx, node, 7);
<FOO>"one of " "these days"</FOO>
```

substringData

Purpose

Returns a substring of a node's character data.

Syntax

```
const oratext *substringData(xmlctx *ctx, const xmlnode *node, ub4 offset, ub4
count)
```

Parameters

ctx	(IN)	XML context
node	(IN)	pointer to node
offset	(IN)	offset of start of substring
count	(IN)	length of substring

Example

```
xmlnode *node;
...
<FOO>"one of these days"</FOO>
substringData(ctx, node, 0, 3) -> "one"
```

Namespace APIs

Namespace APIs provide an interface that is an extension to the DOM and give information relating to the document namespaces.

XML namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URI references. A single XML document may contain elements and attributes (here referred to as a "markup vocabulary") that are defined for and used by multiple software modules. One motivation for this is modularity; if such a markup vocabulary exists which is well-understood and for which there is useful software available, it is better to re-use this markup rather than re-invent it.

Such documents, containing multiple markup vocabularies, pose problems of recognition and collision. Software modules need to be able to recognize the tags and attributes which they are designed to process, even in the face of "collisions" occurring when markup intended for some other software package uses the same element type or attribute name.

These considerations require that document constructs should have universal names, whose scope extends beyond their containing document. This C implementation of XML namespaces provides a mechanism to accomplish this.

Names from XML namespaces may appear as qualified names, which contain a single colon, separating the name into a namespace prefix and a local part. The prefix, which is mapped to a URI reference, selects a namespace. The combination of the universally managed URI namespace and the document's own namespace produces identifiers that are universally unique. Mechanisms are provided for prefix scoping and defaulting.

URI references can contain characters not allowed in names, so cannot be used directly as namespace prefixes. Therefore, the namespace prefix serves as a proxy for a URI reference. An attribute-based syntax described in the W3C Namespace specification is used to declare the association of the namespace prefix with a URI reference.

The implementation of this C Namespace interface followed the XML Namespace standard of revision REC-xml-names-19990114.

Data Structures and Types

```
oratext
xmlattr
xmlnode
```

Functions

`getAttrLocal(xmlattr *attrs)`

Returns attribute local name.

`getAttrNamespace(xmlattr *attr)`

Returns attribute namespace (URI).

`getAttrPrefix(xmlattr *attr)`

Returns attribute prefix.

`getAttrQualifiedName(xmlattr *attr)`

Returns attribute fully qualified name.

`getNodeLocal(xmlnode *node)`

Returns node local name.

`getNodeNamespace(xmlnode *node)`

Returns node namespace (URI).

`getNodePrefix(xmlnode *node)`

Returns node prefix.

`getNodeQualifiedName(xmlnode *node)`

Returns node qualified name.

Data Structure and Type Description

```
ORATEXT
typedef unsigned char oratext;

XMLATTR

typedef struct xmlattr xmlattr;
```

Note: the contents of xmlattr are private and must not be accessed by users.

XMLNODE

```
typedef struct xmlnode xmlnode;
```

Note: the contents of xmlnode are private and must not be accessed by users.

Function Prototypes

getAttrLocal

Purpose
This function returns the local name of this attribute.

Syntax
```
const oratext *getAttrLocal(const xmlattr *attr);
```

Parameters
attr (IN) - pointer to opaque attribute structure (see getAttribute)

Comments

getAttrNamespace

Purpose
This function returns namespace for this attribute.

Syntax
```
const oratext *getAttrNamespace(const xmlattr *attr);
```

Parameters
attr (IN) - pointer to opaque attribute structure (see getAttribute)

Comments

getAttrPrefix

Purpose
This function returns prefix for this attribute.

Syntax

```
const oratext *getAttrPrefix(const xmlattr *attr);
```

Parameters

attr (IN) - pointer to opaque attribute structure (see getAttribute)

Comments

getAttrQualifiedName

Purpose

This function returns fully qualified name for the attribute.

Syntax

```
const oratext *getAttrQualifiedName(const xmlattr *attr);
```

Parameters

attr (IN) - pointer to opaque attribute structure (see getAttribute)

Comments

getNodeLocal

Purpose

This function returns the local name of this node.

Syntax

```
const oratext *getNodeLocal(const xmlnode *node);
```

Parameters

node (IN) - node to get local name from

Comments

getNodeNamespace

Purpose
This function returns namespace for this node.

Syntax
```
const oratext *getNodeNamespace(const xmlnode *node);
```

Parameters
node (IN) - node to get namespace from

Comments

getNodePrefix

Purpose
This function returns prefix for this node.

Syntax
```
const oratext *getNodePrefix(const xmlnode *node);
```

Parameters
node (IN) - node to get prefix from

Comments

getNodeQualifiedName

Purpose
This function returns fully qualified name for this node.

Syntax
```
const oratext *getNodeQualifiedName(const xmlnode *node);
```

Parameters

node (IN) - node to get name from

Comments

Part III

XDK for C++ Packages

This section contains the following chapters:

- Chapter 7, "XML Parser for C++"
- Chapter 8, "Oracle XML Class Generator 1.0.0.0.0 (C++)"

7 XML Parser for C++

This chapter describes the following sections:

- Class APIs
- Parser APIs
- C++ Sax APIs
- C++ DOM APIs

Class: Attr

This class contains methods for accessing the name and value of a single document node attribute.

getName	Return name of attribute
getValue	Return "value" (definition) of attribute
getSpecified	Return attribute's "specified" flag value
setValue	Set an attribute's value

getName

Function
Return name of attribute

Prototype
`String getName()`

Arguments
None

Returns
`String` -- Name of attribute

getValue

Function
Return "value" (definition) of attribute

Prototype
`String getValue()`

Arguments

None

Returns

Value of attribute

getSpecified

Function

Return value of attribute's "specified" flag. The DOM says:

If this attribute was explicitly given a value in the original document, this is true; otherwise, it is false. Note that the implementation is in charge of this attribute, not the user. If the user changes the value of the attribute (even if it ends up having the same value as the default value) then the specified flag is automatically flipped to true. To re-specify the attribute as the default value from the DTD, the user must delete the attribute. The implementation will then make a new attribute available with specified set to false and the default value (if one exists).

Prototype

```
boolean getSpecified()
```

Arguments

None

Returns

Value of specified flag

setValue

Function

Sets an attribute's "value"

Prototype

```
void setValue(String value)
```

Arguments

`value -- Attribute's new value`

Returns

Value of attribute

Class: CDATASection

This class implements the CDATA node type, a subclass of **Text**.

Class: Comment

This class implements the COMMENT node type, a subclass of **CharacterData**.

Class: Document

This class contains methods for creating and retrieving nodes.

createAttribute	Create an ATTRIBUTE node
createCDATASection	Create a CDATA node
createComment	Create a COMMENT node
createDocumentFragment	Create a DOCUMENT_FRAGMENT node
createElement	Create an ELEMENT node
createEntityReference	Create an ENTITY_REFERENCE node
createProcessingInstruction	Create a PROCESSING_INSTRUCTION node
createTextNode	Create a TEXT node
getElementsByTagName	Select nodes based on tag name
getImplementation	Return DTD for document

createAttribute

Function
Create a new attribute node. Use setValue to set its value.

Prototype
Attr* createAttribute(String name)

Arguments
`name` -- name of attribute

Returns
`Attr*` -- pointer to created node

createCDATASection

Function
Create a new CDATA node with the given contents.

Prototype
`Attr* createCDATASection(String name)`

Arguments
`data` -- contents of node

Returns
`CDATASection*` -- pointer to created node

createComment

Function
Create a new comment node with the given contents.

Prototype
`Comment* createComment(String data)`

Arguments
`data` -- contents of node

Returns
`Comment*` -- pointer to created node

createDocumentFragment

Function
Create a new document fragment node.

Prototype
`DocumentFragment* createDocumentFragment()`

Arguments
None

Returns
`DocumentFragment*` -- pointer to created node

createElement

Function
Create a new element node with the given (tag) name.

Prototype
`Element* createElement(String tagName)`

Arguments
`tagName` -- element's tagname

Returns
`Element*` -- pointer to created node

createEntityReference

Function
Create a new entity reference node.

Prototype
`EntityReference* createEntityReference(String name)`

Arguments
`name` -- name of entity to reference

Returns
`EntityReference*` -- pointer to created node

createProcessingInstruction

Function
Create a new processing instruction node.

Prototype
```
ProcessingInstruction* createProcessingInstruction(String target, String data)
```

Arguments
`target` -- target part of PI

`data` -- data for node

Returns
`ProcessingInstruction*` -- pointer to created node

createTextNode

Function
Create a new TEXT node.

Prototype
```
Text* createTextNode(String data)
```

Arguments
`data` -- data for node

Returns
`Text*` -- pointer to created node

getElementsByTagName

Function
Returns a NodeList of all the Elements with a given tag name in the order in which they would be encountered in a preorder traversal of the Document tree. The special value "*" matches all tags.

Prototype

`NodeList* getElementsByTagName(String tagname)`

Arguments

`tagname` -- tag name to select

Returns

`NodeList*` -- list of matches, NULL if none

getImplementation

Function

Returns the DOMImplementation structure, currently useless. Perhaps it will be used in later DOM versions.

Prototype

`DOMImplementation* getImplementation()`

Arguments

None

Returns

`DOMImplementation*` -- pointer to structure

Class: DocumentType

This class contains methods for accessing information about the Document Type Definition (DTD) of a document.

getName	Return name of DTD
getEntities	Return NamedNodeMap of DTD's (general) entities
getNotations	**Return NamedNodeMap of DTD's notations**

getName

Function
Return name of DTD

Prototype
`String getName()`

Arguments
None

Returns
`String` -- Name of DTD

getEntities

Function
Returns map of DTD's (general) entities

Prototype
`NamedNodeMap* getEntities()`

Arguments
None

Returns

`NamedNodeMap*` -- map of entities

getNotations

Function

Return map of DTD's notations

Prototype

```
NamedNodeMap* getNotations()
```

Arguments

None

Returns

`NamedNodeMap*` -- map of notations

Class: DOMImplementation

This class contains methods relating to the specific DOM implementation supported by the parser.

hasFeature	Detect if the named feature is supported

hasFeature

Function

Test if the DOM implementation implements a specific feature.

Prototype

`boolean hasFeature(DOMString feature, DOMString version)`

Arguments

`feature` -- The package name of the feature to test. In Level 1, the legal values are "HTML" and "XML" (case-insensitive)

`version` -- This is the version number of the package name to test. In Level 1, this is the string "1.0". If the version is not specified, supporting any version of the feature will cause the method to return true.

Returns

`boolean` -- feature is supported

Class: Element

This class contains methods pertaining to element nodes.

getTagName	Return the node’s tag name
getAttribute	Select an attribute given its name
setAttribute	Create a new attribute given its name and value
removeAttribute	Remove an attribute given its name
getAttributeNode	Remove an attribute given its name
setAttributeNode	Add a new attribute node
removeAttributeNode	Remove an attribute node
getElementsByTagName	Return a list of element nodes with the given tag name
normalize	"Normalize" an element (merge adjacent text nodes)

getTagName

Function

Return the tag name of the element. The DOM says: "...even though there is a generic nodeName attribute on the Node interface, there is still a tagName attribute on the Element interface; these two attributes must contain the same value, but the Working Group considers it worthwhile to support both, given the different constituencies the DOM API must satisfy

Prototype

```
String getTagName()
```

Arguments

None

Returns

`String` -- Tag name of node

getAttribute

Function
Return "value" (definition) of named attribute

Prototype
`String getAttribute(String name)`

Arguments
`name` -- name of attribute

Returns
Value of attribute

setAttribute

Function
Create a new attribute

Prototype
`Attr* setAttribute(String name, String value)`

Arguments
`name` -- name of new attribute

`value` -- value of new attribute

Returns
Pointer to create attribute

removeAttribute

Function
Removes the named attribute

Prototype
`void removeAttribute(String name)`

Arguments
`name` -- name of attribute to remove

Returns
None

getAttributeNode

Function
Return pointer to named attribute

Prototype
```
Attr* getAttributeNode(DOMString name)
```

Arguments
`name` -- name of attribute

Returns
`Attr*` -- pointer to attribute, or NULL if none such

setAttributeNode

Function
Set (add) new attribute

Prototype
```
boolean setAttributeNode(Attr* newAttr, Attr** oldAttr)
```

Arguments
`newAttr` -- pointer to new attribute

`oldAttr` -- returned pointer to replaced attribute

Returns
`boolean` -- success

removeAttributeNode

Function
Remove the named attribute

Prototype
`Attr* removeAttributeNode(Attr* oldAttr)`

Arguments
`oldAttr` -- attribute to remove

Returns
`Attr*` -- oldAttr passed back

getElementsByTagName

Function
Create a list of matching elements

Prototype
`NodeList* getElementsByTagName(DOMString name)`

Arguments
`name` -- tagname to match, "*" for all

Returns
`NodeList*` -- list of matches

normalize

Function
Normalize an element, i.e. merge all adjacent TEXT nodes

Prototype
`void normalize(void)`

Arguments

None

Returns

None

Class: Entity

This class implements the ENTITY node type, a subclass of **Node**.

getNotationName	Return entity's NDATA (notation name)
getPublicId	Return entity's public ID
getSystemId	**Return entity's system ID**

getNotationName

Function
Return an entity node's notation name (NDATA)

Prototype
`String* getNotationName()`

Arguments
None

Returns
`String` -- node's NDATA

getPublicId

Function
Return an entity node's public ID

Prototype
`String getPublicId()`

Arguments
None

Returns
`String` -- entity's public ID

getSystemId

Function
Return an entity node's system ID

Prototype
```
String getSystemId()
```

Arguments
None

Returns
`String` -- entity's system ID

Class: EntityReference

This class implements the ENTITY_REFERENCE node type, a subclass of **Node**.

Class: NamedNodeMap

This class contains methods for accessing the number of nodes in a node map and fetching individual nodes.

item	Return *n*th node in map.
getLength	Return number of nodes in map
getNamedItem	Select a node by name
setNamedItem	Set a node into the map
removeNamedItem	Remove the named node from map

item

Function

Return *n*th node in node map

Prototype

```
Node* item(size_t index)
```

Arguments

`size_t index`-- zero-based node number

Returns

`Node*` -- Pointer to `index`'th node in map

getLength

Function

Return number of nodes in map

Prototype

```
size_t getLength()
```

Arguments
None

Returns
`size_t` -- Number of nodes in map

getNamedItem

Function
Selects the node with the given name from the map

Prototype
```
Node* getNamedItem(String name)
```

Arguments
`name` -- Name of node to select

Returns
`Node*` -- Pointer to named node, NULL of none such exists

setNamedItem

Function
Adds a node to the map, replacing any node that already exists with the same name

Prototype
```
boolean setNamedItem(Node *node, Node **old)
```

Arguments
`node` -- Name of node to add

`old` -- Pointer to replaced node, NULL if node is new

Returns
`boolean` -- Success

removeNamedItem

Function
Removes the node with the given name from the node map

Prototype
`Node* removeNamedItem(String name)`

Arguments
`name` -- Name of node to remove

Returns
`Node*` -- Pointer to removed node, NULL if none such exists

Class: Node

This class contains methods for details about a document node

appendChild	Append a new child to the end of the current node's list of children
cloneNode	Clone an existing node and optionally all its children
getAttributes	Return structure contains all defined node attributes
getChildNode	Return specific indexed child of given node
getChildNodes	Return structure contains all child nodes of given node
getFirstChild	Return first child of given node
getLastChild	Return last child of given node
getLocal	Returns the local name of the node
getNameSpace	Return a node's namespace
getNextSibling	Return a node's next sibling
getName	Return name of node
getType	Return numeric type-code of node
getValue	Return "value" (data) of node
getOwnerDocument	Return document node which contains a node
getParentNode	Return parent node of given node
getPrefix	Returns the namespace prefix for the node
getPreviousSibling	Returns the previous sibling of the current node
getQualifiedName	Return namespace qualified node of given node
hasAttributes	Determine if node has any defined attributes
hasChildNodes	Determine if node has children
insertBefore	Insert new child node into a node's list of children
numChildNodes	Return count of number of child nodes of given node
removeChild	Remove a node from the current node's list of children

replaceChild	Replace a child node with another
setValue	Sets a node's value (data)

appendChild

Function
Append a new child to the current node's list of children

Prototype
```
Node* appendChild(Node *newChild)
```

Arguments
`newChild` -- new child node

Returns
`Node*` -- newChild is passed back

cloneNode

Function
Returns a duplicate of this node, i.e., serves as a generic copy constructor for nodes. The duplicate node has no parent (parentNode returns NULL).

Cloning an Element copies all attributes and their values, including those generated by the XML processor to represent defaulted attributes, but this method does not copy any text it contains unless it is a deep clone, since the text is contained in a child Text node. Cloning any other type of node simply returns a copy of this node.

Prototype
```
Node* cloneNode(boolean deep)
```

Arguments
`deep` -- recursion flag

Returns
`Node*` -- Pointer to new clone

getAttributes

Function
Return structure of all attributes for node

Prototype
`NamedNodeMap* getAttributes()`

Arguments
None

Returns
`NamedNodeMap*` -- Pointer to structure describing all attributes for node, or `NULL` if no attributes are defined

getChildNode

Function
Return one of the node's children

Prototype
`Node* getChildNode(uword index)`

Arguments
`index` -- child number, starting at 0

Returns
`Node*` -- Pointer to `index`'th child of node

getChildNodes

Function
Return node's children

Prototype
`NodeList* getChildNodes()`

Arguments
None

Returns
`NodeList*` -- Pointer to structure describing all the node's children

getFirstChild

Function
Return the node's first child

Prototype
```
Node* getFirstChild()
```

Arguments
None

Returns
`Node*` -- Pointer to the node's first child

getLastChild

Function
Return the node's last child

Prototype
```
Node* getLastChild()
```

Arguments
None

Returns
`Node*` -- Pointer to the node's last child

getLocal

Function
Return the node's local name

Prototype
`String getLocal()`

Arguments
None

Returns
`String` -- node's local name

getNamespace

Function
Return the node's namespace

Prototype
`String getNamespace()`

Arguments
None

Returns
`String` -- node's namespace (may be NULL)

getNextSibling

Function
Returns the next sibling of the node, that is, the child of its parent which comes next

Prototype
`Node* getNextSibling()`

Arguments
None

Returns
`Node*` -- Node's next sibling, NULL if last child

getName

Function
Return name of node, or `NULL` if the node has no name

Prototype
`String getName()`

Arguments
None

Returns
`String` -- Node's name

getType

Function
Return numeric type-code for node

Prototype
`short getType()`

Arguments
None

Returns
`short` -- Node's numeric type code

Type codes:
`ELEMENT_NODE`

```
ATTRIBUTE_NODE
TEXT_NODE
CDATA_SECTION_NODE
ENTITY_REFERENCE_NODE
ENTITY_NODE
PROCESSING_INSTRUCTION_NODE
COMMENT_NODE
DOCUMENT_NODE
DOCUMENT_TYPE_NODE
DOCUMENT_FRAGMENT_NODE
NOTATION_NODE
```

getValue

Function
Return "value" (data) of node, or `NULL` if the node has no value

Prototype
`String getValue()`

Arguments
None

Returns
`String` -- Node's "value"

getOwnerDocument

Function
Return document node which contains the current node

Prototype
`Document* getOwnerDocument()`

Arguments
None

Returns
`Document*` -- Pointer to document node

getParentNode

Function
Return node's parent

Prototype
```
Node* getParentNode()
```

Arguments
None

Returns
`Node*` -- Pointer to the node's parent node

getPrefix

Function
Return the namespace prefix of node

Prototype
```
String getPrefix()
```

Arguments
None

Returns
`String` -- Node's namespace prefix, may be NULL

getPreviousSibling

Function
Returns the previous sibling of the node, that is, the child of its parent which came before

Prototype

`Node* getPreviousSibling()`

Arguments

None

Returns

`Node*` -- Node's previous sibling, NULL if first child

getQualifiedName

Function

Return the fully qualified (namespace) name of node

Prototype

`String getQualifiedName()`

Arguments

None

Returns

`String` -- Node's qualified name

hasAttributes

Function

Determine if node has any defined attributes

Prototype

`boolean hasAttributes()`

Arguments

None

Returns

`boolean` -- TRUE if node has attributes

hasChildNodes

Function

Determine if node has any children

Prototype

```
boolean hasChildNodes()
```

Arguments

None

Returns

`boolean` -- TRUE if node has child nodes

insertBefore

Function

Insert a new child node into the list of children of a parent, before the reference node. If refChild is NULL, appends the new node to the end.

Prototype

```
Node* insertBefore(Node *newChild, Node *refChild)
```

Arguments

`newChild` -- new node to insert

`refChild` -- reference node; new node comes before

Returns

`Node*` -- newChild passed back

numChildNodes

Function

Return count of node's children

Prototype

`uword numChildNodes()`

Arguments

None

Returns

`uword` -- Number of children this node has (might be 0)

removeChild

Function

Remove a child node from the current node's list of children

Prototype

`Node* removeChild(Node *oldChild)`

Arguments

`oldChild` -- old node being removed

Returns

`Node*` -- oldChild is passed back

replaceChild

Function

Replace one node with another. newChild replaces oldChild in the list of children in oldChild's parent.

Prototype

`Node* replaceChild(Node *newChild, Node *oldChild)`

Arguments

`newChild` -- new replacement node

`oldChild` -- old node being replaced

Returns
`Node*` -- oldChild is passed back

setValue

Function
Sets a node's "value" (data)

Prototype
```
void setValue(String data)
```

Arguments
`data` -- New data for node

Returns
`void`

Class: NodeList

This class contains methods for extracting nodes from a NodeList

item	Return *n*th node in list
getLength	Return number of nodes in list

item

Function
Return *n*th node in node list

Prototype
`Node* item(size_t index)`

Arguments
`size_t index`-- zero-based node number

Returns
Pointer to `index`'th node in list

getLength

Function
Return number of nodes in list

Prototype
`size_t getLength()`

Arguments
None

Returns
Number of nodes in list

Class: Notation

This class implements the NOTATION node type, a subclass of **Node**.

getData	Return notation’s data
getTarget	Return notation’s target
setData	Set notation’s data

getData

Function
Return a notation's data

Prototype
`String getData()`

Arguments
None

Returns
`String` -- node's data

getTarget

Function
Return a notation's target

Prototype
`String getTarget()`

Arguments
None

Returns
`String` -- node's target

setData

Function
Set a notation's data

Prototype
```
void setData(String data)
```

Arguments
`data` -- new data

Returns
None

Class: ProcessingInstruction

This class implements the PROCESSING_INSTRUCTION node type, a subclass of **Node**.

getData	Return the PI’s data
getTarget	Return the PI’s target
setData	Set the PI’s data

getData

Function
Return data for a processing instruction

Prototype
`String getData()`

Arguments
None

Returns
`String` -- PI's data

getTarget

Function
Return a processing instruction's target value

Prototype
`String getTarget()`

Arguments
None

Returns
`String` -- target value

setData

Function
Set the data for a processing instruction

Prototype
```
void setData(String data)
```

Arguments
`data` -- PI's new data

Returns
None

Class: Text

This class contains methods for accessing and modifying the data associated with text nodes (subclasses CharacterData).

splitText	Get data (value) of text node

splitText

Function

Split a text node in two. The original node retains its data up to the split point, and the remaining data is turned into a new text node which follows.

Prototype

```
Text* splitText(unsigned long offset)
```

Arguments

`offset` -- split point

Returns

`Text*` -- Pointer to new text node

Class: XMLParser

This class contains top-level methods for invoking the parser and returning high-level information about a document.

xmlinit	Initialize XML parser
xmlterm	Terminate XML parser
xmlparse	Parse a document from a file
xmlParseBuffer	Parse a document from a buffer
getContent	Returns the content model for an element
getModifier	Returns the modifier ('?', '*' or '+') for a content-model node
getDocument	Returns the root node of a parsed document
getDocumentElement	Returns the root element (node) of a parsed document
getDocType	Returns the document type string
isStandAlone	Returns the value of the standalone flag

xmlinit

Function
Initialize XML parser

Prototype
```
uword xmlinit(oratext *encoding,
              void (*msghdlr)(void *msgctx, oratext *msg, ub4 errcode),
              void *msgctx, lpxsaxcb *saxcb, void *saxcbctx, oratext *lang)
```

Arguments
`encoding` -- Input file's encoding, default **UTF8**
`msghdlr` -- Error message callback
`msgctx` -- User-defined context pointer passed to msghdlr

`saxcb` -- SAX callback structure (iff using SAX)
`saxcbctx` -- User-defined SAX context structure passed to SAX callback functions
`lang` -- Language for error message (not used)

Returns

`uword` -- Numeric error code, 0 meaning success

xmlterm

Function

Terminate XML parser, tear down, free memory, etc

Prototype

`void xmlterm()`

Arguments

None

Returns

`void`

xmlparse

Function

Parses a document

Prototype

`uword xmlparse(oratext *doc, oratext *encoding, ub4 flags)`

Arguments

`doc` -- document path

`encoding` -- document's encoding

`flags` -- Mask of flag bits

Flags:

`XML_FLAG_VALIDATE` -- Validate document against DTD

`XML_FLAG_DISCARD_WHITESPACE` -- Discard ignorable whitespace

Returns

`uword` -- Error code, 0 on success

xmlparseBuffer

Function

Parses a document

Prototype

```
uword xmlparseBuffer(oratext *buffer, size_t len, oratext *encoding, ub4 flags)
```

Arguments

`buffer` -- buffer containing document to parse

`len` -- length of document

`encoding` -- document's encoding

`flags` -- Mask of flag bits

Flags:

`XML_FLAG_VALIDATE` -- Validate document against DTD

`XML_FLAG_DISCARD_WHITESPACE` -- Discard ignorable whitespace

Returns

`uword` -- Error code, 0 on success

getContent

Function

Returns the content model for a node. Content model nodes are Nodes and can be traversed and examined with the same functions as the parsed document.

Prototype

```
Node* getContent(Node *node)
```

Arguments

`node` -- node whose content model to return

Returns

`Node*` -- root node of content model tree

getModifier

Function

Returns the modifier for a content model node. The modifier is one of `XMLCPMOD_NONE` (no modifier), `XMLCPMOD_OPT` ('?', optional), `XMLCPMOD_0MORE` ('*', zero or more), or `XMLCPMOD_1MORE` ('+', one or more).

Prototype

```
xmlcpmod getContent(Node *node)
```

Arguments

`node` -- content model node whose modifer to return

Returns

`xmlcpmod` -- enumeration as described above

getDocument

Function

After a document has been successfully parsed, returns a pointer to the root node of the document. Compare with `getDocumentElement` which returns the root *element* node.

Prototype

```
Node* getDocument()
```

Arguments

None

Returns

`Node*` -- Pointer to root node of document

getDocumentElement

Function

After a document has been successfully parsed, returns a pointer to the root element (node) of the document

Prototype

`Element* getDocumentElement()`

Arguments

None

Returns

`Element*` -- Pointer to root element (node) of document

getDocType

Function

Returns a pointer to a "DocType" structure which describes the DTD

Prototype

`DocumentType* getDocType()`

Arguments

None

Returns

`DocumentType*` -- Pointer to DTD descriptor

isStandalone

Function

Returns TRUE if the document is specified as standalone on the `<?xml?>` line, FALSE otherwise

Prototype

```
boolean isStandalone()
```

Arguments

None

Returns

`boolean` -- Value of standalone flag

C++ Parser APIs

XMLParser

C++ SAX API

The SAX API is based on callbacks. Instead of the entire document being parsed and turned into a data structure which may be referenced (by the DOM interface), the SAX interface is serial. As the document is processed, appropriate SAX user callback functions are invoked. Each callback function returns an error code, zero meaning success, any non-zero value meaning failure. If a non-zero code is returned, document processing is stopped.

To use SAX, an `xmlsaxcb` structure is initialized with function pointers and passed to the `xmlinit()` call. A pointer to a user-defined context structure may also be included; that context pointer will be passed to each SAX function.

Note this SAX functionality is identical to the C version.

SAX callback structure

```
typedef struct
{
   sword (*)(void *ctx);
   sword (*)(void *ctx);
   sword (*)(void *ctx, const oratext *name, struct xmlarray *attrs);
   sword (*)(void *ctx, const oratext *name);
   sword (*)(void *ctx, const oratext *ch, size_t len);
   sword (*)(void *ctx, const oratext *ch, size_t len);
   sword (*)(void *ctx, const oratext *target, const oratext *data);
   sword (*)(void *ctx, const oratext *name,
                            const oratext *publicId, const oratext *systemId);
   sword (*)(void *ctx, const oratext *name, const oratext *publicId,
                                  const oratext *systemId, const oratext
*notationName);
   sword (*)(void *ctx, const oratext *qname,
                              const oratext *local, const oratext *namespace);
} xmlsaxcb;
```

startDocument

Function

Called once when document processing is first starting

Prototype

`sword startDocument(void *ctx)`

Arguments

`ctx` -- User-defined context as passed to initialize()

Returns

`sword` -- Error code, 0 for success, non-0 for error.

endDocument

Function

Called once when document processing is finished

Prototype

`sword endDocument(void *ctx)`

Arguments

`ctx` -- User-defined context as passed to initialize()

Returns

`sword` -- Error code, 0 for success, non-0 for error.

startElement

Function
Called once for each new document element

Prototype
```
sword startElement(void *ctx, const oratext *name, struct xmlarray *attrs)
```

Arguments
`ctx` -- User-defined context as passed to initialize()

`name` -- name of node

`attrs` -- array of node's attributes

Returns
`sword` -- Error code, 0 for success, non-0 for error.

endElement

Function
Called once when each document element closes

Prototype
```
sword endElement(void *ctx, const oratext *name)
```

Arguments
`ctx` -- User-defined context as passed to initialize()

`name` -- name of node

Returns
`sword` -- Error code, 0 for success, non-0 for error.

characters

Function

Called for each piece of literal text

Prototype

```
sword characters(void *ctx, const oratext *ch, size_t len)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`ch` -- pointer to text

`len` -- number of character in text

Returns

`sword` -- Error code, 0 for success, non-0 for error.

IgnorableWhitespace

Function

Called for each piece of ignorable (non-significant) whitespace

Prototype

```
sword ignorableWhitespace(void *ctx, const oratext *ch, size_t len)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`ch` -- pointer to whitespace text

`len` -- number of characters of whitespace

Returns

`sword` -- Error code, 0 for success, non-0 for error.

processingInstruction

Function

Called once for each PI (Processing Instruction)

Prototype

```
sword processingInstruction(void *ctx, const oratext *target,
                            const oratext *data)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`target` -- PI target

`data` -- PI data

Returns

`sword` -- Error code, 0 for success, non-0 for error.

notationDecl

Function

Called once for each NOTATION

Prototype

```
sword notationDecl(void *ctx, const oratext *name,
                   const oratext *publicId, const oratext *systemId)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`name` -- name of notation

`publicId` -- Public ID

`systemId` -- System ID

Returns

`sword` -- Error code, 0 for success, non-0 for error.

unparsedEntityDecl

Function

Called once for each unparsed entity declaration

Prototype

```
sword unparsedEntityDecl(void *ctx, const oratext *name, const oratext
*publicId,
                         const oratext *systemId, const oratext *notationName)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`name` -- name of entity

`publicId` -- Public ID

`systemId` -- System ID

`notationName` -- notation name

Returns

`sword` -- Error code, 0 for success, non-0 for error.

nsStartElement

Function

Namespace variant of startElement: Called once for each new document element, when the element uses uses an explicit namespace

Prototype

```
sword startElement(void *ctx, const oratext *qname,
                   const oratext *local, const oratext *namespace)
```

Arguments

`ctx` -- User-defined context as passed to initialize()

`qname` -- qualified namespace

`local` -- umm

`namespace` -- yes, well

Returns

`sword` -- Error code, 0 for success, non-0 for error.

C++ *DOM* API's

Class (Subclass)

Attr	Node
CDATASection	Text
CharacterData	Node
Comment	CharacterData
Document	Node
DocumentFragment	Node
DocumentType	Node
DOMImplementation	
Element	Node
Entity	Node
EntityReference	Node
NamedNodeMap	
Node	
NodeList	
Notation	Node
ProcessingInstruction	Node
Text	CharacterData

8

Oracle XML Class Generator 1.0.0.0.0 (C++)

This chapter details the XML Class Generator for C++ and how it defines each element. It contains:

Document Type Defintiions

XML Documents with DTD

C++ Source files

Sample/Example Usage

Class: XMLClass Generator

Class: generated

Overview of the XML Class Generator for C++

The XML Class Generator takes a Document Type Definition (DTD) and generates classes for each defined element. Those classes are then used in a C++ program to construct XML documents conforming to the DTD. Supported operating systems are Solaris 2.6, Linux 2.2, and NT 4 (Service Pack 3 and above).

Input

Input is an XML document containing a DTD. The document body itself is ignored; only the DTD is relevant, though the dummy document must conform to the DTD. The underlying XML parser only accepts file names for the document and associated external entities. In future releases, no dummy document will be required, and URIs for additional protocols will be accepted.

The following input file encodings are supported:

UTF-8, UTF-16, US-ASCII, ISO-10646-UCS-2, ISO-8859-1, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8, ISO-8859-9, EUC-JP, SHIFT_JIS, BIG5, GB2312, KOI8-R, EBCDIC-CP-US, EBCDIC-CP-CA, EBCDIC-CP-NL, EBCDIC-CP-WT, EBCDIC-CP-DK, EBCDIC-CP-NO, EBCDIC-CP-FI, EBCDIC-CP-SE, EBCDIC-CP-IT, EBCDIC-CP-ES, EBCDIC-CP-GB, EBCDIC-CP-FR, EBCDIC-CP-HE, EBCDIC-CP-BE, EBCDIC-CP-CH, EBCDIC-CP-ROECE, EBCDIC-CP-YU, and EBCDIC-CP-IS.

In addition, any character set specified in Appendix A, Character Sets, of the Oracle National Language Support Guide may be used.
The default encoding is UTF-8. It is recommended that you set the default encoding explicitly if using only single byte character sets (such as US-ASCII or any of the ISO-8859 character sets) for performance up to 25% faster than with multibyte character sets such as UTF-8.

Output

Output is a pair of C++ source files, .cpp and .h, named after the DTD. Constructors are provided for each class (element) that allow an object to be created in two different ways: initially empty, then adding the children or data after the initial creation, or created with the initial full set of children or initial data. A method is provided for #PCDATA (and Mixed) elements to set the data and, when appropriate, set an element's attributes.

Relevant XML Standards

The W3C recommendation for Extensible Markup Language (XML) 1.0

The W3C recommendation for Document Object Model Level 1 1.0

The W3C proposed recommendation for Namespaces in XML

The Simple API for XML (SAX) 1.0

sample/Example usage

xmlcg Usage

The standalone parser may be called as an executable by invoking

```
bin/xmlcg like
xmlcg [flags] &lt;XML document&gt;
```

where the optional flags are as follows:

```
-d
directory
Specify output directory (default is current directory)
```

-e

encoding

Specify default input file encoding

-h help

show this usage help

Class: XMLClassGenerator

This class contain the method for generating classes based on a DTD.

METHOD INDEX

generate	Generate classes

METHODS

generate

Function:

Generates classes for the given DTD. Two files are created in the output directory outdir (or in the currect directory if outdir is NULL): DTDname.h and DTDname.cpp, both named after the DTD. One class is generated for each defined element in the DTD.

Prototype:

```
uword generate(DocumentType *dtd, char *outdir)
```

Arguments:

dtd -- DTD whose elements to generate classes for

outdir -- directory in which to place output files

Returns:

uword -- error code, 0 on success

Class: generated

A generated class is produced for each element defined in the DTD. It has the same name as the element.

Constructors are provided which create an empty element, or make it with an initial set of children or data. Methods are provided to add children or data after construction, and to set attributes. There are two styles of creation: make an empty element, then add the children one at a time, or construct the element with initial data or children. For example, given the element declaration

```
<!ELEMENT B (#PCDATA | F)*>
```

The following constructors will be provided:

B(Document *doc);

B(Document *doc, String data);

B(Document *doc, F *theF);

The first constructor just makes an empty element with no children. The second initializes it with PCDATA, and the third with a single child node of element F. An element like B which may contain PCDATA is also given a method to add the data post-construction:

```
void addData(Document *doc, String data);
```

The following usages are equivalent:

b = new B("data");

and

b = new B();

b->addData("data");

Similarly, the following are also equivalent:

f = new F(...);

b = new B(f);

and

f = new F(...);

b = new B();

b->addNode(f);

The presense of modifiers '?' (optional), '*' (zero or more), and '+' (one or more) is ignored when forming the constructors. For example, for the element

```
<!ELEMENT Sample (A* | (B, (C? | (D, E)*)) | F)+>
```

the following constructors are made:

Sample(Document *doc);

Sample(Document *doc, A *theA);

Sample(Document *doc, B *theB, C *theC);

Sample(Document *doc, B *theB, D *theD, E *theE);

Sample(Document *doc, F *theF);

as if the modifiers were not present. If you cannot make the desired final element using one of the forms which take initial children, you must start with the empty element and add nodes as needed with addNode as above.

For each attribute for an element, a method is provided to set its value, named setattrname. For example, for the element declaration

```
<!ELEMENT D (#PCDATA)>
<!ATTLIST D foo CDATA #REQUIRED>
```

the class D will have the method

```
Attr* setfoo(String value);
```

Note: The constructed element is not tested for validity as it is being made. The user to explicitly call the XMLParser's validate method on the final element.

METHOD INDEX

Constructors	Constructors for the class
addData	Adds PCDATA to the element
addNode	Adds a node to the element
setattribute	Sets one of the element's attributes

METHOD

Constructors

Function:

Constructs an element which will belong to the given document. The first form makes the element with no children (use addData and addNode as appropriate to fill it out). The second variable form is used to provide initial data or children, the exact choices of which depend on the element definition. See the example at the beginning of this document.

Prototype:

```
class(Document *doc)
class(Document *doc, ...)
```

Arguments:

doc -- document which the element belongs to

... -- varying arguments depending on the element definition

Returns:

none

addData

Function:

Adds data to the element. That is, appends to it a PCDATA subnode with the given value. If multiple addData calls are made, the node will have multiple PCDATA subnodes, which should be normalized when construction is finished.

Prototype:

```
void addData(Document *doc, String data)
```

Arguments:

doc -- document which the element belongs to

data -- data to be added

Returns:

none

addNode

Function:

Adds (append) a child node to the element. No effort is made to validate the resulting element structure at this time; it is the user's responsibility to form the element properly, which may be verified with XMLParser::validate.

Prototype:

`void addNode(node thenode)`

Arguments:

the node -- node to be added

Returns:

none

setattribute

Function:

Sets the element's attribute with the given value. One method is provided for each attribute, named after the attribute as setattribute.

Prototype:

`Attr* setattribute(String value)`

Arguments:

value -- the attribute's value

Returns:

Attr* -- the created attribute

Part IV

XML-SQL Utility (XSU) Packages

This section contains the following chapters:

- Chapter 9, "Oracle XSU Java API"
- Chapter 10, "XSU's PL/SQL API"

9 Oracle XSU Java API

This chapter contains the description of the XSU's java API:

- oracle.xml.sql.dml.OracleXMLSave
- oracle.xml.sql.query.OracleXMLQuery
- oracle.xml.sql.OracleXMLSQLException
- oracle.xml.sql.OracleXMLSQLNoRowsException

oracle.xml.sql.dml.OracleXMLSave

Syntax

```
public class OracleXMLSave extends java.lang.Object

java.lang.Object
  |
  +--oracle.xml.sql.dml.OracleXMLSave
```

Description

OracleXMLSave - this class supports canonical mapping from XML to object-relational tables or views. It supports inserts, updates and deletes. The user first creates the class by passing in the table name on which these DML operations need to be done. After that, the user is free to use the insert/update/delete on this table. A typical sequence might look like

```
OracleXMLSave sav = new OracleXMLSave(conn, "emp");
// insert processing
sav.insertXML(xmlDoc); -or-
// Update processing
String[]tempArr = new String[2];
tempArr[0] = "EMPNO";
// set the empno as the key for updates..
sav.setUpdateKeyColumnList(tempArray);
sav.updateXML(xmlDoc); -or-
sav.deleteXML(xmlDoc);
sav.close();
```

A lot of useful functions are provided in this class to help in identifying the key columns for update or delete and to restrict the columns being updated.

Member Summary	
Fields	
DATE_FORMAT	date format for use in setDateFormat(String)
DEFAULT_BATCH_SIZE	default insert batch size is 17
Constructors	
OracleXMLSave(Connection, String)	The public constructor for OracleXMLSave utility

Member Summary

Methods	
close()	It closes/deallocates all the context associated with this object.
deleteXML(Document)	Deletes records from the db. object corresponding to the records in an XML doc.
deleteXML(InputStream)	Deletes records from the db. object corresponding to the records in an XML doc.
Deletes records from the db. object corresponding to the records in an XML doc.	Deletes records from the db. object corresponding to the records in an XML doc.
deleteXML(URL)	Deletes records from the db. object corresponding to the records in an XML doc.
insertXML(Document)	Inserts an XML document into a specified table. Returns the number of rows processed.
insertXML(InputStream)	Inserts an XML document into a specified table. Returns the number of rows processed.
insertXML(String)	Inserts an XML document into a specified table. Returns the number of rows processed.
insertXML(URL)	Inserts an XML document into a specified table. Returns the number of rows processed.
setBatchSize(int)	This call changes the batch size used during DML operations.
setCommitBatch(int)	Sets the commit batch size.
setDateFormat(String)	Sets the format of the date tags.
setIgnoreCase(boolean)	Used to tell XSU to ignores the case of the tag names when matching them with the column names of the table/view.
setKeyColumnList(String[])	Sets the list of columns to be used for identifying a particular row in the database table during update or delete.
setRowTag(String)	Names the tag used in the XML doc., to enclose the XML elements corresponding to each row value..
setUpdateColumnList(String[])	Set the column values to be updated.
updateXML(Document)	Updates the table given the XML document.
updateXML(InputStream)	Updates the table given the XML document.
updateXML(String)	Updates the table given the XML document.
updateXML(URL)	Updates the table given the XML document.

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

DATE_FORMAT

`public static final java.lang.String DATE_FORMAT`

DEFAULT_BATCH_SIZE

`public static int DEFAULT_BATCH_SIZE`

default insert batch size is 17

Constructors

OracleXMLSave(Connection, String)

`public  OracleXMLSave(java.sql.Connection oconn, java.lang.String tabName)`

The public constructor for OracleXMLSave utility

Methods

close()

`public void close()`

deleteXML(Document)

`public int deleteXML(org.w3c.dom.Document doc)`

Deletes records from the db. object corresponding to the records in an XML doc.

deleteXML(InputStream)

`public int deleteXML(java.io.InputStream xmlStream)`

Deletes records from the db. object corresponding to the records in an XML doc.

deleteXML(String)

`public int deleteXML(java.lang.String xmlDoc)`

Deletes records from the db. object corresponding to the records in an XML doc.

deleteXML(URL)

`public int deleteXML(java.net.URL url)`

Deletes records from the db. object corresponding to the records in an XML doc.

insertXML(Document)

`public int insertXML(org.w3c.dom.Document doc)`

Inserts an XML document into a specified table. Returns the number of rows processed.

insertXML(InputStream)

`public int insertXML(java.io.InputStream xmlStream)`

Inserts an XML document into a specified table. Returns the number of rows processed.

insertXML(String)

`public int insertXML(java.lang.String xmlDoc)`

Inserts an XML document into a specified table. Returns the number of rows processed.

insertXML(URL)

`public int insertXML(java.net.URL url)`

Inserts an XML document into a specified table. Returns the number of rows processed .

setBatchSize(int)

`public void setBatchSize(int size)`

This call changes the batch size used during DML operations. When performing inserts, updates or deletes, it is better to batch the operations so that the database can execute it in one shot rather than as separate statements. The flip side is that more memory is needed to hold all the bind values before the operation is done. Note that when batching is used, the commits occur only in terms of batches. So if one of the statement inside a batch fails, the whole batch is rolled back. If this behaviour is unaccepatable, then set the batch size to 1. The default batch size is DEFAULT_BATCH_SIZE.

setCommitBatch(int)

`public void setCommitBatch(int size)`

Sets the commit batch size. The commit batch size refers to the number or records inserted after which a commit should follow. Note that if commitBatch is < 1 or the session is in "auto-commit" mode then the XSU does not make any explicit commit's. By default the commit-batch size is 0

setDateFormat(String)

```
public void setDateFormat(java.lang.String dateformat)
```

This sets the format of the date tags. By default, OracleXMLSave assumes that the date is in format 'MM/dd/yyyy HH:mm:ss' which is what is output by OracleXMLQuery is. You can override this by setting the format here. The format MUST be a valid format that can be understood by JDBC. See the JDBC documentation on java.text.SimpleDateFormat to get the list of valid date formats. This change is global. Passing a null value or an empty string resets this to the default format.

setIgnoreCase(boolean)

```
public void setIgnoreCase(boolean ignore)
```

setIgnoreCase ignores the case of the tag names when comparing them with the column names of the table/view. By default the tag names must be in the same case as that of the corresponding column names. Calling setIgnoreCase(true) will change it to ignore the case. Note: that after the first call to insertXML, we cache a lot of metadata. If setIgnoreCase status is changed, all the metadata has to be re-aquired.

setKeyColumnList(String[])

```
public void setKeyColumnList(java.lang.String[] keyColNames)
```

Sets the list of columns to be used for identifying a particular row in the database table during update or delete. This call is ignored for the insert case. The key columns must be set before updates can be done. IT is optional for deletes. When this key columns is set, then the values from these tags in the XML document is used to identify the database row for update or delete.

setRowTag(String)

```
public void setRowTag(java.lang.String rowTag)
```

setRowTag sets the name of the ROW tag that is used to distinguish each row value. Setting the value of this to null implies that there is no row tag present and the top level elements of the document correspond to the rows themselves.

setUpdateColumnList(String[])

```
public void setUpdateColumnList(java.lang.String[] updColNames)
```

Set the column values to be updated. This is only valid for inserts and updates, and is ignored for deletes. In case of insert, the default is to insert values to all the columns in the table. In case of updates, the default is to only update the columns corresponding to the tags present in the ROW element of the XML document. When specified, these columns alone will get updated in the update or insert statement. All other tags will be ignored.

updateXML(Document)

`public int updateXML(org.w3c.dom.Document doc)`
This flavor of updateXML takes in a stream to the document

updateXML(InputStream)

`public int updateXML(java.io.InputStream xmlStream)`
This flavor of updateXML takes in a stream to the document

updateXML(String)

`public int updateXML(java.lang.String xmlDoc)`
This flavor of updateXML takes in the XML document itself in the form of a String. Returns the number of Rows updated..!

updateXML(URL)

`public int updateXML(java.net.URL url)`
Updates an XML document from a specified URL into a specified table, given a database connection object and a list of column names to update The column names can be null. updKeyNames lists the names of all columns for which the where clause is checked updColNames lists the column names to update.. For example, consider the employee table emp, TABLE emp(empno NUMBER, ename VARCHAR2(20), hiredate DATE); Now, you want to update the table using an XML document and you want to use the tag values EMPNO to get the right row to update and only update the hiredate values. You can send in an XML document containing the HIREDATE value and the EMPNO value and call the updateXML routine with the updKeyColNames containing the EMPNO string. Returns the number of XML row elements processed. Note that this may or may not be equal to the number of database rows modified based on whether the rows selected through the XML document uniquely identified the rows in the table.

Inherited Member Summary
Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

oracle.xml.sql.query.OracleXMLQuery

Syntax

```
public class OracleXMLQuery extends java.lang.Object

java.lang.Object
|
+--oracle.xml.sql.query.OracleXMLQuery
```

Description

The `OracleXMLQuery` class does the generation of XML given a SQL query. Following is a very simple example:

```
import java.sql.*;
import oracle.xml.sql.query.*;
import oracle.jdbc.driver.*;

public class sample
{
public static void main(String args[]) throws Exception
{
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
Connection conn =
DriverManager.getConnection("jdbc:oracle:oci8:scott/tiger@");
OracleXMLQuery qry =  new OracleXMLQuery(conn, "select * from emp");
System.out.println(qry.getXMLString());
conn.close();
}
}
```

Member Summary

Fields	
DTD	The DTD is used to specified that the DTD is to be generated
ERROR_TAG	The ERROR_TAG specifies the default tag name for the ERROR document
MAXROWS_ALL	The MAXROWS_NONE specifies that all rows be included in the result
NONE	The NONE is used to specified that no DTD is to be generated

Member Summary	
ROW_TAG	The ROW_TAG specifies the default tag name for the ROW elements
ROWIDATTR_TAG	The ROWIDATTR_TAG specifies the default tag name for the ROW elements
ROWSET_TAG	The ROWSET_TAG specifies the default tag name for the document
SKIPROWS_ALL	The SKIPROWS_ALL specifies that all rows be skipped in the result.
Constructors	
OracleXMLQuery(Connection, ResultSet)	Constructor for the OracleXMLQueryObject.
OracleXMLQuery(Connection, String)	Constructor for the OracleXMLQueryObject.
OracleXMLQuery(OracleXMLDataSet)	Constructor for the OracleXMLQueryObject.
Methods	
close()	Close any open resource.
getXMLDOM()	getXMLDOM returns a DOM representation of the XML document using the OracleXMLDocGenDOM class to generate the DOM tree
getXMLDOM(Node)	The root node is also passed in.
getXMLDOM(Node, int)	This method returns a Document object containing the XML doc.
getXMLMetaData(int, boolean)	This functions returns the DTD for the XML document which would have been generated by a getXML call.
getXMLString()	getXMLString returns a string representation of the XML document using the OracleXMLDocGenString class to create the string; generates no DTD.
getXMLString(int)	getXMLString returns a string representation of the XML document using the OracleXMLDocGenString class to create the string.
getXMLString(Node)	All the children created are appended to this root node.
getXMLString(Node, int)	This method returns a String object containing the XML doc.
keepObjectOpen(boolean)	The default behavior for all the getXML functions which DO NOT TAKE in a ResultSet object is to close the ResultSet object and Statement objects at the end of the call; this method allows for change of this behaviour. This is very useful when retreiving few records at a time because it preserves all the metadata.
setCollIdAttrName(String)	
setDataHeader(Reader, String)	Sets the XML data header.
setDateFormat(String)	Sets the format of the generated dates in the XML doc.
setEncoding(String)	Sets the encoding PI (processing instruction) in the XML doc.

Member Summary	
setErrorTag(String)	setErrorTag sets the name of the error tag (default "ERROR") in case an error is raised.
setException(Exception)	Allows the user to pass in an exception, and have the XSU handle it.
setMaxRows(int)	This sets the maximum number of rows to be retreived from the query result after "skipRows" number of rows are skipped You can use one of MAXROWS_NONE, MAXROWS_ALL, MAXROWS_DEFAULT
setMetaHeader(Reader)	Sets the XML meta header.
setRaiseException(boolean)	setRaiseException function when called with the value true, will raise exceptions such as no-data found, instead of returning a NULL row out.
setRaiseNoRowsException(boolean)	Tells the XSU to throw or not to throw an OracleXMLNoRowsException in the case when for one reason or another, the XML doc generated is empty.
setRowIdAttrName(String)	Sets the name of the id attribute of the row enclosing tag.
setRowIdAttrValue(String)	Specifies the scalar column whose value is to be assigned to the id attribute of the row enclosing tag.
setRowsetTag(String)	Sets the tag to be used to enclose the xml dataset.
setRowTag(String)	Sets the tag to be used to enclose the xml element corresponding to a db.
setSkipRows(int)	Sets the number of rows to skip.
setStylesheetHeader(String)	Sets the stylesheet header (i.e. stylesheet processing instructions) in the generated XML doc.
setStylesheetHeader(String, String)	Sets the stylesheet header (i.e. stylesheet processing instructions) in the generated XML doc.
setXSLT(Reader, String)	Sets the stylesheet to be applied during XML generation.
setXSLT(String, String)	Sets the stylesheet to be applied during XML generation.
useLowerCaseTagNames()	This will set the case to be lower for all tag names.
useNullAttributeIndicator(boolean)	Specified weather to use an XML attribute to indicate NULLness; or to do it by omitting the inclusion of the particular entity in the XML document.
useUpperCaseTagNames()	This will set the case to be upper for all tag names.

Inherited Member Summary

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

DTD

`public static final int DTD`
The DTD is used to specified that the DTD is to be generated

ERROR_TAG

`public static final java.lang.String ERROR_TAG`
The ERROR_TAG specifies the default tag name for the ERROR document

MAXROWS_ALL

`public static final int MAXROWS_ALL`
The MAXROWS_NONE specifies that all rows be included in the result

NONE

`public static final int NONE`
The NONE is used to specified that no DTD is to be generated

ROW_TAG

`public static final java.lang.String ROW_TAG`
The ROW_TAG specifies the default tag name for the ROW elements

ROWIDATTR_TAG

`public static final java.lang.String ROWIDATTR_TAG`
The ROWIDATTR_TAG specifies the default tag name for the ROW elements

ROWSET_TAG

`public static final java.lang.String ROWSET_TAG`
The ROWSET_TAG specifies the default tag name for the document

SKIPROWS_ALL

```
public static final int SKIPROWS_ALL
```

The SKIPROWS_ALL specifies that all rows be skipped in the result This is the same as saying MAXROWS_NONE

Constructors

OracleXMLQuery(Connection, ResultSet)

```
public  OracleXMLQuery(java.sql.Connection conn, java.sql.ResultSet rset)
```

OracleXMLQuery(Connection, String)

```
public  OracleXMLQuery(java.sql.Connection conn, java.lang.String sqlString)
```

OracleXMLQuery(OracleXMLDataSet)

```
public  OracleXMLQuery(oracle.xml.sql.dataset.OracleXMLDataSet dset)
```

Methods

close()

```
public void close()
```

close any open resource, created by the OracleXML engine. This will not close for instance resultset supplied by the user

getXMLDOM()

```
public oracle.xml.sql.query.Document getXMLDOM()
```

getXMLDOM returns a DOM representation of the XML document using the OracleXMLDocGenDOM class to generate the DOM tree

getXMLDOM(Node)

```
public oracle.xml.sql.query.Document getXMLDOM(oracle.xml.sql.query.Node root)
```

The root node is also passed in. All the children created are appended to this root node. The return value is the whole document (the owner document of the root Node).

getXMLDOM(Node, int)

```
public oracle.xml.sql.query.Document getXMLDOM(oracle.xml.sql.query.Node root,
int metaType)
```

This method returns a Document object containing the XML doc. The method takes a Node object which, if not NULL, is treated as the "root" node for the DOM tree of the XML doc. The other parameter is used to specify the type of XML metadata to be generated, if any.

getXMLMetaData(int, boolean)

```
public java.lang.String getXMLMetaData(int metaType, boolean withVersion)
```

This functions returns the DTD for the XML document which would have been generated by a getXML call. The "metaType" parameter specifies the type of XML metadata to be generated. Valid values is DTD. The withVersion parameter specifies if version header is to be generated or not.

getXMLMetaData(int, boolean, OracleXMLDocGen)

```
public void getXMLMetaData(int metaType, boolean withVersion,
oracle.xml.sql.docgen.OracleXMLDocGen doc)
```

getXMLString()

```
public java.lang.String getXMLString()
```

getXMLString returns a string representation of the XML document using the OracleXMLDocGenString class to create the string; generates no DTD.

getXMLString(int)

```
public java.lang.String getXMLString(int metaType)
```

getXMLString returns a string representation of the XML document using the OracleXMLDocGenString class to create the string.

getXMLString(Node)

```
public java.lang.String getXMLString(oracle.xml.sql.query.Node root)
```

All the children created are appended to this root node. A new ROWTAG tag is created under this root node and the children are added.

getXMLString(Node, int)

```
public java.lang.String getXMLString(oracle.xml.sql.query.Node root, int
metaType)
```

This method returns a String object containing the XML doc. The method takes a Node object which, if not NULL, is treated as the "root" node for the DOM tree of the XML doc. The other parameter is used to specify the type of XML metadata to be generated, if any. Valid values are OracleXMLQuery.NONE and

OracleXMLQuery.DTD. Note that when the root is non-null, no DTD is generated even if requested.

keepObjectOpen(boolean)

`public void keepObjectOpen(boolean alive)`
The default behavior for all the getXML functions which DO NOT TAKE in a ResultSet object is to close the ResultSet object and Statement objects at the end of the call. If you need to use the persistant feature, where by calling getXML repeatedly you get the next set of rows, you need to turn off this behavior by calling this function with value true. i.e. OracleXMLQuery would not close the ResultSet and Statement objects after the getXML calls. You can call the close() function to explicitly close the cursor state.

setCollIdAttrName(String)

`public void setCollIdAttrName(java.lang.String collIdAttr)`
Sets the collection element's id-attribute name. Passing in a null, or an empty string with inhibit the printing of the collIDAttr which is also the default behaviour.

setDataHeader(Reader, String)

`public void setDataHeader(java.io.Reader header, java.lang.String docTag)`
Sets the XML data header. When set, the header is inserted at the begining of the data part of each XML document generated by this object. The docTag parameter specifies the tag name to be used to enclose the XML resulting from adding the "header" and the generated XML. To unset the header, call this function with null passed in for the "header" parameter.

setDateFormat(String)

`public void setDateFormat(java.lang.String mask)`
Sets the format of the generated dates in the XML doc. The syntax of the date format patern (i.e. the date mask), should conform to the requirements of the java.text.SimpleDateFormat class. Setting the mask to `null` or an empty string, unsets the date mask.

Parameters:

`mask` - the date mask

setEncoding(String)

`public void setEncoding(java.lang.String enc)`

Sets the encoding PI (processing instruction) in the XML doc. If `null` or an empty string are specified as the encoding, then the default characterset is specified in the encoding PI.

Parameters:

`enc` - characterset encoding of the XML doc

setErrorTag(String)

`public void setErrorTag(java.lang.String errorName)`
setErrorTag sets the name of the error tag (default "ERROR") in case an error is raised. Remember that when an error is raised NO DTD is generated.

setException(Exception)

`public void setException(java.lang.Exception e)`
Allows the user to pass in an exception, and have the XSU handle it.

setMaxRows(int)

`public void setMaxRows(int maxRows)`
This sets the maximum number of rows to be retreived from the query result after "skipRows" number of rows are skipped You can use one of MAXROWS_NONE, MAXROWS_ALL, MAXROWS_DEFAULT

setMetaHeader(Reader)

`public void setMetaHeader(java.io.Reader header)`
Sets the XML meta header. When set, the header is inserted at the begining of the metadata part (e.g. DTD) of each XML document generated by this object. To unset the header, call this function with null passed in for the "header" parameter.

setRaiseException(boolean)

`public void setRaiseException(boolean exp)`
setRaiseException function when called with the value true, will raise exceptions such as no-data found, instead of returning a NULL row out. You might want to catch the exception and handle the case

setRaiseNoRowsException(boolean)

`public void setRaiseNoRowsException(boolean exp)`
set this to true to raise an Exception if no data is found default is false, meaning if no rows are found an empty document instance will be given Remember that

setting this would raise the OracleXMLSQLException even if the actual ResultSet generated Data but because of the maxRows and skipRows parameter, no Rows were selected.

setRowIdAttrName(String)

`public void setRowIdAttrName(java.lang.String rowIdTag)`

This sets the name of the attribute "ID" used in the row separator. By giving a value of null, you can turn off the ID printing

setRowIdAttrValue(String)

`public void setRowIdAttrValue(java.lang.String colName)`

This makes the ROW tag's id attribute to have the same value as that of the column name specified. The column must be a scalar column. A value of null would revert it to the default behaviour which is the row count.

setRowsetTag(String)

`public void setRowsetTag(java.lang.String rowsetName)`

Sets the tag name which is to be used as markup for each resulting record / row. The default tag name is "ROW"; a null or empty string implies that each row is not to be markup encapsulated (i.e. no row separators are to be used. In this situation the result set produced by the query can either contain many record which are one column wide, or one record which is many columns wide.

setRowTag(String)

`public void setRowTag(java.lang.String rowTagName)`

Sets the tag name which is to be used as markup for each resulting record / row. The default tag name is "ROW"; a null or empty string implies that each row is not to be markup encapsulated (i.e. no row separators are to be used. In this situation the result set produced by the query can either contain many record which are one column wide, or one record which is many columns wide.

setSkipRows(int)

`public void setSkipRows(int skipRows)`

This sets the number of rows to skip before printing the the result You can use one of SKIPROWS_NONE, SKIPROWS_ALL, SKIPROWS_DEFAULT

setStylesheetHeader(String)

`public void setStylesheetHeader(java.lang.String styleSheetURI)`

setStylesheetHeader(String, String)

`public void setStylesheetHeader(java.lang.String styleSheetURI, java.lang.String styleSheetType)`

setXSLT(Reader, String)

`public void setXSLT()`
Registers a stylesheet to be applied to generated XML. If a stylesheet was already registered, it gets replaced by the new one. To un-register the stylesheet pass in a `null` or an empty string for the `stylesheet` argument.

setXSLT(String, String)

`public void setXSLT()`
Registers a stylesheet to be applied to generated XML. If a stylesheet was already registered, it gets replaced by the new one. To un-register the stylesheet pass in a `null` or an empty string for the `stylesheet` argument.

useLowerCaseTagNames()

`public void useLowerCaseTagNames()`
This will set the case to be lower for all tag names. Note, make this call after all the desired tags have been set.

useNullAttributeIndicator(boolean)

`public void useNullAttributeIndicator(boolean useNullAttr)`
Specified weather to use an XML attribute to indicate NULLness; or to do it by omitting the inclusion of the particular entity in the XML document.

useUpperCaseTagNames()

`public void useUpperCaseTagNames()`
This will set the case to be upper for all tag names. Note, make this call after all the desired tags have been set.

oracle.xml.sql.OracleXMLSQLException

Syntax

```
public class OracleXMLSQLException extends java.lang.RuntimeException

java.lang.Object
|
+--java.lang.Throwable
|
+--java.lang.Exception
|
+--java.lang.RuntimeException
|
+--oracle.xml.sql.OracleXMLSQLException
```

Direct Known Subclasses:

`OracleXMLSQLNoRowsException`

All Implemented Interfaces:

`java.io.Serializable`

Description

Member Summary

Constructors	
OracleXMLSQLException(Exception)	
OracleXMLSQLException(Exception, String)	
OracleXMLSQLException(String)	
OracleXMLSQLException(String, Exception)	
OracleXMLSQLException(String, Exception, String)	
OracleXMLSQLException(String, int)	
OracleXMLSQLException(String, int, String)	

Member Summary	
OracleXMLSQLException(String, String)	
Methods	
getErrorCode()	
getParentException()	returns the original exception, if there was one; otherwise, it returns null
getXMLErrorString()	prints the XML error string with the given error tag name
getXMLSQLErrorString()	prints the SQL parameters as well in the error message
setErrorTag(String)	Sets the error tag name which is then used by getXMLErrorString and getXMLSQLErrorString, to generate xml error reports

Inherited Member Summary

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, getMessage, printStackTrace, printStackTrace, printStackTrace, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

OracleXMLSQLException(Exception)

```
public OracleXMLSQLException(java.lang.Exception e)
```

OracleXMLSQLException(Exception, String)

```
public OracleXMLSQLException(java.lang.Exception e, java.lang.String
errorTagName)
```

OracleXMLSQLException(String)

```
public OracleXMLSQLException(java.lang.String message)
```

OracleXMLSQLException(String, Exception)

```
public OracleXMLSQLException(java.lang.String message, java.lang.Exception e)
```

OracleXMLSQLException(String, Exception, String)

```
public  OracleXMLSQLException(java.lang.String message, java.lang.Exception e,
java.lang.String errorTagName)
```

OracleXMLSQLException(String, int)

```
public  OracleXMLSQLException(java.lang.String message, int errorCode)
```

OracleXMLSQLException(String, int, String)

```
public  OracleXMLSQLException(java.lang.String message, int errorCode,
java.lang.String errorTagName)
```

OracleXMLSQLException(String, String)

```
public  OracleXMLSQLException(java.lang.String message, java.lang.String
errorTagName)
```

Methods

getErrorCode()

```
public int getErrorCode()
```

getParentException()

```
public java.lang.Exception getParentException()
```

returns the original exception, if there was one; otherwise, it returns null

getXMLErrorString()

```
public java.lang.String getXMLErrorString()
```

prints the XML error string with the given error tag name

getXMLSQLErrorString()

```
public java.lang.String getXMLSQLErrorString()
```

prints the SQL parameters as well in the error message

setErrorTag(String)

```
public void setErrorTag(java.lang.String tagName)
```

Sets the error tag name which is then used by getXMLErrorString and getXMLSQLErrorString, to generate xml error reports

oracle.xml.sql.OracleXMLSQLNoRowsException

Syntax

```
public class OracleXMLSQLNoRowsException extends OracleXMLSQLException

java.lang.Object
|
+--java.lang.Throwable
|
+--java.lang.Exception
|
+--java.lang.RuntimeException
|
+--OracleXMLSQLException
|
+--oracle.xml.sql.OracleXMLSQLNoRowsException
```

All Implemented Interfaces:

`java.io.Serializable`

Description

Member Summary

Constructors	
OracleXMLSQLNoRowsException()	
OracleXMLSQLNoRowsException(String)	

Inherited Member Summary

Methods inherited from interface OracleXMLSQLException

getErrorCode(), getParentException(), getXMLErrorString(), getXMLSQLErrorString(), setErrorTag(String)

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, getMessage, printStackTrace, printStackTrace, printStackTrace, toString

Inherited Member Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructors

OracleXMLSQLNoRowsException()

```
public  OracleXMLSQLNoRowsException()
```

OracleXMLSQLNoRowsException(String)

```
public  OracleXMLSQLNoRowsException(java.lang.String errorTag)
```

10 XSU’s PL/SQL API

XSU’s PL/SQL API consists of two packages:

- DBMS_XMLQuery -- provides DB_to_XML type functionality.
- DBMS_XMLSave -- provides XML_to_DB type functionality.

DBMS_XMLQuery

Types:

ctxType

The type of the query context handle. This the return type of "DBMS_ XMLQuery.newContext()".

Constants:

DEFAULT_ROWSETTAG

The tag name for the element enclosing the XML generated from the result set (i.e. for most cases the root node tag name) -- ROWSET

DEFAULT_ERRORTAG

The default tag to enclose raised errors -- ERROR.

DEFAULT_ROWIDATTR

The default name for the cardinality attribute of XML elements coresponding to db. records. -- NUM

DEFAULT_ROWTAG

The default tag name for the element cooresponding to db. records. -- ROW

DEFAULT_DATE_FORMAT

Default date mask. -- 'MM/dd/yyyy HH:mm:ss'

ALL_ROWS

The ALL_ROWS parameter is to indicate that all rows are needed in the output.

NONE

Used to specifies that the output should not coOracle8*i*XML Reference Guidentain any XML metadata (e.g. no DTD).

DTD

Used to specify that the generation of the DTD is desired.

LOWER_CASE

Use lower cased tag names.

UPPER_CASE

Use upper case tag names.

Function and Procedure Index:

PROCEDURE

`closeContext(ctxType)`
It closes / deallocates a particular query context

FUNCTION

`getDTD(ctxType, BOOLEAN := false)`

RETURN

`CLOB`
Generates the DTD based on the SQL query used to init.

PROCEDURE

`getDTD(ctxType, CLOB, BOOLEAN := false)`
Generates the DTD based on the SQL query used to init.

PROCEDURE

`getExceptionContent(ctxType, NUMBER, VARCHAR2)`

FUNCTION

`getXML(ctxType, NUMBER := NONE)`

RETURN

`CLOB`
Generates the XML doc.

PROCEDURE

`getXML(ctxType, CLOB, NUMBER := NONE)`
Generates the XML doc.

FUNCTION

`newContext(VARCHAR2)`

RETURN

`ctxType`

It creates a query context, and it returns the context handle.

FUNCTION

`newContext(CLOB)`

RETURN

`ctxType`
It creates a query context, and it returns the context handle.

PROCEDURE

`propagateOriginalException(ctxType, BOOLEAN)`
Tells the XSU that if an exception is raised, and is being thrown, the XSU should throw the very exception raised; rather then, wrapping it with an OracleXMLSQLException.

PROCEDURE

`setBindValue(ctxType, VARCHAR2, VARCHAR2)`
Sets a value for a particular bind name.

PROCEDURE

`setCollIdAttrName(ctxType, VARCHAR2`
Sets the name of the id attribute of the collection element's separator tag.

PROCEDURE

`setDataHeader(ctxType, CLOB := null, VARCHAR2 := null)`
Sets the xml data header.

PROCEDURE

`setDateFormat(ctxType, VARCHAR2)`
Sets the format of the generated dates in the XML doc.

PROCEDURE

`setErrorTag(ctxType, VARCHAR2)`
Sets the tag to be used to enclose the xml error docs.

PROCEDURE

`setMaxRows (ctxType, NUMBER)`
Sets the max number of rows to be converted to XML.

PROCEDURE

`setMetaHeader(ctxType, CLOB := null)`
Sets the XML meta header.

PROCEDURE

`setRaiseException(ctxType, BOOLEAN)`
Tells the XSU to throw the raised exceptions.

PROCEDURE

`setRaiseNoRowsException(ctxType, BOOLEAN)`
Tells the XSU to throw or not to throw an OracleXMLNoRowsException in the case when for one reason or another, the XML doc generated is empty.

PROCEDURE

`setRowIdAttrName(ctxType, VARCHAR2)`
Sets the name of the id attribute of the row enclosing tag.

PROCEDURE

`setRowIdAttrValue(ctxType, VARCHAR2)`
Specifies the scalar column whose value is to be assigned to the id attribute of the row enclosing tag.

PROCEDURE

`setRowsetTag(ctxType, VARCHAR2)`
Sets the tag to be used to enclose the xml dataset.

PROCEDURE

`setRowTag(ctxType, VARCHAR2)`
Sets the tag to be used to enclose the xml element corresponding to a db.

PROCEDURE

`setSkipRows(ctxType, NUMBER)`
Sets the number of rows to skip.

PROCEDURE

`setStylesheetHeader(ctxType, VARCHAR2, VARCHAR2 := 'text/xsl')`
Sets the stylesheet header (i.e.

PROCEDURE

`setTagCase(ctxType, NUMBER)`
Specified the case of the generated XML tags.

PROCEDURE

`setXSLT(ctxType, VARCHAR2, VARCHAR2 := null)`
Registers a stylesheet to be applied to generated XML.

PROCEDURE

`setXSLT(ctxType, CLOB, VARCHAR2 := null)`
Registers a stylesheet to be applied to generated XML.

PROCEDURE

`useNullAttributeIndicator(ctxType, BOOLEAN)`
Specified weather to use an XML attribute to indicate NULLness, or to do it by omitting the inclusion of the particular entity in the XML document.

Functions and Procedures:

newContext

FUNCTION

`newContext(sqlQuery IN VARCHAR2)`

RETURN

`ctxType`
It creates a query context, and it returns the context handle.

Parameters

sqlQuery - SQL query, the results of which to convert to XML

Returns:

The context handle.

newContext

FUNCTION

`newContext(sqlQuery IN CLOB)`

RETURN

`ctxType`
It creates a query context, and it returns the context handle.

Parameters

sqlQuery - SQL query, the results of which to convert to XML

Returns:

The context handle.

closeContext

PROCEDURE

`closeContext(ctxHdl IN ctxType)`
It closes / deallocates a particular query context

Parameters

ctxHdl - context handle

setRowsetTag

PROCEDURE

`setRowsetTag(ctxHdl IN ctxType, tag IN VARCHAR2)`
Sets the tag to be used to enclose the xml dataset.

Parameters

`ctxHdl - context handle`
tag - tag name

setRowTag

PROCEDURE

`setRowTag(ctxHdl IN ctxType, tag IN VARCHAR2)`
Sets the tag to be used to enclose the xml element corresponding to a db. record.

Parameters

`ctxHdl - context handle`
tag - tag name

setErrorTag

PROCEDURE

`setErrorTag(ctxHdl IN ctxType, tag IN VARCHAR2)`
Sets the tag to be used to enclose the xml error docs.

Parameters

ctxHdl - context handle

tag - tag name

setRowIdAttrName

PROCEDURE

`setRowIdAttrName(ctxHdl IN ctxType, attrName IN VARCHAR2)`
Sets the name of the id attribute of the row enclosing tag. Passing null or an empty string for the tag results the row id attribute to be omitted.

Parameters

`ctxHdl - context handle`
`attrName - attribute name`

setRowIdAttrValue

PROCEDURE

`setRowIdAttrValue(ctxHdl IN ctxType, colName IN VARCHAR2)`
Specifies the scalar column whose value is to be assigned to the id attribute of the row enclosing tag. Passing null or an empty string for the colName results the row id attribute being assigned the row count value (i.e. 0, 1, 2 and so on).

Parameters

ctxHdl - context handle

colName - column whose value is to be assigned to the row id attr

setCollIdAttrName

PROCEDURE

`setCollIdAttrName(ctxHdl IN ctxType, attrName IN VARCHAR2)`
Sets the name of the id attribute of the collection element's separator tag. Passing null or an empty string for the tag results the row id attribute to be omitted.

Parameters

ctxHdl - context handle

attrName - attribute name

useNullAttributeIndicator

PROCEDURE

`useNullAttributeIndicator(ctxHdl IN ctxType, flag IN BOOLEAN)`
Specified weather to use an XML attribute to indicate NULLness, or to do it by omitting the inclusion of the particular entity in the XML document.

Parameters

ctxHdl - context handle

flag - use attribute to indicate null?

setTagCase

PROCEDURE

`setTagCase(ctxHdl IN ctxType, tCase IN NUMBER)`
Specified the case of the generated XML tags.

Parameters

ctxHdl - context handle

tCase - the tag's case (0-asAre, 1-lower, 2-upper)

setDateFormat

PROCEDURE

`setDateFormat(ctxHdl IN ctxType, mask IN VARCHAR2)`
Sets the format of the generated dates in the XML doc. The syntax of the date format patern (i.e. the date mask), should conform to the requirements of the java.text.SimpleDateFormat class. Setting the mask to null or an empty string, results the use of the default mask -- DEFAULT_DATE_FORMAT.

Parameters

ctxHdl - context handle

mask - the date mask

setMaxRows

PROCEDURE

`setMaxRows (ctxHdl IN ctxType, rows IN NUMBER)`

Sets the max number of rows to be converted to XML. By default there is no max set.

Parameters

ctxHdl - context handle

rows - max number of rows to generate

setSkipRows

PROCEDURE

`setSkipRows(ctxHdl IN ctxType, rows IN NUMBER)`
Sets the number of rows to skip. By default 0 rows are skipped.

Parameters

ctxHdl - context handle

rows - number of rows to skip

setStylesheetHeader

PROCEDURE

```
setStylesheetHeader(ctxHdl IN ctxType, uri IN VARCHAR2, type IN VARCHAR2 :=
'text/xsl')
```
Sets the stylesheet header (i.e. stylesheet processing instructions) in the generated XML doc. Note: Passing null for the uri argument will unset the stylesheet header and the stylesheet type.

Parameters

ctxHdl - context handle

uri - stylesheet URI

type - stylesheet type; defaults to 'text/xsl'

setXSLT

PROCEDURE

`setXSLT(ctxHdl IN ctxType, uri IN VARCHAR2, ref IN VARCHAR2 := null)`

Registers a stylesheet to be applied to generated XML. If a stylesheet was already registered, it gets replaced by the new one. To un-register the stylesheet pass in a null for the uri argument.

Parameters

ctxHdl - context handle

uri - stylesheet URI

ref - URL for include, import and external entities

setXSLT

PROCEDURE

```
setXSLT(ctxHdl IN ctxType, stylesheet CLOB, ref IN VARCHAR2 := null)
```

Registers a stylesheet to be applied to generated XML. If a stylesheet was already registered, it gets replaced by the new one. To un-register the stylesheet pass in a null or an empty string for the stylesheet argument.

Parameters

ctxHdl - context handle

stylesheet - the stylesheet

ref - URL for include, import and external entities

setBindValue

PROCEDURE

```
setBindValue(ctxHdl IN ctxType, bindName IN VARCHAR2, bindValue IN VARCHAR2)
```

Sets a value for a particular bind name.

Parameters

ctxHdl - context handle

bindName - bind name

bindValue - bind value

setMetaHeader

PROCEDURE

`setMetaHeader(ctxHdl IN ctxType, header IN CLOB := null)`

Sets the XML meta header. When set, the header is inserted at the begining of the metadata part (DTD or XMLSchema) of each XML document generated by this object. Note that the last meta header specified is the one that is used; furthermore, passing in null for the header, parameter unsets the meta header.

Parameters

ctxHdl - context handle

header - header

setDataHeader

PROCEDURE

`setDataHeader(ctxHdl IN ctxType, header IN CLOB := null, tag IN VARCHAR2 := null)`

Sets the xml data header. The data header is an XML entity which is appended at the begining of the query-generated xml entity (ie. rowset). The two entities are enclosed by the tag specified via the docTag argument. Note that the last data header specified is the one that is used; furthermore, passing in null for the header, parameter unsets the data header.

Parameters

ctxHdl - context handle

header - header

tag - tag used to enclose the data header and the rowset

setRaiseException

PROCEDURE

`setRaiseException(ctxHdl IN ctxType, flag IN BOOLEAN)`

Tells the XSU to throw the raised exceptions. If this call isn't made or if false is passed to the flag argument, the XSU catches the SQL exceptions and generates an XML doc out of the exception's message.

Parameters

ctxHdl - context handle

flag - throw raised exceptions?

setRaiseNoRowsException

PROCEDURE

`setRaiseNoRowsException(ctxHdl IN ctxType, flag IN BOOLEAN)`
Tells the XSU to throw or not to throw an OracleXMLNoRowsException in the case when for one reason or another, the XML doc generated is empty. By default, the exception is not thrown.

Parameters

ctxHdl - context handle

flag - throw OracleXMLNoRowsException if no data?

propagateOriginalException

PROCEDURE

`propagateOriginalException(ctxHdl IN ctxType, flag IN BOOLEAN)`
Tells the XSU that if an exception is raised, and is being thrown, the XSU should throw the very exception raised; rather then, wrapping it with an OracleXMLSQLException.

Parameters

ctxHdl - context handle

flag - propagate original exception?

getExceptionContent

PROCEDURE

`getExceptionContent(ctxHdl IN ctxType, errNo OUT NUMBER, errMsg OUT VARCHAR2)`
Via its arguments, this method returns the thrown exception's error code and error message (i.e. sql error code) This is to get around the fact that the jvm throws an exception on top of whatever exception was raised; thus, rendering pl/sql unable to access the original exception.

Parameters
ctxHdl - context handle

errNo - error number

errMsg - error message

getDTD

FUNCTION
`getDTD(ctxHdl IN ctxType, withVer IN BOOLEAN := false)`

RETURN
CLOB

Generates the DTD based on the SQL query used to init. the context.

Parameters
ctxHdl - context handle

withVer - generate the version info?

Returns:
The DTD.

getDTD

PROCEDURE
`getDTD(ctx IN ctxType, xDoc IN CLOB, withVer IN BOOLEAN := false)`
Generates the DTD based on the SQL query used to init. the context.

Parameters
ctxHdl - context handle

xDoc - lob into which to write the generated XML doc

withVer - generate the version info?

getXML

FUNCTION

`getXML(ctxHdl IN ctxType, metaType IN NUMBER := NONE)`

RETURN

CLOB

Generates the XML doc. based on the SQL query used to init. the context.

Parameters

ctxHdl - context handle

metaType - xml metadata type (i.e. none or DTD)

Returns:

The XML document.

getXML

PROCEDURE

`getXML(ctxHdl IN ctxType, xDoc IN CLOB, metaType IN NUMBER := NONE)`
Generates the XML doc. based on the SQL query used to init. the context.

Parameters

ctxHdl - context handle

xDoc - lob into which to write the generated XML doc

metaType - xml metadata type (i.e. none or DTD)

DBMS_XMLSave

Types

ctxType

The type of the query context handle. This the return type of "DBMS_
XMLSave.newContext()".

Constants:

DEFAULT_ROWTAG

The default tag name for the element cooresponding to db. records. -- ROW

DEFAULT_DATE_FORMAT

Default date mask. -- ’MM/dd/yyyy HH:mm:ss’

MATCH_CASE

Used to specify that when mapping XML elements to DB. entities the XSU should be case sensitive.

IGNORE_CASE

Used to specify that when mapping XML elements to DB. entities the XSU should be case insensitive.

Function and Procedure Index:

PROCEDURE

`clearKeyColumnList(ctxType)`
Clears the key colubmn list.

PROCEDURE

`clearUpdateColumnList(ctxType)`
Clears the update column list.

PROCEDURE

`closeContext(ctxType)`
It closes / deallocates a particular save context

FUNCTION

`deleteXML(ctxType, CLOB)`

RETURN

NUMBER

Deletes records specified by data from the XML document, from the table specified at the context creation time.

FUNCTION

`deleteXML(ctxType, VARCHAR2)`

RETURN

NUMBER

Deletes records specified by data from the XML document, from the table specified at the context creation time.

PROCEDURE

`getExceptionContent(ctxType, NUMBER, VARCHAR2)`
Via its arguments, this method returns the thrown exception's error code and error message (i.e.

FUNCTION

`insertXML(ctxType, CLOB)`

RETURN

NUMBER

Inserts the XML document into the table specified at the context creation time.

FUNCTION

`insertXML(ctxType, VARCHAR2)`

RETURN
NUMBER

Inserts the XML document into the table specified at the context creation time.

FUNCTION
`newContext(targetTable IN VARCHAR2)`

RETURN
`ctxType`
It creates a save context, and it returns the context handle.

PROCEDURE
`propagateOriginalException(ctxType, BOOLEAN)`
Tells the XSU that if an exception is raised, and is being thrown, the XSU should throw the very exception raised; rather then, wrapping it with an OracleXMLSQLException.

PROCEDURE
`setBatchSize(ctxType, NUMBER)`
Changes the batch size used during DML operations.

PROCEDURE
`setCommitBatch(ctxType, NUMBER)`
Sets the commit batch size.

PROCEDURE
`setDateFormat(ctxType, VARCHAR2)`
Describes to the XSU the format of the dates in the XML document.

PROCEDURE
`setIgnoreCase(ctxType, NUMBER)`
The XSU does mapping of XML elements to db.

PROCEDURE
`setKeyColumn(ctxType, VARCHAR2)`
This methods adds a column to the "key column list".

PROCEDURE

`setRowTag(ctxType, VARCHAR2)`
Names the tag used in the XML doc., to enclose the XML elements corresponding to db.

PROCEDURE

`setUpdateColumn(ctxType, VARCHAR2)`
Adds a column to the "update column list".

PROCEDURE

`getExceptionContent(ctxType, NUMBER, VARCHAR2)`
Updates the table specified at the context creation time with data from the XML document.

PROCEDURE

`propagateOriginalException(ctxType, BOOLEAN)`
Updates the table specified at the context creation time with data from the XML document.

Functions and Procedures:

newContext

FUNCTION
`newContext(targetTable IN VARCHAR2)`

RETURN
`ctxType`
It creates a save context, and it returns the context handle.

Parameters
targetTable **-** the target table into which to load the XML doc

Returns
The context handle.

closeContext

PROCEDURE
`closeContext(ctxHdl IN ctxType)`
It closes / deallocates a particular save context

Parameters
`ctxHdl - context handle`

setRowTag

PROCEDURE
`setRowTag(ctxHdl IN ctxType, tag IN VARCHAR2)`
Names the tag used in the XML doc., to enclose the XML elements corresponding to db. records.

Parameters
ctxHdl **-** context handle

tag - tag name

setIgnoreCase

PROCEDURE

`setIgnoreCase(ctxHdl IN ctxType, flag IN NUMBER)`
The XSU does mapping of XML elements to db. columns / attrs. based on the element names (xml tags). This function tells the XSU to do this match case insensitive.

Parameters

ctxHdl - context handle

flag - ignore tag case in the XML doc? 0-false 1-true

setDateFormat

PROCEDURE

`setDateFormat(ctxHdl IN ctxType, mask IN VARCHAR2)`
Describes to the XSU the format of the dates in the XML document. The syntax of the date format patern (i.e. the date mask), should conform to the requirements of the java.text.SimpleDateFormat class. Setting the mask to null or an empty string, results the use of the default mask -- OracleXMLCore.DATE_FORMAT.

Parameters

ctxHdl - context handle

mask - the date mask

setBatchSize

PROCEDURE

`setBatchSize(ctxHdl IN ctxType, batchSize IN NUMBER);`
Changes the batch size used during DML operations. When performing inserts, updates or deletes, it is better to batch the operations so that they get executed in one shot rather than as separate statements. The flip side is that more memory is needed to buffer all the bind values. Note that when batching is used, a commit occurs only after a batch is executed. So if one of the statement inside a batch fails, the whole batch is rolled back. This is a small price to pay considering the

performance gain; nevertheless, if this behaviour is unaccepatable, then set the batch size to 1.

Parameters

ctxHdl - context handle

batchSize - batch size

See Also:

DEFAULT_BATCH_SIZE

setCommitBatch

PROCEDURE

`setCommitBatch(ctxHdl IN ctxType, batchSize IN NUMBER);`
Sets the commit batch size. The commit batch size refers to the number or records inserted after which a commit should follow. Note that if commitBatch is < 1 or the session is in "auto-commit" mode then the XSU does not make any explicit commit's. By default the commit-batch size is 0.

Parameters

ctxHdl - context handle

batchSize - commit batch size

setUpdateColumn

PROCEDURE

`setUpdateColumn(ctxHdl IN ctxType, colName IN VARCHAR2);`
Adds a column to the "update column list". In case of insert, the default is to insert values to all the columns in the table; on the other hand, in case of updates, the default is to only update the columns corresponding to the tags present in the ROW element of the XML document. When the update column list is specified, the columns making up this list alone will get updated or inserted into.

Parameters

ctxHdl - context handle

colName - column to be added to the update column list

clearUpdateColumnList

PROCEDURE

`clearUpdateColumnList(ctxHdl IN ctxType)`
Clears the update column list.

Parameters

ctxHdl - context handle

See Also:

setUpdateColumn

setKeyColumn

PROCEDURE

`setKeyColumn(ctxHdl IN ctxType, colName IN VARCHAR2)`
This methods adds a column to the "key column list". In case of update or delete, it is the columns in the key column list that make up the where clause of the update/delete statement. The key columns list must be specified before updates can be done; yet, it is only optional for delete operations.

Parameters

ctxHdl - context handle

colName - column to be added to the key column list

clearKeyColumnList

PROCEDURE

`clearKeyColumnList(ctxHdl IN ctxType)`
Clears the key column list.

Parameters

ctxHdl - context handle

See Also:

setKeyColumn

insertXML

FUNCTION
`insertXML(ctxHdl IN ctxType, xDoc IN VARCHAR2)`

RETURN
NUMBER

Inserts the XML document into the table specified at the context creation time.

Parameters
ctxHdl - context handle

xDoc - string containing the XML document

Returns:
The number of rows inserted.

insertXML

FUNCTION
`insertXML(ctxHdl IN ctxType, xDoc IN CLOB)`

RETURN
NUMBER

Inserts the XML document into the table specified at the context creation time.

Parameters
ctxHdl - context handle

xDocl - string containing the XML document

Returns
The number of rows inserted.

updateXML

FUNCTION

`updateXML(ctxHdl IN ctxType, xDoc IN VARCHAR2)`

RETURN

NUMBER

Updates the table specified at the context creation time with data from the XML document.

Parameters

ctxHdl - context handle

xDoc - string containing the XML document

Returns:

The number of rows updated.

updateXML

FUNCTION

`updateXML(ctxHdl IN ctxType, xDoc IN CLOB)`

RETURN

NUMBER

Updates the table specified at the context creation time with data from the XML document.

Parameters

ctxHdl - context handle

xDocl - string containing the XML document

Returns

The number of rows updated.

deleteXML

FUNCTION

`deleteXML(ctxHdl IN ctxType, xDoc IN VARCHAR2)`

RETURN

NUMBER

Deletes records specified by data from the XML document, from the table specified at the context creation time.

Parameters

ctxHdl - context handle

xDoc - string containing the XML document

Returns

The number of rows deleted.

deleteXML

FUNCTION

`deleteXML(ctxHdl IN ctxType, xDoc IN CLOB)`

RETURN

NUMBER

Deletes records specified by data from the XML document, from the table specified at the context creation time.

Parameters

ctxHdl - context handle

xDocl - string containing the XML document

Returns

The number of rows deleted.

propagateOriginalException

PROCEDURE

`propagateOriginalException(ctxHdl IN ctxType, flag IN BOOLEAN)`
Tells the XSU that if an exception is raised, and is being thrown, the XSU should throw the very exception raised; rather then, wrapping it with an OracleXMLSQLException.

Parameters

ctxHdl - context handle

flag - propagate original exception? 0-false 1-true

getExceptionContent

PROCEDURE

`getExceptionContent(ctxHdl IN ctxType, errNo OUT NUMBER, errMsg OUT VARCHAR2)`
Via its arguments, this method returns the thrown exception's error code and error message (i.e. sql error code) This is to get around the fact that the jvm throws an exception on top of whatever exception was raised; thus, rendering pl/sql unable to access the original exception.

Parameters

ctxHdl - context handle

errNo - error number

errMsg - error message

Index

A

activeFound() - oracle.xml.async.ResourceManager.activeFound(), 4-32, 4-90, 4-165
addDOMBuilderErrorListener(DOMBuilderErrorListener) - oracle.xml.async.DOMBuilder.addDOMBuilderErrorListener(oracle.xml.async.DOMBuilderErrorListener), 4-6, 4-70, 4-145
addDOMBuilderListener(DOMBuilderListener) - oracle.xml.async.DOMBuilder.addDOMBuilderListener(oracle.xml.async.DOMBuilderListener), 4-6, 4-70, 4-145
addXSLTransformerErrorListener(XSLTransformerErrorListener) - oracle.xml.async.XSLTransformer.addXSLTransformerErrorListener(oracle.xml.async.XSLTransformerErrorListener), 4-35, 4-93, 4-168
addXSLTransformerListener(XSLTransformerListener) - oracle.xml.async.XSLTransformer.addXSLTransformerListener(oracle.xml.async.XSLTransformerListener), 4-35, 4-93, 4-168
ANY - oracle.xml.parser.v2.ElementDecl.ANY, 1-32
appendChild(Node) - oracle.xml.parser.v2.XMLNode.appendChild(org.w3c.dom.Node), 1-106
ASTERISK - oracle.xml.parser.v2.ElementDecl.ASTERISK, 1-32
AttListDecl - oracle.xml.parser.v2.XMLToken.AttListDecl, 1-135
AttName - oracle.xml.parser.v2.XMLToken.AttName, 1-136
ATTRDECL - oracle.xml.parser.v2.XMLNode.ATTRDECL, 1-106
Attribute - oracle.xml.parser.v2.XMLToken.Attribute, 1-136
AttValue - oracle.xml.parser.v2.XMLToken.AttValue, 1-136

C

CANTREAD_XSQL - oracle.xml.xsql.Res.CANTREAD_XSQL, 3-5
CANTREAD_XSQL_MSG - oracle.xml.xsql.Res.CANTREAD_XSQL_MSG, 3-5
CDATA - oracle.xml.parser.v2.AttrDecl.CDATA, 1-6
cDATASection(char[], int, int) - oracle.xml.parser.v2.DefaultXMLDocumentHandler.cDATASection(char[], int, int), 1-11
cDATASection(char[], int, int) - oracle.xml.parser.v2.XMLDocumentHandler.cDATASection(char[], int, int), 1-86
CDSect - oracle.xml.parser.v2.XMLToken.CDSect, 1-136
CGDocument - oracle.xml.classgen.CGDocument, 2-1
CGNode - oracle.xml.classgen.CGNode, 2-2

CharData - oracle.xml.parser.v2.XMLToken.CharData, 1-136
checkNamespace(String, String) - oracle.xml.parser.v2.XMLElement.checkNamespace(java.lang.String, java.lang.String), 1-93
ClassGenerator - oracle.xml.classgen.ClassGenerator, 2-4
ClassGenerator() - oracle.xml.classgen.ClassGenerator.ClassGenerator(), 2-5
CLASSNOTFOUND - oracle.xml.xsql.Res.CLASSNOTFOUND, 3-5
CLASSNOTFOUND_MSG - oracle.xml.xsql.Res.CLASSNOTFOUND_MSG, 3-5
cloneNode(boolean) - oracle.xml.parser.v2.DTD.cloneNode(boolean), 1-24
cloneNode(boolean) - oracle.xml.parser.v2.XMLAttr.cloneNode(boolean), 1-57
cloneNode(boolean) - oracle.xml.parser.v2.XMLDocument.cloneNode(boolean), 1-71
cloneNode(boolean) - oracle.xml.parser.v2.XMLElement.cloneNode(boolean), 1-93
cloneNode(boolean) - oracle.xml.parser.v2.XMLNode.cloneNode(boolean), 1-107
close() - oracle.xml.sql.dml.OracleXMLSave.close(), 9-4
close() - oracle.xml.sql.query.OracleXMLQuery.close(), 9-12
collectTimingInfo(boolean) - oracle.xml.sql.dml.OracleXMLSave.collectTimingInfo(boolean), 9-4
COMMA - oracle.xml.parser.v2.ElementDecl.COMMA, 1-32
Comment - oracle.xml.parser.v2.XMLToken.Comment, 1-136
comment(String) - oracle.xml.parser.v2.DefaultXMLDocumentHandler.comment(java.lang.String), 1-11
comment(String) - oracle.xml.parser.v2.XMLDocumentHandler.comment(java.lang.String), 1-86
CONN_FILE - oracle.xml.xsql.Res.CONN_FILE, 3-5
CONN_FILE_MSG - oracle.xml.xsql.Res.CONN_FILE_MSG, 3-5
createAttribute(String) - oracle.xml.parser.v2.XMLDocument.createAttribute(java.lang.String), 1-71
createAttribute(String, String) - oracle.xml.parser.v2.NodeFactory.createAttribute(java.lang.String, java.lang.String), 1-36
createCDATASection(String) - oracle.xml.parser.v2.NodeFactory.createCDATASection(java.lang.String), 1-36
createCDATASection(String) - oracle.xml.parser.v2.XMLDocument.createCDATASection(java.lang.String), 1-72
createComment(String) - oracle.xml.parser.v2.NodeFactory.createComment(java.lang.String), 1-37
createComment(String) - oracle.xml.parser.v2.XMLDocument.createComment(java.lang.String), 1-72
createDocument() - oracle.xml.parser.v2.NodeFactory.createDocument(), 1-37
createDocumentFragment() - oracle.xml.parser.v2.XMLDocument.createDocumentFragment(), 1-73
createElement(String) - oracle.xml.parser.v2.NodeFactory.createElement(java.lang.String), 1-37
createElement(String) - oracle.xml.parser.v2.XMLDocument.createElement(java.lang.String), 1-73
createEntityReference(String) - oracle.xml.parser.v2.XMLDocument.createEntityReference(java.lang.String), 1-74
createNestedRequest(URL, Dictionary) - oracle.xml.xsql.XSQLPageRequest.createNested

Request(java.net.URL, java.util.Dictionary), 3-24
createNestedRequest(URL, Dictionary) - oracle.xml.xsql.XSQLServletPageRequest.createNestedRequest(java.net.URL, java.util.Dictionary), 3-48
createProcessingInstruction(String, String) - oracle.xml.parser.v2.NodeFactory.createProcessingInstruction(java.lang.String, java.lang.String), 1-38
createProcessingInstruction(String, String) - oracle.xml.parser.v2.XMLDocument.createProcessingInstruction(java.lang.String, java.lang.String), 1-74
createTextNode(String) - oracle.xml.parser.v2.NodeFactory.createTextNode(java.lang.String), 1-38
createTextNode(String) - oracle.xml.parser.v2.XMLDocument.createTextNode(java.lang.String), 1-75
createTimestamp(String) - oracle.xml.sql.dml.OracleXMLSave.createTimestamp(java.lang.String), 9-4
createXMLTable(Connection, String) - oracle.xml.transviewer.DBAccess.createXMLTable(java.sql.Connection, java.lang.String), 4-106, 4-181

D

DATE_FORMAT - oracle.xml.sql.dml.OracleXMLSave.DATE_FORMAT, 9-4
DBAccess - oracle.xml.transviewer.DBAccess, 4-105, 4-180
DBAccess() - oracle.xml.transviewer.DBAccess.DBAccess(), 4-106, 4-181
debugPrintToFile(String, String) - oracle.xml.xsql.XSQLDiagnostic.debugPrintToFile(java.lang.String, java.lang.String), 3-20
DEFAULT - oracle.xml.parser.v2.AttrDecl.DEFAULT, 1-6
DEFAULT_BATCH_SIZE - oracle.xml.sql.dml.OracleXMLSave.DEFAULT_BATCH_SIZE, 9-4
DefaultXMLDocumentHandler - oracle.xml.parser.v2.DefaultXMLDocumentHandler, 1-9
DefaultXMLDocumentHandler() - oracle.xml.parser.v2.DefaultXMLDocumentHandler.DefaultXMLDocumentHandler(), 1-10
deleteXML(Document) - oracle.xml.sql.dml.OracleXMLSave.deleteXML(org.w3c.dom.Document), 9-4
deleteXML(InputStream) - oracle.xml.sql.dml.OracleXMLSave.deleteXML(java.io.InputStream), 9-4
deleteXML(String) - oracle.xml.sql.dml.OracleXMLSave.deleteXML(java.lang.String), 9-4
deleteXML(URL) - oracle.xml.sql.dml.OracleXMLSave.deleteXML(java.net.URL), 9-5
deleteXMLName(Connection, String, String) - oracle.xml.transviewer.DBAccess.deleteXMLName(java.sql.Connection, java.lang.String, java.lang.String), 4-107, 4-182
DictionaryOfParamsAsXMLDocument(Dictionary) - oracle.xml.xsql.XSQLUtil.DictionaryOfParamsAsXMLDocument(java.util.Dictionary), 3-54
doGet(HttpServletRequest, HttpServletResponse) - oracle.xml.xsql.XSQLServlet.doGet(oracle.xml.xsql.HttpServletRequest, oracle.xml.xsql.HttpServletResponse), 3-45
DOMBuilder - oracle.xml.async.DOMBuilder, 4-3, 4-68, 4-143
DOMBuilder() - oracle.xml.async.DOMBuilder.DOMBuilder(), 4-6, 4-70, 4-145
DOMBuilder(int) - oracle.xml.async.DOMBuilder.DOMBuilder(int), 4-6, 4-70, 4-145
DOMBuilderBeanInfo - oracle.xml.async.DOMBuilderBeanInfo, 4-18, 4-80, 4-155
DOMBuilderBeanInfo() - oracle.xml.async.DOMBuilderBeanInfo.DOMBuilderBeanInfo(), 4-19, 4-81, 4-156
domBuilderError(DOMBuilderEvent) -

oracle.xml.async.DOMBuilderListener.domBuil derError(oracle.xml.async.DOMBuilderEvent), 4-29, 4-87, 4-162
domBuilderErrorCalled(DOMBuilderErrorEvent) - oracle.xml.async.DOMBuilderErrorListener.do mBuilderErrorCalled(oracle.xml.async.DOMBui lderErrorEvent), 4-26, 4-84, 4-159
DOMBuilderErrorEvent - oracle.xml.async.DOMBuilderErrorEvent, 4-2 0, 4-82, 4-157
DOMBuilderErrorEvent(Object, Exception) - oracle.xml.async.DOMBuilderErrorEvent.DOM BuilderErrorEvent(java.lang.Object, java.lang.Exception), 4-21, 4-83, 4-158
DOMBuilderErrorListener - oracle.xml.async.DOMBuilderErrorListener, 4 -26, 4-84, 4-159
DOMBuilderEvent - oracle.xml.async.DOMBuilderEvent, 4-27, 4-85, 4-160
DOMBuilderEvent(Object, int) - oracle.xml.async.DOMBuilderEvent.DOMBuild erEvent(java.lang.Object, int), 4-28, 4-86, 4-161
DOMBuilderListener - oracle.xml.async.DOMBuilderListener, 4-29, 4-87, 4-162
domBuilderOver(DOMBuilderEvent) - oracle.xml.async.DOMBuilderListener.domBuil derOver(oracle.xml.async.DOMBuilderEvent), 4-30, 4-87, 4-162
domBuilderStarted(DOMBuilderEvent) - oracle.xml.async.DOMBuilderListener.domBuil derStarted(oracle.xml.async.DOMBuilderEvent) , 4-30, 4-88, 4-163
DOMParser() - oracle.xml.parser.v2.DOMParser.DOMParser(), 1-17
doPost(HttpServletRequest, HttpServletResponse) - oracle.xml.xsql.XSQLServlet.doPost(oracle.xml. xsql.HttpServletRequest, oracle.xml.xsql.HttpServletResponse), 3-46
dropXMLTable(Connection, String) - oracle.xml.transviewer.DBAccess.dropXMLTabl e(java.sql.Connection, java.lang.String), 4-107, 4-182
DTD - oracle.xml.parser.v2.DTD, 1-22
DTD - oracle.xml.sql.query.OracleXMLQuery.DTD, 9 -11
DTDName - oracle.xml.parser.v2.XMLToken.DTDName, 1 -136

E

ElemDeclName - oracle.xml.parser.v2.XMLToken.ElemDeclNam e, 1-136
ELEMENT - oracle.xml.parser.v2.ElementDecl.ELEMENT, 1-32
ElementDecl - oracle.xml.parser.v2.ElementDecl, 1-30
ELEMENTDECL - oracle.xml.parser.v2.XMLNode.ELEMENTDEC L, 1-106
elementdecl - oracle.xml.parser.v2.XMLToken.elementdecl, 1-137
ELEMENTS - oracle.xml.parser.v2.ElementDecl.ELEMENTS, 1-32
EMPTY - oracle.xml.parser.v2.ElementDecl.EMPTY, 1-3 3
EmptyElemTag - oracle.xml.parser.v2.XMLToken.EmptyElemTa g, 1-137
endDoctype() - oracle.xml.parser.v2.DefaultXMLDocumentHan dler.endDoctype(), 1-12
endDoctype() - oracle.xml.parser.v2.XMLDocumentHandler.en dDoctype(), 1-87
endElement(NSName) - oracle.xml.parser.v2.DefaultXMLDocumentHan dler.endElement(oracle.xml.parser.v2.NSName) , 1-12
endElement(NSName) - oracle.xml.parser.v2.XMLDocumentHandler.en

dElement(oracle.xml.parser.v2.NSName), 1-87
ENTITIES -
oracle.xml.parser.v2.AttrDecl.ENTITIES, 1-6
ENTITY -
oracle.xml.parser.v2.AttrDecl.ENTITY, 1-6
EntityDecl -
oracle.xml.parser.v2.XMLToken.EntityDecl, 1-137
EntityDeclName -
oracle.xml.parser.v2.XMLToken.EntityDeclName, 1-137
EntityValue -
oracle.xml.parser.v2.XMLToken.EntityValue, 1-137
ENUMERATION -
oracle.xml.parser.v2.AttrDecl.ENUMERATION, 1-7
ERR_OUTPUT -
oracle.xml.xsql.Res.ERR_OUTPUT, 3-6
ERR_OUTPUT_MSG -
oracle.xml.xsql.Res.ERR_OUTPUT_MSG, 3-6
ERROR -
oracle.xml.parser.v2.XMLParseException.ERROR, 1-118
ERROR_TAG -
oracle.xml.sql.query.OracleXMLQuery.ERROR_TAG, 9-11
ERRORINCLUDING -
oracle.xml.xsql.Res.ERRORINCLUDING, 3-6
ERRORINCLUDING_MSG -
oracle.xml.xsql.Res.ERRORINCLUDING_MSG, 3-6
ERRORLOADINGURL -
oracle.xml.xsql.Res.ERRORLOADINGURL, 3-6
ERRORLOADINGURL_MSG -
oracle.xml.xsql.Res.ERRORLOADINGURL_MSG, 3-6
ERRORREADINGPARAM -
oracle.xml.xsql.Res.ERRORREADINGPARAM, 3-6
ERRORREADINGPARAM_MSG -
oracle.xml.xsql.Res.ERRORREADINGPARAM_MSG, 3-6
ETag -
oracle.xml.parser.v2.XMLToken.ETag, 1-137
ETagName -
oracle.xml.parser.v2.XMLToken.ETagName, 1-137
expectedElements(Element) -
oracle.xml.parser.v2.ElementDecl.expectedElements(org.w3c.dom.Element), 1-33
expectedElements(Element) -
oracle.xml.parser.v2.XMLDocument.expectedElements(org.w3c.dom.Element), 1-75
ExternalID -
oracle.xml.parser.v2.XMLToken.ExternalID, 1-137

F

FATAL_ERROR -
oracle.xml.parser.v2.XMLParseException.FATAL_ERROR, 1-118
FATAL_SHEETPOOL -
oracle.xml.xsql.Res.FATAL_SHEETPOOL, 3-6
FATAL_SHEETPOOL_MSG -
oracle.xml.xsql.Res.FATAL_SHEETPOOL_MSG, 3-6
findAttrDecl(String) -
oracle.xml.parser.v2.ElementDecl.findAttrDecl(java.lang.String), 1-33
findElementDecl(String) -
oracle.xml.parser.v2.DTD.findElementDecl(java.lang.String), 1-25
findEntity(String, boolean) -
oracle.xml.parser.v2.DTD.findEntity(java.lang.String, boolean), 1-25
findNotation(String) -
oracle.xml.parser.v2.DTD.findNotation(java.lang.String), 1-25
FIXED - oracle.xml.parser.v2.AttrDecl.FIXED, 1-7
fontGet(AttributeSet) -
oracle.xml.srcviewer.XMLSourceView.fontGet(javax.swing.text.AttributeSet), 4-53, 4-128
fontSet(MutableAttributeSet, Font) -
oracle.xml.srcviewer.XMLSourceView.fontSet(javax.swing.text.MutableAttributeSet, java.awt.Font), 4-54, 4-128
format(int, String) - oracle.xml.xsql.Res.format(int,

java.lang.String), 3-11

G

generate(DTD, String) - oracle.xml.classgen.ClassGenerator.generate(oracle.xml.classgen.DTD, java.lang.String), 2-5
getAttrDecls() - oracle.xml.parser.v2.ElementDecl.getAttrDecls(), 1-34
getAttribute(String) - oracle.xml.parser.v2.XMLElement.getAttribute(java.lang.String), 1-94
getAttributeNameFont() - oracle.xml.srcviewer.XMLSourceView.getAttributeNameFont(), 4-54, 4-129
getAttributeNameForeground() - oracle.xml.srcviewer.XMLSourceView.getAttributeNameForeground(), 4-54, 4-129
getAttributeNode(String) - oracle.xml.parser.v2.XMLElement.getAttributeNode(java.lang.String), 1-94
getAttributes() - oracle.xml.parser.v2.XMLElement.getAttributes(), 1-95
getAttributes() - oracle.xml.parser.v2.XMLNode.getAttributes(), 1-107
getAttributeValueFont() - oracle.xml.srcviewer.XMLSourceView.getAttributeValueFont(), 4-54, 4-129
getAttributeValueForeground() - oracle.xml.srcviewer.XMLSourceView.getAttributeValueForeground(), 4-54, 4-129
getAttrPresence() - oracle.xml.parser.v2.AttrDecl.getAttrPresence(), 1-8
getAttrType() - oracle.xml.parser.v2.AttrDecl.getAttrType(), 1-8
getBackground() - oracle.xml.srcviewer.XMLSourceView.getBackground(), 4-55, 4-129
getCDATAFont() - oracle.xml.srcviewer.XMLSourceView.getCDATAFont(), 4-55, 4-130
getCDATAForeground() - oracle.xml.srcviewer.XMLSourceView.getCDATAForeground(), 4-55, 4-130
getChildNodes() - oracle.xml.parser.v2.DTD.getChildNodes(), 1-25
getChildNodes() - oracle.xml.parser.v2.XMLNode.getChildNodes(), 1-108
getChildrenByTagName(String) - oracle.xml.parser.v2.XMLElement.getChildrenByTagName(java.lang.String), 1-95
getChildrenByTagName(String, String) - oracle.xml.parser.v2.XMLElement.getChildrenByTagName(java.lang.String, java.lang.String), 1-95
getColumnNumber(int) - oracle.xml.parser.v2.XMLParseException.getColumnNumber(int), 1-119
getCommentDataFont() - oracle.xml.srcviewer.XMLSourceView.getCommentDataFont(), 4-55, 4-130
getCommentDataForeground() - oracle.xml.srcviewer.XMLSourceView.getCommentDataForeground(), 4-55, 4-130
getConnectionName() - oracle.xml.xsql.XSQLPageRequest.getConnectionName(), 3-24
getConnectionName() - oracle.xml.xsql.XSQLPageRequestImpl.getConnectionName(), 3-31
getContentElements() - oracle.xml.parser.v2.ElementDecl.getContentElements(), 1-34
getContentType() - oracle.xml.parser.v2.ElementDecl.getContentType(), 1-34
getCookie(String) - oracle.xml.xsql.XSQLServletPageRequest.getCookie(java.lang.String), 3-49
getDefaultValue() - oracle.xml.parser.v2.AttrDecl.getDefaultValue(), 1-8
getDoctype() -

oracle.xml.async.DOMBuilder.getDoctype(), 4-70, 4-146
getDoctype() - oracle.xml.parser.v2.DOMParser.getDoctype(), 1-17
getDoctype() - oracle.xml.parser.v2.XMLDocument.getDoctype(), 1-75
getDocument() - oracle.xml.async.DOMBuilder.getDocument(), 4-7, 4-71, 4-146
getDocument() - oracle.xml.parser.v2.DOMParser.getDocument(), 1-17
getDocumentElement() - oracle.xml.parser.v2.XMLDocument.getDocumentElement(), 1-76
getEditedText() - oracle.xml.srcviewer.XMLSourceView.getEditedText(), 4-56, 4-130
getElementDecls() - oracle.xml.parser.v2.DTD.getElementDecls(), 1-26
getElementsByTagName(String) - oracle.xml.parser.v2.XMLDocument.getElementsByTagName(java.lang.String), 1-76
getElementsByTagName(String) - oracle.xml.parser.v2.XMLElement.getElementsByTagName(java.lang.String), 1-96
getElementsByTagName(String, String) - oracle.xml.parser.v2.XMLElement.getElementsByTagName(java.lang.String, java.lang.String), 1-96
getEncoding() - oracle.xml.parser.v2.XMLDocument.getEncoding(), 1-76
getEntities() - oracle.xml.parser.v2.DTD.getEntities(), 1-26
getEnumerationValues() - oracle.xml.parser.v2.AttrDecl.getEnumerationValues(), 1-8
getErrorCode() - oracle.xml.sql.OracleXMLSQLException.getErrorCode(), 9-20
getErrorWriter() - oracle.xml.xsql.XSQLPageRequest.getErrorWriter(), 3-25
getErrorWriter() - oracle.xml.xsql.XSQLPageRequestImpl.getErrorWriter(), 3-31
getException() - oracle.xml.async.DOMBuilderErrorEvent.getException(), 4-25, 4-83, 4-158
getException() - oracle.xml.async.XSLTransformerErrorEvent.getException(), 4-43, 4-99, 4-174
getException(int) - oracle.xml.parser.v2.XMLParseException.getException(int), 1-119
getExpandedName() - oracle.xml.parser.v2.NSName.getExpandedName(), 1-39
getExpandedName() - oracle.xml.parser.v2.XMLAttr.getExpandedName(), 1-57
getExpandedName() - oracle.xml.parser.v2.XMLElement.getExpandedName(), 1-96
getExpandedName(int) - oracle.xml.parser.v2.SAXAttrList.getExpandedName(int), 1-45
getFirstChild() - oracle.xml.parser.v2.XMLNode.getFirstChild(), 1-108
getHttpServletRequest() - oracle.xml.xsql.XSQLServletPageRequest.getHttpServletRequest(), 3-49
getHttpServletResponse() - oracle.xml.xsql.XSQLServletPageRequest.getHttpServletResponse(), 3-49
getIcon(int) - oracle.xml.async.DOMBuilderBeanInfo.getIcon(int), 4-19, 4-81, 4-156
getIcon(int) - oracle.xml.async.XSLTransformerBeanInfo.getIcon(int), 4-41, 4-97, 4-172
getIcon(int) - oracle.xml.srcviewer.XMLSourceViewBeanInfo.getIcon(int), 4-66, 4-140
getIcon(int) -

oracle.xml.transviewer.XMLTransformPanelBeanInfo.getIcon(int), 4-115, 4-190
getIcon(int) - oracle.xml.treeviewer.XMLTreeViewBeanInfo.getIcon(int), 4-123, 4-198, 4-205
getID() - oracle.xml.async.DOMBuilderEvent.getID(), 4-28, 4-86, 4-161
getId() - oracle.xml.async.DOMBuilder.getId(), 4-7, 4-71, 4-146
getID() - oracle.xml.async.XSLTransformerEvent.getID(), 4-46, 4-102, 4-177
getId() - oracle.xml.async.XSLTransformer.getId(), 4-36, 4-93, 4-168
getImplementation() - oracle.xml.parser.v2.XMLDocument.getImplementation(), 1-77
getInternalObj() - oracle.xml.sql.query.OracleXMLQuery.getInternalObj(), 9-12
getJDBCConnection() - oracle.xml.xsql.XSQLPageRequest.getJDBCConnection(), 3-25
getJDBCConnection() - oracle.xml.xsql.XSQLPageRequestImpl.getJDBCConnection(), 3-32
getJTextPane() - oracle.xml.srcviewer.XMLSourceView.getJTextPane(), 4-56, 4-131
getLastChild() - oracle.xml.parser.v2.XMLNode.getLastChild(), 1-108
getLength() - oracle.xml.parser.v2.SAXAttrList.getLength(), 1-45
getLineNumber(int) - oracle.xml.parser.v2.XMLParseException.getLineNumber(int), 1-119
getLocalName() - oracle.xml.parser.v2.NSName.getLocalName(), 1-39
getLocalName() - oracle.xml.parser.v2.XMLAttr.getLocalName(), 1-57
getLocalName() - oracle.xml.parser.v2.XMLElement.getLocalName(), 1-97
getLocalName(int) - oracle.xml.parser.v2.SAXAttrList.getLocalName(int), 1-45
getMessage() - oracle.xml.async.DOMBuilderErrorEvent.getMessage(), 4-25, 4-83, 4-158
getMessage() - oracle.xml.async.XSLTransformerErrorEvent.getMessage(), 4-43, 4-99, 4-174
getMessage(int) - oracle.xml.parser.v2.XMLParseException.getMessage(int), 1-119
getMessageType(int) - oracle.xml.parser.v2.XMLParseException.getMessageType(int), 1-120
getMinimumSize() - oracle.xml.srcviewer.XMLSourceView.getMinimumSize(), 4-56, 4-131
getName() - oracle.xml.parser.v2.DTD.getName(), 1-27
getName() - oracle.xml.parser.v2.XMLAttr.getName(), 1-58
getName(int) - oracle.xml.parser.v2.SAXAttrList.getName(int), 1-46
getNameSize() - oracle.xml.transviewer.DBAccess.getNameSize(), 4-107, 4-182
getNamespace() - oracle.xml.parser.v2.NSName.getNamespace(), 1-39
getNamespace() - oracle.xml.parser.v2.XMLAttr.getNamespace(), 1-58
getNamespace() - oracle.xml.parser.v2.XMLElement.getNamespace(), 1-97
getNamespace(int) - oracle.xml.parser.v2.SAXAttrList.getNamespace(int), 1-46
getNextSibling() -

oracle.xml.parser.v2.XMLNode.getNextSibling( ), 1-109
getNodeAtOffset(int) - oracle.xml.srcviewer.XMLSourceView.getNode AtOffset(int), 4-56, 4-131
getNodeName() - oracle.xml.parser.v2.XMLNode.getNodeName() , 1-109
getNodeType() - oracle.xml.parser.v2.XMLNode.getNodeType(), 1-109
getNodeValue() - oracle.xml.parser.v2.XMLAttr.getNodeValue(), 1-58
getNodeValue() - oracle.xml.parser.v2.XMLNode.getNodeValue() , 1-109
getNodeValue() - oracle.xml.parser.v2.XMLText.getNodeValue(), 1-132
getNotations() - oracle.xml.parser.v2.DTD.getNotations(), 1-27
getNumMessages() - oracle.xml.parser.v2.XMLParseException.getNu mMessages(), 1-120
getOwnerDocument() - oracle.xml.parser.v2.XMLDocument.getOwner Document(), 1-77
getOwnerDocument() - oracle.xml.parser.v2.XMLNode.getOwnerDocu ment(), 1-110
getPageEncoding() - oracle.xml.xsql.XSQLPageRequest.getPageEnco ding(), 3-25
getPageEncoding() - oracle.xml.xsql.XSQLPageRequestImpl.getPage Encoding(), 3-32
getParameter(String) - oracle.xml.xsql.XSQLPageRequest.getParameter (java.lang.String), 3-25
getParameter(String) - oracle.xml.xsql.XSQLPageRequestImpl.getPara meter(java.lang.String), 3-32
getParameter(String) - oracle.xml.xsql.XSQLServletPageRequest.getPar ameter(java.lang.String), 3-49
getParentException() - oracle.xml.sql.OracleXMLSQLException.getPar entException(), 9-20
getParentNode() - oracle.xml.parser.v2.XMLAttr.getParentNode(), 1-59
getParentNode() - oracle.xml.parser.v2.XMLDocumentFragment.g etParentNode(), 1-84
getParentNode() - oracle.xml.parser.v2.XMLNode.getParentNode( ), 1-110
getParseTree() - oracle.xml.parser.v2.ElementDecl.getParseTree( ), 1-34
getPCDATAFont() - oracle.xml.srcviewer.XMLSourceView.getPCD ATAFont(), 4-57, 4-131
getPCDATAForeground() - oracle.xml.srcviewer.XMLSourceView.getPCD ATAForeground(), 4-57, 4-132
getPIDataFont() - oracle.xml.srcviewer.XMLSourceView.getPIDat aFont(), 4-57, 4-132
getPIDataForeground() - oracle.xml.srcviewer.XMLSourceView.getPIDat aForeground(), 4-57, 4-132
getPINameFont() - oracle.xml.srcviewer.XMLSourceView.getPINa meFont(), 4-57, 4-132
getPINameForeground() - oracle.xml.srcviewer.XMLSourceView.getPINa meForeground(), 4-58, 4-132
getPostedDocument() - oracle.xml.xsql.XSQLPageRequest.getPostedDo cument(), 3-25
getPostedDocument() - oracle.xml.xsql.XSQLPageRequestImpl.getPoste dDocument(), 3-32
getPostedDocument() - oracle.xml.xsql.XSQLServletPageRequest.getPo stedDocument(), 3-49
getPreferredSize() - oracle.xml.treeviewer.XMLTreeView.getPreferr

edSize(), 4-120, 4-195, 4-202
getPrefix() -
oracle.xml.parser.v2.NSName.getPrefix(), 1-39
getPrefix() -
oracle.xml.parser.v2.XMLAttr.getPrefix(), 1-59
getPrefix() -
oracle.xml.parser.v2.XMLElement.getPrefix(), 1-97
getPrefix(int) -
oracle.xml.parser.v2.SAXAttrList.getPrefix(int), 1-47
getPreviousSibling() -
oracle.xml.parser.v2.XMLNode.getPreviousSibling(), 1-111
getPropertyDescriptors() -
oracle.xml.async.DOMBuilderBeanInfo.getPropertyDescriptors(), 4-19, 4-81, 4-156
getPropertyDescriptors() -
oracle.xml.async.XSLTransformerBeanInfo.getPropertyDescriptors(), 4-41, 4-97, 4-172
getPropertyDescriptors() -
oracle.xml.srcviewer.XMLSourceViewBeanInfo.getPropertyDescriptors(), 4-66, 4-141
getPropertyDescriptors() -
oracle.xml.transviewer.XMLTransformPanelBeanInfo.getPropertyDescriptors(), 4-115, 4-190
getPropertyDescriptors() -
oracle.xml.treeviewer.XMLTreeViewBeanInfo.getPropertyDescriptors(), 4-123, 4-198, 4-205
getPublicId() -
oracle.xml.parser.v2.DTD.getPublicId(), 1-27
getPublicId(int) -
oracle.xml.parser.v2.XMLParseException.getPublicId(int), 1-120
getQualifiedName() -
oracle.xml.parser.v2.NSName.getQualifiedName(), 1-40
getQualifiedName() -
oracle.xml.parser.v2.XMLAttr.getQualifiedName(), 1-59
getQualifiedName() -
oracle.xml.parser.v2.XMLElement.getQualifiedName(), 1-97
getQualifiedName(int) -
oracle.xml.parser.v2.SAXAttrList.getQualifiedName(int), 1-47
getReleaseVersion() -
oracle.xml.async.DOMBuilder.getReleaseVersion(), 4-7, 4-71, 4-146
getReleaseVersion() -
oracle.xml.parser.v2.XMLParser.getReleaseVersion(), 1-122
getRequestParamsAsXMLDocument() -
oracle.xml.xsql.XSQLPageRequest.getRequestParamsAsXMLDocument(), 3-25
getRequestParamsAsXMLDocument() -
oracle.xml.xsql.XSQLPageRequestImpl.getRequestParamsAsXMLDocument(), 3-32
getRequestParamsAsXMLDocument() -
oracle.xml.xsql.XSQLServletPageRequest.getRequestParamsAsXMLDocument(), 3-49
getRequestType() -
oracle.xml.xsql.XSQLPageRequest.getRequestType(), 3-25
getRequestType() -
oracle.xml.xsql.XSQLServletPageRequest.getRequestType(), 3-49
getResource() -
oracle.xml.async.ResourceManager.getResource(), 4-32, 4-90, 4-165
getResult() -
oracle.xml.async.DOMBuilder.getResult(), 4-71, 4-146
getResult() -
oracle.xml.async.XSLTransformer.getResult(), 4-36, 4-93, 4-168
getServletInfo() -
oracle.xml.xsql.XSQLServlet.getServletInfo(), 3-46
getSourceDocumentURI() -
oracle.xml.xsql.XSQLPageRequest.getSourceDocumentURI(), 3-25
getSourceDocumentURI() -
oracle.xml.xsql.XSQLPageRequestImpl.getSourceDocumentURI(), 3-32
getSpecified() -
oracle.xml.parser.v2.XMLAttr.getSpecified(), 1-60
getStandalone() -
oracle.xml.parser.v2.XMLDocument.getStandal

one(), 1-77
getString(int) -
oracle.xml.xsql.Res.getString(int), 3-11
getStructVal(Node, OracleColumnName) -
oracle.xml.sql.dml.OracleXMLSave.getStructVa
l(org.w3c.dom.Node,
oracle.xml.sql.name.OracleColumnName), 9-5
getStylesheetParameter(String) -
oracle.xml.xsql.XSQLPageRequest.getStylesheet
Parameter(java.lang.String), 3-26
getStylesheetParameter(String) -
oracle.xml.xsql.XSQLPageRequestImpl.getStyle
sheetParameter(java.lang.String), 3-33
getStylesheetParameters() -
oracle.xml.xsql.XSQLPageRequest.getStylesheet
Parameters(), 3-26
getStylesheetParameters() -
oracle.xml.xsql.XSQLPageRequestImpl.getStyle
sheetParameters(), 3-33
getStylesheetURI() -
oracle.xml.xsql.XSQLPageRequest.getStylesheet
URI(), 3-26
getStylesheetURI() -
oracle.xml.xsql.XSQLPageRequestImpl.getStyle
sheetURI(), 3-33
getSymbolFont() -
oracle.xml.srcviewer.XMLSourceView.getSymb
olFont(), 4-58, 4-133
getSymbolForeground() -
oracle.xml.srcviewer.XMLSourceView.getSymb
olForeground(), 4-58, 4-133
getSystemId() -
oracle.xml.parser.v2.DTD.getSystemId(), 1-27
getSystemId(int) -
oracle.xml.parser.v2.XMLParseException.getSy
stemId(int), 1-120
getTagFont() -
oracle.xml.srcviewer.XMLSourceView.getTagFo
nt(), 4-58, 4-133
getTagForeground() -
oracle.xml.srcviewer.XMLSourceView.getTagFo
reground(), 4-58, 4-133
getTagName() -
oracle.xml.parser.v2.XMLElement.getTagName
(), 1-98
getTarget() -
oracle.xml.parser.v2.XMLPI.getTarget(), 1-129
getText() -
oracle.xml.srcviewer.XMLSourceView.getText()
, 4-59, 4-133
getType(int) -
oracle.xml.parser.v2.SAXAttrList.getType(int),
1-47
getType(String) -
oracle.xml.parser.v2.SAXAttrList.getType(java.l
ang.String), 1-47
getUserAgent() -
oracle.xml.xsql.XSQLPageRequest.getUserAgen
t(), 3-26
getUserAgent() -
oracle.xml.xsql.XSQLPageRequestImpl.getUser
Agent(), 3-33
getUserAgent() -
oracle.xml.xsql.XSQLServletPageRequest.getUs
erAgent(), 3-50
getValidationMode() -
oracle.xml.async.DOMBuilder.getValidationMo
de(), 4-8, 4-71, 4-146
getValidationMode() -
oracle.xml.parser.v2.XMLParser.getValidation
Mode(), 1-122
getValue() -
oracle.xml.parser.v2.XMLAttr.getValue(), 1-60
getValue(int) -
oracle.xml.parser.v2.SAXAttrList.getValue(int),
1-48
getValue(String) -
oracle.xml.parser.v2.SAXAttrList.getValue(java.
lang.String), 1-48
getVersion() -
oracle.xml.parser.v2.XMLDocument.getVersion
(), 1-77
getWriter() -
oracle.xml.xsql.XSQLPageRequest.getWriter(),
3-26
getWriter() -
oracle.xml.xsql.XSQLPageRequestImpl.getWrit
er(), 3-33
getXMLData(Connection, String, String) -
oracle.xml.transviewer.DBAccess.getXMLData(j

ava.sql.Connection, java.lang.String, java.lang.String), 4-107, 4-182
getXMLDOM() - oracle.xml.sql.query.OracleXMLQuery.getXMLDOM(), 9-12
getXMLDOM(Node, int) - oracle.xml.sql.query.OracleXMLQuery.getXMLDOM(oracle.xml.sql.query.Node, int), 9-12
getXMLErrorString() - oracle.xml.sql.OracleXMLSQLException.getXMLErrorString(), 9-20
getXMLMetaData(int, boolean) - oracle.xml.sql.query.OracleXMLQuery.getXMLMetaData(int, boolean), 9-13
getXMLMetaData(int, boolean, OracleXMLDocGen) - oracle.xml.sql.query.OracleXMLQuery.getXMLMetaData(int, boolean, oracle.xml.sql.docgen.OracleXMLDocGen), 9-13
getXMLNames(Connection, String) - oracle.xml.transviewer.DBAccess.getXMLNames(java.sql.Connection, java.lang.String), 4-108, 4-183
getXMLSQLErrorString() - oracle.xml.sql.OracleXMLSQLException.getXMLSQLErrorString(), 9-20
getXMLString() - oracle.xml.sql.query.OracleXMLQuery.getXMLString(), 9-13
getXMLString(boolean) - oracle.xml.sql.query.OracleXMLQuery.getXMLString(boolean), 9-13
getXMLString(Node) - oracle.xml.sql.query.OracleXMLQuery.getXMLString(oracle.xml.sql.query.Node), 9-13
getXMLString(Node, int) - oracle.xml.sql.query.OracleXMLQuery.getXMLString(oracle.xml.sql.query.Node, int), 9-13
getXMLTableNames(Connection, String) - oracle.xml.transviewer.DBAccess.getXMLTableNames(java.sql.Connection, java.lang.String), 4-108, 4-183
getXSQLConnection() - oracle.xml.xsql.XSQLPageRequest.getXSQLConnection(), 3-26
getXSQLConnection() - oracle.xml.xsql.XSQLPageRequestImpl.getXSQLConnection(), 3-34

H

handleAction(Node) - oracle.xml.xsql.XSQLActionHandler.handleAction(oracle.xml.xsql.Node), 3-14
hasChildNodes() - oracle.xml.parser.v2.DTD.hasChildNodes(), 1-28
hasChildNodes() - oracle.xml.parser.v2.XMLNode.hasChildNodes(), 1-111
HttpRequestAsXMLDocument(HttpServletRequest, String) - oracle.xml.xsql.XSQLHttpUtil.HttpRequestAsXMLDocument(oracle.xml.xsql.HttpServletRequest, java.lang.String), 3-22

I

ID - oracle.xml.parser.v2.AttrDecl.ID, 1-7
IDREF - oracle.xml.parser.v2.AttrDecl.IDREF, 1-7
IDREFS - oracle.xml.parser.v2.AttrDecl.IDREFS, 1-7
ILLFORMEDXMLPARAMVAL - oracle.xml.xsql.Res.ILLFORMEDXMLPARAMVAL, 3-6
ILLFORMEDXMLPARAMVAL_MSG - oracle.xml.xsql.Res.ILLFORMEDXMLPARAMVAL_MSG, 3-6
ILLFORMEDXMLRESOURCE - oracle.xml.xsql.Res.ILLFORMEDXMLRESOURCE, 3-7
ILLFORMEDXMLRESOURCE_MSG - oracle.xml.xsql.Res.ILLFORMEDXMLRESOURCE_MSG, 3-7
IMPLIED - oracle.xml.parser.v2.AttrDecl.IMPLIED, 1-7
init(ServletConfig) - oracle.xml.xsql.XSQLServlet.init(oracle.xml.xsql.ServletConfig), 3-46

init(XSQLPageRequest, Element) -
oracle.xml.xsql.XSQLActionHandlerImpl.init(oracle.xml.xsql.XSQLPageRequest, oracle.xml.xsql.Element), 3-16
init(XSQLPageRequest, Element) -
oracle.xml.xsql.XSQLActionHandler.init(oracle.xml.xsql.XSQLPageRequest, oracle.xml.xsql.Element), 3-14
inJServ() -
oracle.xml.xsql.XSQLServlet.inJServ(), 3-46
insertBefore(Node, Node) -
oracle.xml.parser.v2.XMLNode.insertBefore(org.w3c.dom.Node, org.w3c.dom.Node), 1-111
insertXML(Document) -
oracle.xml.sql.dml.OracleXMLSave.insertXML(org.w3c.dom.Document), 9-5
insertXML(InputStream) -
oracle.xml.sql.dml.OracleXMLSave.insertXML(java.io.InputStream), 9-5
insertXML(String) -
oracle.xml.sql.dml.OracleXMLSave.insertXML(java.lang.String), 9-5
insertXML(URL) -
oracle.xml.sql.dml.OracleXMLSave.insertXML(java.net.URL), 9-5
insertXMLData(Connection, String, String, String) -
oracle.xml.transviewer.DBAccess.insertXMLData(java.sql.Connection, java.lang.String, java.lang.String, java.lang.String), 4-108, 4-183
INSTANTIATIONERR -
oracle.xml.xsql.Res.INSTANTIATIONERR, 3-7
INSTANTIATIONERR_MSG -
oracle.xml.xsql.Res.INSTANTIATIONERR_MSG, 3-7
INVALID_URI -
oracle.xml.xsql.Res.INVALID_URI, 3-7
INVALID_URI_MSG -
oracle.xml.xsql.Res.INVALID_URI_MSG, 3-7
InvalidContentException -
oracle.xml.classgen.InvalidContentException, 2-7
InvalidContentException() -
oracle.xml.classgen.InvalidContentException.InvalidContentException(), 2-8
InvalidContentException(String) -
oracle.xml.classgen.InvalidContentException.InvalidContentException(java.lang.String), 2-8
INVALIDURL -
oracle.xml.xsql.Res.INVALIDURL, 3-7
INVALIDURL_MSG -
oracle.xml.xsql.Res.INVALIDURL_MSG, 3-7
isEditable() -
oracle.xml.srcviewer.XMLSourceView.isEditable(), 4-59, 4-134
isIncludedRequest() -
oracle.xml.xsql.XSQLPageRequestImpl.isIncludedRequest(), 3-34
isIncludedRequest() -
oracle.xml.xsql.XSQLPageRequest.isIncludedRequest(), 3-26
isOracleDriver() -
oracle.xml.xsql.XSQLPageRequestImpl.isOracleDriver(), 3-34
isOracleDriver() -
oracle.xml.xsql.XSQLPageRequest.isOracleDriver(), 3-26
isXMLTable(Connection, String) -
oracle.xml.transviewer.DBAccess.isXMLTable(java.sql.Connection, java.lang.String), 4-109, 4-184

M

main(String[]) -
oracle.xml.parser.v2.oraxsl.main(java.lang.String[]), 1-43
main(String[]) -
oracle.xml.transviewer.XMLTransViewer.main(java.lang.String[]), 4-117, 4-192
main(String[]) -
oracle.xml.xsql.XSQLCommandLine.main(java.lang.String[]), 3-18
MAXROWS_ALL -
oracle.xml.sql.query.OracleXMLQuery.MAXROWS_ALL, 9-11
MAXROWS_DEFAULT -
oracle.xml.sql.query.OracleXMLQuery.MAXROWS_DEFAULT, 9-11
MISSING_ARGS -

oracle.xml.xsql.Res.MISSING_ARGS, 3-7
MISSING_ARGS_MSG - oracle.xml.xsql.Res.MISSING_ARGS_MSG, 3-7
MISSING_ATTR - oracle.xml.xsql.Res.MISSING_ATTR, 3-7
MISSING_ATTR_MSG - oracle.xml.xsql.Res.MISSING_ATTR_MSG, 3-7
MIXED - oracle.xml.parser.v2.ElementDecl.MIXED, 1-33
msg(String) - oracle.xml.xsql.XSQLDiagnostic.msg(java.lang.String), 3-20

N

NAMED_CONN - oracle.xml.xsql.Res.NAMED_CONN, 3-8
NAMED_CONN_MSG - oracle.xml.xsql.Res.NAMED_CONN_MSG, 3-8
newDocument() - oracle.xml.xsql.XSQLParserHelper.newDocument(), 3-39
NMTOKEN - oracle.xml.parser.v2.AttrDecl.NMTOKEN, 1-7
NMTOKENS - oracle.xml.parser.v2.AttrDecl.NMTOKENS, 1-7
NO_CONN - oracle.xml.xsql.Res.NO_CONN, 3-8
NO_CONN_DEF - oracle.xml.xsql.Res.NO_CONN_DEF, 3-8
NO_CONN_DEF_MSG - oracle.xml.xsql.Res.NO_CONN_DEF_MSG, 3-8
NO_CONN_MSG - oracle.xml.xsql.Res.NO_CONN_MSG, 3-8
NO_XSQL_FILE - oracle.xml.xsql.Res.NO_XSQL_FILE, 3-8
NO_XSQL_FILE_MSG - oracle.xml.xsql.Res.NO_XSQL_FILE_MSG, 3-8
NodeFactory - oracle.xml.parser.v2.NodeFactory, 1-35
NodeFactory() - oracle.xml.parser.v2.NodeFactory.NodeFactory(), 1-36
NOFUNCTIONNAME - oracle.xml.xsql.Res.NOFUNCTIONNAME, 3-8
NOFUNCTIONNAME_MSG - oracle.xml.xsql.Res.NOFUNCTIONNAME_MSG, 3-8
NONE - oracle.xml.sql.query.OracleXMLQuery.NONE, 9-11
NOPOSTEDXML - oracle.xml.xsql.Res.NOPOSTEDXML, 3-8
NOPOSTEDXML_MSG - oracle.xml.xsql.Res.NOPOSTEDXML_MSG, 3-8
NOQUERYSUPPLIED - oracle.xml.xsql.Res.NOQUERYSUPPLIED, 3-9
NOQUERYSUPPLIED_MSG - oracle.xml.xsql.Res.NOQUERYSUPPLIED_MSG, 3-9
normalize() - oracle.xml.parser.v2.XMLElement.normalize(), 1-98
NOTANACTIONHANDLER - oracle.xml.xsql.Res.NOTANACTIONHANDLER, 3-9
NOTANACTIONHANDLER_MSG - oracle.xml.xsql.Res.NOTANACTIONHANDLER_MSG, 3-9
NOTATION - oracle.xml.parser.v2.AttrDecl.NOTATION, 1-7
NotationDecl - oracle.xml.parser.v2.XMLToken.NotationDecl, 1-138
NSName - oracle.xml.parser.v2.NSName, 1-38
NSResolver - oracle.xml.parser.v2.NSResolver, 1-41
NULLPARAM - oracle.xml.xsql.Res.NULLPARAM, 3-9
NULLPARAM_MSG - oracle.xml.xsql.Res.NULLPARAM_MSG, 3-9

O

OR - oracle.xml.parser.v2.ElementDecl.OR, 1-33
oracle.xml.async - oracle.xml.async, 4-1
oracle.xml.classgen - oracle.xml.classgen, 2-1
oracle.xml.parser.v2 - oracle.xml.parser.v2, 1-1
OracleXMLQuery(Connection, ResultSet) - oracle.xml.sql.query.OracleXMLQuery.OracleXMLQuery(java.sql.Connection, java.sql.ResultSet), 9-12
OracleXMLQuery(Connection, String) - oracle.xml.sql.query.OracleXMLQuery.OracleXMLQuery(java.sql.Connection, java.lang.String), 9-12
OracleXMLQuery(OracleXMLDataSet) - oracle.xml.sql.query.OracleXMLQuery.OracleXMLQuery(oracle.xml.sql.dataset.OracleXMLDataSet), 9-12
OracleXMLSave(Connection, OracleColumnName[]) - oracle.xml.sql.dml.OracleXMLSave.OracleXMLSave(java.sql.Connection, oracle.xml.sql.name.OracleColumnName[]), 9-4
OracleXMLSave(Connection, String) - oracle.xml.sql.dml.OracleXMLSave.OracleXMLSave(java.sql.Connection, java.lang.String), 9-4
oracle.xml.sql.dml - oracle.xml.sql.dml, 9-1
OracleXMLSQLException(Exception) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.Exception), 9-19
OracleXMLSQLException(Exception, String) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.Exception, java.lang.String), 9-19
OracleXMLSQLException(String) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String), 9-19
OracleXMLSQLException(String, Exception) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String, java.lang.Exception), 9-19
OracleXMLSQLException(String, Exception, String) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String, java.lang.Exception, java.lang.String), 9-20
OracleXMLSQLException(String, int) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String, int), 9-20
OracleXMLSQLException(String, int, String) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String, int, java.lang.String), 9-20
OracleXMLSQLException(String, String) - oracle.xml.sql.OracleXMLSQLException.OracleXMLSQLException(java.lang.String, java.lang.String), 9-20
OracleXMLSQLNoRowsException() - oracle.xml.sql.OracleXMLSQLNoRowsException.OracleXMLSQLNoRowsException(), 9-22
OracleXMLSQLNoRowsException(String) - oracle.xml.sql.OracleXMLSQLNoRowsException.OracleXMLSQLNoRowsException(java.lang.String), 9-22
oracle.xml.treeviewer - oracle.xml.treeviewer, 4-199
oracle.xml.xsql - oracle.xml.xsql, 3-1
oraxsl - oracle.xml.parser.v2.oraxsl, 1-42
oraxsl() - oracle.xml.parser.v2.oraxsl.oraxsl(), 1-43
OWAXMLMALFORMED - oracle.xml.xsql.Res.OWAXMLMALFORMED, 3-9
OWAXMLMALFORMED_MSG - oracle.xml.xsql.Res.OWAXMLMALFORMED_MSG, 3-9

P

parse(InputSource) - oracle.xml.async.DOMBuilder.parse(org.xml.sax.InputSource), 4-8, 4-72, 4-147
parse(InputSource) - oracle.xml.parser.v2.XMLParser.parse(org.xml.sax.InputSource), 1-123
parse(InputStream) - oracle.xml.async.DOMBuilder.parse(java.io.InputStream), 4-72, 4-147
parse(InputStream) -

oracle.xml.parser.v2.XMLParser.parse(java.io.InputStream), 1-123
parse(InputStream, URL, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.parse(java.io.InputStream, java.net.URL, java.io.PrintWriter), 3-39
parse(Reader) - oracle.xml.async.DOMBuilder.parse(java.io.Reader), 4-9, 4-72, 4-147
parse(Reader) - oracle.xml.parser.v2.XMLParser.parse(java.io.Reader), 1-123
parse(Reader, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.parse(java.io.Reader, java.io.PrintWriter), 3-39
parse(String) - oracle.xml.async.DOMBuilder.parse(java.lang.String), 4-9, 4-73, 4-148
parse(String) - oracle.xml.parser.v2.XMLParser.parse(java.lang.String), 1-124
parse(URL) - oracle.xml.async.DOMBuilder.parse(java.net.URL), 4-10, 4-73, 4-148
parse(URL) - oracle.xml.parser.v2.XMLParser.parse(java.net.URL), 1-124
parse(URL, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.parse(java.net.URL, java.io.PrintWriter), 3-39
parseDocument() - oracle.xml.parser.v2.XMLTokenizer.parseDocument(), 1-142
parseDTD(InputSource, String) - oracle.xml.async.DOMBuilder.parseDTD(org.xml.sax.InputSource, java.lang.String), 4-10, 4-74, 4-149
parseDTD(InputSource, String) - oracle.xml.parser.v2.DOMParser.parseDTD(org.xml.sax.InputSource, java.lang.String), 1-17
parseDTD(InputStream, String) - oracle.xml.async.DOMBuilder.parseDTD(java.io.InputStream, java.lang.String), 4-11, 4-74, 4-149
parseDTD(InputStream, String) - oracle.xml.parser.v2.DOMParser.parseDTD(java.io.InputStream, java.lang.String), 1-18
parseDTD(Reader, String) - oracle.xml.async.DOMBuilder.parseDTD(java.io.Reader, java.lang.String), 4-12, 4-74, 4-150
parseDTD(Reader, String) - oracle.xml.parser.v2.DOMParser.parseDTD(java.io.Reader, java.lang.String), 1-18
parseDTD(String, String) - oracle.xml.async.DOMBuilder.parseDTD(java.lang.String, java.lang.String), 4-12, 4-75, 4-150
parseDTD(String, String) - oracle.xml.parser.v2.DOMParser.parseDTD(java.lang.String, java.lang.String), 1-19
parseDTD(URL, String) - oracle.xml.async.DOMBuilder.parseDTD(java.net.URL, java.lang.String), 4-13, 4-75, 4-150
parseDTD(URL, String) - oracle.xml.parser.v2.DOMParser.parseDTD(java.net.URL, java.lang.String), 1-19
parseFromString(String, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.parseFromString(java.lang.String, java.io.PrintWriter), 3-39
parseFromString(StringBuffer, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.parseFromString(java.lang.StringBuffer, java.io.PrintWriter), 3-39
PI - oracle.xml.parser.v2.XMLToken.PI, 1-138
PITarget - oracle.xml.parser.v2.XMLToken.PITarget, 1-138
PLUS - oracle.xml.parser.v2.ElementDecl.PLUS, 1-33
POSTEDXML_ERR - oracle.xml.xsql.Res.POSTEDXML_ERR, 3-9
POSTEDXML_ERR_MSG - oracle.xml.xsql.Res.POSTEDXML_ERR_MSG, 3-9
print(Document, PrintWriter) - oracle.xml.xsql.XSQLParserHelper.print(oracle.xml.xsql.Document, java.io.PrintWriter), 3-39
print(OutputStream) - oracle.xml.parser.v2.XMLDocument.print(java.io.OutputStream), 1-78
print(OutputStream) -

oracle.xml.parser.v2.XMLNode.print(java.io.OutputStream), 1-112
print(OutputStream, String) - oracle.xml.parser.v2.XMLDocument.print(java.io.OutputStream, java.lang.String), 1-78
print(OutputStream, String) - oracle.xml.parser.v2.XMLNode.print(java.io.OutputStream, java.lang.String), 1-112
print(PrintWriter) - oracle.xml.parser.v2.XMLDocument.print(java.io.PrintWriter), 1-78
print(PrintWriter) - oracle.xml.parser.v2.XMLNode.print(java.io.PrintWriter), 1-112
printedErrorHeader() - oracle.xml.xsql.XSQLPageRequestImpl.printedErrorHeader(), 3-34
printedErrorHeader() - oracle.xml.xsql.XSQLPageRequest.printedErrorHeader(), 3-26
printExternalDTD(OutputStream) - oracle.xml.parser.v2.DTD.printExternalDTD(java.io.OutputStream), 1-28
printExternalDTD(OutputStream) - oracle.xml.parser.v2.XMLDocument.printExternalDTD(java.io.OutputStream), 1-79
printExternalDTD(OutputStream, String) - oracle.xml.parser.v2.DTD.printExternalDTD(java.io.OutputStream, java.lang.String), 1-28
printExternalDTD(OutputStream, String) - oracle.xml.parser.v2.XMLDocument.printExternalDTD(java.io.OutputStream, java.lang.String), 1-79
printExternalDTD(PrintWriter) - oracle.xml.parser.v2.DTD.printExternalDTD(java.io.PrintWriter), 1-29
printExternalDTD(PrintWriter) - oracle.xml.parser.v2.XMLDocument.printExternalDTD(java.io.PrintWriter), 1-79
process() - oracle.xml.xsql.XSQLRequest.process(), 3-42
process(Dictionary) - oracle.xml.xsql.XSQLRequest.process(java.util.Dictionary), 3-42
process(Dictionary, PrintWriter, PrintWriter) - oracle.xml.xsql.XSQLRequest.process(java.util.Dictionary, java.io.PrintWriter, java.io.PrintWriter), 3-42
process(PrintWriter, PrintWriter) - oracle.xml.xsql.XSQLRequest.process(java.io.PrintWriter, java.io.PrintWriter), 3-42
processToDocument(Document, String, XSQLPageRequest) - oracle.xml.xsql.XSQLStylesheetProcessor.processToDocument(oracle.xml.xsql.Document, java.lang.String, oracle.xml.xsql.XSQLPageRequest), 3-52
processToWriter(Document, String, XSQLPageRequest) - oracle.xml.xsql.XSQLStylesheetProcessor.processToWriter(oracle.xml.xsql.Document, java.lang.String, oracle.xml.xsql.XSQLPageRequest), 3-52
processToXML() - oracle.xml.xsql.XSQLRequest.processToXML(), 3-43
processToXML(Dictionary) - oracle.xml.xsql.XSQLRequest.processToXML(java.util.Dictionary), 3-43
processToXML(Dictionary, PrintWriter) - oracle.xml.xsql.XSQLRequest.processToXML(java.util.Dictionary, java.io.PrintWriter), 3-43
processToXML(PrintWriter) - oracle.xml.xsql.XSQLRequest.processToXML(java.io.PrintWriter), 3-43
processXSL(XSLStylesheet, InputStream, URL) - oracle.xml.async.XSLTransformer.processXSL(oracle.xml.async.XSLStylesheet, java.io.InputStream, java.net.URL), 4-36, 4-93, 4-168
processXSL(XSLStylesheet, InputStream, URL) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, java.io.InputStream, java.net.URL), 1-148
processXSL(XSLStylesheet, Reader, URL) - oracle.xml.async.XSLTransformer.processXSL(oracle.xml.async.XSLStylesheet, java.io.Reader, java.net.URL), 4-36, 4-94, 4-169
processXSL(XSLStylesheet, Reader, URL) - oracle.xml.parser.v2.XSLProcessor.processXSL(

oracle.xml.parser.v2.XSLStylesheet, java.io.Reader, java.net.URL), 1-149
processXSL(XSLStylesheet, URL, URL) - oracle.xml.async.XSLTransformer.processXSL(oracle.xml.async.XSLStylesheet, java.net.URL, java.net.URL), 4-36, 4-94, 4-169
processXSL(XSLStylesheet, URL, URL) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, java.net.URL, java.net.URL), 1-149
processXSL(XSLStylesheet, XMLDocument) - oracle.xml.async.XSLTransformer.processXSL(oracle.xml.async.XSLStylesheet, oracle.xml.async.XMLDocument), 4-37, 4-94, 4-169
processXSL(XSLStylesheet, XMLDocument) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocument), 1-150
processXSL(XSLStylesheet, XMLDocument, OutputStream) - oracle.xml.async.XSLTransformer.processXSL(oracle.xml.async.XSLStylesheet, oracle.xml.async.XMLDocument, java.io.OutputStream), 4-37, 4-94, 4-169
processXSL(XSLStylesheet, XMLDocument, OutputStream) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocument, java.io.OutputStream), 1-151
processXSL(XSLStylesheet, XMLDocument, PrintWriter) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocument, java.io.PrintWriter), 1-151
processXSL(XSLStylesheet, XMLDocumentFragment) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocumentFragment), 1-150
processXSL(XSLStylesheet, XMLDocumentFragment, OutputStream) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocumentFragment, java.io.OutputStream), 1-150
processXSL(XSLStylesheet, XMLDocumentFragment, PrintWriter) - oracle.xml.parser.v2.XSLProcessor.processXSL(oracle.xml.parser.v2.XSLStylesheet, oracle.xml.parser.v2.XMLDocumentFragment, java.io.PrintWriter), 1-151

Q

QMARK - oracle.xml.parser.v2.ElementDecl.QMARK, 1-33

R

Reference - oracle.xml.parser.v2.XMLToken.Reference, 1-138
releaseResource() - oracle.xml.async.ResourceManager.releaseResource(), 4-32, 4-90, 4-165
removeAttribute(String) - oracle.xml.parser.v2.XMLElement.removeAttribute(java.lang.String), 1-98
removeAttributeNode(Attr) - oracle.xml.parser.v2.XMLElement.removeAttributeNode(org.w3c.dom.Attr), 1-99
removeChild(Node) - oracle.xml.parser.v2.XMLNode.removeChild(org.w3c.dom.Node), 1-113
removeDOMBuilderErrorListener(DOMBuilderErrorListener) - oracle.xml.async.DOMBuilder.removeDOMBuilderErrorListener(oracle.xml.async.DOMBuilderErrorListener), 4-13, 4-76, 4-151
removeDOMBuilderListener(DOMBuilderListener) - oracle.xml.async.DOMBuilder.removeDOMBuilderListener(oracle.xml.async.DOMBuilderListener), 4-14, 4-76, 4-151
removeDOMTransformerErrorListener(XSLTransfor

merErrorListener) - oracle.xml.async.XSLTransformer.removeDOMTransformerErrorListener(oracle.xml.async.XSLTransformerErrorListener), 4-37, 4-94, 4-169
removeXSLTransformerListener(XSLTransformerListener) - oracle.xml.async.XSLTransformer.removeXSLTransformerListener(oracle.xml.async.XSLTransformerListener), 4-37, 4-94, 4-169
replaceChild(Node, Node) - oracle.xml.parser.v2.XMLDocument.replaceChild(org.w3c.dom.Node, org.w3c.dom.Node), 1-80
replaceChild(Node, Node) - oracle.xml.parser.v2.XMLNode.replaceChild(org.w3c.dom.Node, org.w3c.dom.Node), 1-113
replaceXMLData(Connection, String, String, String) - oracle.xml.transviewer.DBAccess.replaceXMLData(java.sql.Connection, java.lang.String, java.lang.String, java.lang.String), 4-109, 4-184
requestProcessed() - oracle.xml.xsql.XSQLPageRequestImpl.requestProcessed(), 3-34
requestProcessed() - oracle.xml.xsql.XSQLPageRequest.requestProcessed(), 3-27
REQUIRED - oracle.xml.parser.v2.AttrDecl.REQUIRED, 1-8
Res - oracle.xml.xsql.Res, 3-2
Res() - oracle.xml.xsql.Res.Res(), 3-11
resolveNamespacePrefix(String) - oracle.xml.parser.v2.NSResolver.resolveNamespacePrefix(java.lang.String), 1-41
resolveNamespacePrefix(String) - oracle.xml.parser.v2.XMLElement.resolveNamespacePrefix(java.lang.String), 1-99
ResourceManager - oracle.xml.async.ResourceManager, 4-31, 4-89, 4-164
ResourceManager(int) - oracle.xml.async.ResourceManager.ResourceManager(int), 4-32, 4-89, 4-165
ROW_TAG - oracle.xml.sql.query.OracleXMLQuery.ROW_TAG, 9-11
ROWIDATTR_TAG - oracle.xml.sql.query.OracleXMLQuery.ROWIDATTR_TAG, 9-11
ROWSET_TAG - oracle.xml.sql.query.OracleXMLQuery.ROWSET_TAG, 9-11
run() - oracle.xml.async.DOMBuilder.run(), 4-14, 4-76, 4-151
run() - oracle.xml.async.XSLTransformer.run(), 4-37, 4-94, 4-169

S

safeURLAsString(URL) - oracle.xml.xsql.XSQLUtil.safeURLAsString(java.net.URL), 3-54
SAXAttrList - oracle.xml.parser.v2.SAXAttrList, 1-44
SAXParser() - oracle.xml.parser.v2.SAXParser.SAXParser(), 1-51
SCHEMA - oracle.xml.sql.query.OracleXMLQuery.SCHEMA, 9-12
select(Document, String) - oracle.xml.xsql.XSQLUtil.select(oracle.xml.xsql.Document, java.lang.String), 3-54
select(Element, String) - oracle.xml.xsql.XSQLUtil.select(oracle.xml.xsql.Element, java.lang.String), 3-54
select(XMLDocument, String) - oracle.xml.xsql.XSQLUtil.select(oracle.xml.xsql.XMLDocument, java.lang.String), 3-55
select(XMLElement, String) - oracle.xml.xsql.XSQLUtil.select(oracle.xml.xsql.XMLElement, java.lang.String), 3-55
selectFirst(Document, String) - oracle.xml.xsql.XSQLUtil.selectFirst(oracle.xml.xsql.Document, java.lang.String), 3-55
selectFirst(Element, String) - oracle.xml.xsql.XSQLUtil.selectFirst(oracle.xml.xsql.Element, java.lang.String), 3-55
selectFirst(XMLDocument, String) -

oracle.xml.xsql.XSQLUtil.selectFirst(oracle.xml.xsql.XMLDocument, java.lang.String), 3-55
selectFirst(XMLElement, String) - oracle.xml.xsql.XSQLUtil.selectFirst(oracle.xml.xsql.XMLElement, java.lang.String), 3-55
selectNodeAt(int) - oracle.xml.srcviewer.XMLSourceView.selectNodeAt(int), 4-59, 4-134
selectNodes(String, NSResolver) - oracle.xml.parser.v2.XMLNode.selectNodes(java.lang.String, oracle.xml.parser.v2.NSResolver), 1-114
selectSingleNode(String, NSResolver) - oracle.xml.parser.v2.XMLNode.selectSingleNode(java.lang.String, oracle.xml.parser.v2.NSResolver), 1-114
setAttribute(String, String) - oracle.xml.parser.v2.XMLElement.setAttribute(java.lang.String, java.lang.String), 1-100
setAttributeNameFont(Font) - oracle.xml.srcviewer.XMLSourceView.setAttributeNameFont(java.awt.Font), 4-59, 4-134
setAttributeNameForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setAttributeNameForeground(java.awt.Color), 4-59, 4-134
setAttributeNode(Attr) - oracle.xml.parser.v2.XMLElement.setAttributeNode(org.w3c.dom.Attr), 1-100
setAttributeValueFont(Font) - oracle.xml.srcviewer.XMLSourceView.setAttributeValueFont(java.awt.Font), 4-60, 4-134
setAttributeValueForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setAttributeValueForeground(java.awt.Color), 4-60, 4-134
setBackground(Color) - oracle.xml.srcviewer.XMLSourceView.setBackground(java.awt.Color), 4-60, 4-135
setBaseURL(URL) - oracle.xml.async.DOMBuilder.setBaseURL(java.net.URL), 4-14, 4-76, 4-151
setBaseURL(URL) - oracle.xml.parser.v2.XMLParser.setBaseURL(java.net.URL), 1-125
setBatchSize(int) - oracle.xml.sql.dml.OracleXMLSave.setBatchSize(int), 9-5
setCDATAFont(Font) - oracle.xml.srcviewer.XMLSourceView.setCDATAFont(java.awt.Font), 4-60, 4-135
setCDATAForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setCDATAForeground(java.awt.Color), 4-60, 4-135
setCollIdAttr(String) - oracle.xml.sql.query.OracleXMLQuery.setCollIdAttr(java.lang.String), 9-14
setCollIdAttrName(String) - oracle.xml.sql.query.OracleXMLQuery.setCollIdAttrName(java.lang.String), 9-14
setCommentDataFont(Font) - oracle.xml.srcviewer.XMLSourceView.setCommentDataFont(java.awt.Font), 4-61, 4-135
setCommentDataForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setCommentDataForeground(java.awt.Color), 4-61, 4-136
setCommitBatch(int) - oracle.xml.sql.dml.OracleXMLSave.setCommitBatch(int), 9-5
setConnectionName(String) - oracle.xml.xsql.XSQLPageRequestImpl.setConnectionName(java.lang.String), 3-34
setConnectionName(String) - oracle.xml.xsql.XSQLPageRequest.setConnectionName(java.lang.String), 3-27
setContentType(String) - oracle.xml.xsql.XSQLPageRequestImpl.setContentType(java.lang.String), 3-35
setContentType(String) - oracle.xml.xsql.XSQLPageRequest.setContentType(java.lang.String), 3-27
setContentType(String) - oracle.xml.xsql.XSQLServletPageRequest.setContentType(java.lang.String), 3-50
setDataHeader(Reader, String) - oracle.xml.sql.query.OracleXMLQuery.setDataHeader(java.io.Reader, java.lang.String), 9-14
setDateFormat(String) - oracle.xml.sql.dml.OracleXMLSave.setDateFor

mat(java.lang.String), 9-6
setDateFormat(String) -
oracle.xml.sql.query.OracleXMLQuery.setDateF
ormat(java.lang.String), 9-14
setDebugMode(boolean) -
oracle.xml.async.DOMBuilder.setDebugMode(b
oolean), 4-14, 4-77, 4-152
setDoctype(DTD) -
oracle.xml.async.DOMBuilder.setDoctype(oracl
e.xml.async.DTD), 4-15, 4-77, 4-152
setDoctype(DTD) -
oracle.xml.parser.v2.DefaultXMLDocumentHan
dler.setDoctype(oracle.xml.parser.v2.DTD), 1-
12
setDoctype(DTD) -
oracle.xml.parser.v2.XMLDocumentHandler.set
Doctype(oracle.xml.parser.v2.DTD), 1-87
setDoctype(DTD) -
oracle.xml.parser.v2.XMLParser.setDoctype(ora
cle.xml.parser.v2.DTD), 1-125
setDocument(CGDocument) -
oracle.xml.classgen.CGNode.setDocument(orac
le.xml.classgen.CGDocument), 2-3
setDocumentHandler(DocumentHandler) -
oracle.xml.parser.v2.SAXParser.setDocumentH
andler(org.xml.sax.DocumentHandler), 1-51
setDTDHandler(DTDHandler) -
oracle.xml.parser.v2.SAXParser.setDTDHandler
(org.xml.sax.DTDHandler), 1-52
setEditable(boolean) -
oracle.xml.srcviewer.XMLSourceView.setEditab
le(boolean), 4-61, 4-136
setEncoding(String) -
oracle.xml.parser.v2.XMLDocument.setEncodin
g(java.lang.String), 1-80
setEncoding(String) -
oracle.xml.sql.query.OracleXMLQuery.setEnco
ding(java.lang.String), 9-14
setEntityResolver(EntityResolver) -
oracle.xml.parser.v2.SAXParser.setEntityResolv
er(org.xml.sax.EntityResolver), 1-52
setErrorHandler(ErrorHandler) -
oracle.xml.parser.v2.SAXParser.setErrorHandle
r(org.xml.sax.ErrorHandler), 1-53
setErrorHandler(ErrorHandler) -
oracle.xml.parser.v2.XMLTokenizer.setErrorHa
ndler(org.xml.sax.ErrorHandler), 1-142
setErrorStream(OutputStream) -
oracle.xml.async.DOMBuilder.setErrorStream(j
ava.io.OutputStream), 4-15, 4-77, 4-152
setErrorStream(OutputStream) -
oracle.xml.async.XSLTransformer.setErrorStrea
m(java.io.OutputStream), 4-38, 4-95, 4-170
setErrorStream(OutputStream) -
oracle.xml.parser.v2.DOMParser.setErrorStrea
m(java.io.OutputStream), 1-19
setErrorStream(OutputStream) -
oracle.xml.parser.v2.XMLTokenizer.setErrorStr
eam(java.io.OutputStream), 1-143
setErrorStream(OutputStream) -
oracle.xml.parser.v2.XSLProcessor.setErrorStrea
m(java.io.OutputStream), 1-152
setErrorStream(OutputStream, String) -
oracle.xml.async.DOMBuilder.setErrorStream(j
ava.io.OutputStream, java.lang.String), 4-15,
4-77, 4-152
setErrorStream(OutputStream, String) -
oracle.xml.parser.v2.DOMParser.setErrorStrea
m(java.io.OutputStream,
java.lang.String), 1-20
setErrorStream(PrintWriter) -
oracle.xml.async.DOMBuilder.setErrorStream(j
ava.io.PrintWriter), 4-16, 4-78, 4-153
setErrorStream(PrintWriter) -
oracle.xml.parser.v2.DOMParser.setErrorStrea
m(java.io.PrintWriter), 1-20
setErrorTag(String) -
oracle.xml.sql.OracleXMLSQLException.setErro
rTag(java.lang.String), 9-20
setErrorTag(String) -
oracle.xml.sql.query.OracleXMLQuery.setError
Tag(java.lang.String), 9-15
setException(Exception) -
oracle.xml.sql.query.OracleXMLQuery.setExcep
tion(java.lang.Exception), 9-15
setGenerateComments(boolean) -
oracle.xml.classgen.ClassGenerator.setGenerate
Comments(boolean), 2-5
setIgnoreCase(boolean) -
oracle.xml.sql.dml.OracleXMLSave.setIgnoreCa

se(boolean), 9-6
setIncludingRequest(XSQLPageRequest) - oracle.xml.xsql.XSQLPageRequestImpl.setIncludingRequest(oracle.xml.xsql.XSQLPageRequest), 3-35
setIncludingRequest(XSQLPageRequest) - oracle.xml.xsql.XSQLPageRequest.setIncludingRequest(oracle.xml.xsql.XSQLPageRequest), 3-27
setJavaPackage(String) - oracle.xml.classgen.ClassGenerator.setJavaPackage(java.lang.String), 2-5
setKeyColumnList(String[]) - oracle.xml.sql.dml.OracleXMLSave.setKeyColumnList(java.lang.String[]), 9-6
setLocale(Locale) - oracle.xml.parser.v2.XMLDocument.setLocale(java.util.Locale), 1-80
setLocale(Locale) - oracle.xml.parser.v2.XMLParser.setLocale(java.util.Locale), 1-125
setLocale(Locale) - oracle.xml.parser.v2.XSLProcessor.setLocale(java.util.Locale), 1-152
setMaxRows(int) - oracle.xml.sql.query.OracleXMLQuery.setMaxRows(int), 9-15
setMetaHeader(Reader) - oracle.xml.sql.query.OracleXMLQuery.setMetaHeader(java.io.Reader), 9-15
setNodeFactory(NodeFactory) - oracle.xml.async.DOMBuilder.setNodeFactory(oracle.xml.async.NodeFactory), 4-16, 4-78, 4-153
setNodeFactory(NodeFactory) - oracle.xml.parser.v2.DOMParser.setNodeFactory(oracle.xml.parser.v2.NodeFactory), 1-20
setNodeValue(String) - oracle.xml.parser.v2.XMLAttr.setNodeValue(java.lang.String), 1-60
setNodeValue(String) - oracle.xml.parser.v2.XMLNode.setNodeValue(java.lang.String), 1-115
setOutputDirectory(String) - oracle.xml.classgen.ClassGenerator.setOutputDirectory(java.lang.String), 2-5
setPageEncoding(String) - oracle.xml.xsql.XSQLPageRequestImpl.setPageEncoding(java.lang.String), 3-35
setPageEncoding(String) - oracle.xml.xsql.XSQLPageRequest.setPageEncoding(java.lang.String), 3-27
setPageEncoding(String) - oracle.xml.xsql.XSQLServletPageRequest.setPageEncoding(java.lang.String), 3-50
setPageParam(String, String) - oracle.xml.xsql.XSQLPageRequestImpl.setPageParam(java.lang.String, java.lang.String), 3-35
setPageParam(String, String) - oracle.xml.xsql.XSQLPageRequest.setPageParam(java.lang.String, java.lang.String), 3-27
setParam(String, String) - oracle.xml.parser.v2.XSLStylesheet.setParam(java.lang.String, java.lang.String), 1-155
setPCDATAFont(Font) - oracle.xml.srcviewer.XMLSourceView.setPCDATAFont(java.awt.Font), 4-61, 4-136
setPCDATAForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setPCDATAForeground(java.awt.Color), 4-61, 4-136
setPIDataFont(Font) - oracle.xml.srcviewer.XMLSourceView.setPIDataFont(java.awt.Font), 4-62, 4-136
setPIDataForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setPIDataForeground(java.awt.Color), 4-62, 4-137
setPINameFont(Font) - oracle.xml.srcviewer.XMLSourceView.setPINameFont(java.awt.Font), 4-62, 4-137
setPINameForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setPINameForeground(java.awt.Color), 4-62, 4-137
setPostedDocument(Document) - oracle.xml.xsql.XSQLPageRequestImpl.setPostedDocument(oracle.xml.xsql.Document), 3-35
setPostedDocument(Document) - oracle.xml.xsql.XSQLPageRequest.setPostedDocument(oracle.xml.xsql.Document), 3-27
setPostedDocument(Document) - oracle.xml.xsql.XSQLRequest.setPostedDocume

nt(org.w3c.dom.Document), 3-43
setPreserveWhitespace(boolean) - oracle.xml.async.DOMBuilder.setPreserveWhitespace(boolean), 4-16, 4-78, 4-153
setPreserveWhitespace(boolean) - oracle.xml.parser.v2.XMLParser.setPreserveWhitespace(boolean), 1-125
setPrintedErrorHeader(boolean) - oracle.xml.xsql.XSQLPageRequestImpl.setPrintedErrorHeader(boolean), 3-36
setPrintedErrorHeader(boolean) - oracle.xml.xsql.XSQLPageRequest.setPrintedErrorHeader(boolean), 3-27
setRaiseException(boolean) - oracle.xml.sql.query.OracleXMLQuery.setRaiseException(boolean), 9-15
setRaiseNoRowsException(boolean) - oracle.xml.sql.query.OracleXMLQuery.setRaiseNoRowsException(boolean), 9-15
setRowIdAttrName(String) - oracle.xml.sql.query.OracleXMLQuery.setRowIdAttrName(java.lang.String), 9-16
setRowIdAttrValue(String) - oracle.xml.sql.query.OracleXMLQuery.setRowIdAttrValue(java.lang.String), 9-16
setRowsetTag(String) - oracle.xml.sql.query.OracleXMLQuery.setRowsetTag(java.lang.String), 9-16
setRowTag(String) - oracle.xml.sql.dml.OracleXMLSave.setRowTag(java.lang.String), 9-6
setRowTag(String) - oracle.xml.sql.query.OracleXMLQuery.setRowTag(java.lang.String), 9-16
setSelectedNode(Node) - oracle.xml.srcviewer.XMLSourceView.setSelectedNode(org.w3c.dom.Node), 4-62, 4-137
setSerializationMode(boolean) - oracle.xml.classgen.ClassGenerator.setSerializationMode(boolean), 2-6
setSkipRows(int) - oracle.xml.sql.query.OracleXMLQuery.setSkipRows(int), 9-16
setStandalone(String) - oracle.xml.parser.v2.XMLDocument.setStandalone(java.lang.String), 1-81
setStyleSheet(String) - oracle.xml.sql.query.OracleXMLQuery.setStyleSheet(java.lang.String), 9-16
setStylesheetHeader(String) - oracle.xml.sql.query.OracleXMLQuery.setStylesheetHeader(java.lang.String), 9-16
setStylesheetHeader(String, String) - oracle.xml.sql.query.OracleXMLQuery.setStylesheetHeader(java.lang.String, java.lang.String), 9-17
setStylesheetParameter(String, String) - oracle.xml.xsql.XSQLPageRequestImpl.setStylesheetParameter(java.lang.String, java.lang.String), 3-36
setStylesheetParameter(String, String) - oracle.xml.xsql.XSQLPageRequest.setStylesheetParameter(java.lang.String, java.lang.String), 3-27
setStylesheetURI(String) - oracle.xml.xsql.XSQLPageRequestImpl.setStylesheetURI(java.lang.String), 3-36
setStylesheetURI(String) - oracle.xml.xsql.XSQLPageRequest.setStylesheetURI(java.lang.String), 3-28
setSymbolFont(Font) - oracle.xml.srcviewer.XMLSourceView.setSymbolFont(java.awt.Font), 4-63, 4-137
setSymbolForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setSymbolForeground(java.awt.Color), 4-63, 4-138
setTagFont(Font) - oracle.xml.srcviewer.XMLSourceView.setTagFont(java.awt.Font), 4-63, 4-138
setTagForeground(Color) - oracle.xml.srcviewer.XMLSourceView.setTagForeground(java.awt.Color), 4-63, 4-138
setTextDecl(String, String) - oracle.xml.parser.v2.DefaultXMLDocumentHandler.setTextDecl(java.lang.String, java.lang.String), 1-13
setTextDecl(String, String) - oracle.xml.parser.v2.XMLDocumentHandler.setTextDecl(java.lang.String, java.lang.String), 1-88

setToken(int, boolean) -
oracle.xml.parser.v2.XMLTokenizer.setToken(int, boolean), 1-143
setTokenHandler(XMLToken) -
oracle.xml.parser.v2.XMLTokenizer.setTokenHandler(oracle.xml.parser.v2.XMLToken), 1-143
setUpdateColumnList(String[]) -
oracle.xml.sql.dml.OracleXMLSave.setUpdateColumnList(java.lang.String[]), 9-6
setValidationMode(boolean) -
oracle.xml.async.DOMBuilder.setValidationMode(boolean), 4-17, 4-78, 4-154
setValidationMode(boolean) -
oracle.xml.classgen.ClassGenerator.setValidationMode(boolean), 2-6
setValidationMode(boolean) -
oracle.xml.parser.v2.XMLParser.setValidationMode(boolean), 1-126
setValue(String) -
oracle.xml.parser.v2.XMLAttr.setValue(java.lang.String), 1-61
setVersion(String) -
oracle.xml.parser.v2.XMLDocument.setVersion(java.lang.String), 1-81
setXMLDecl(String, String, String) -
oracle.xml.parser.v2.DefaultXMLDocumentHandler.setXMLDecl(java.lang.String, java.lang.String, java.lang.String), 1-13
setXMLDecl(String, String, String) -
oracle.xml.parser.v2.XMLDocumentHandler.setXMLDecl(java.lang.String, java.lang.String, java.lang.String), 1-88
setXMLDocument(Document) -
oracle.xml.srcviewer.XMLSourceView.setXMLDocument(org.w3c.dom.Document), 4-63, 4-138
setXMLDocument(Document) -
oracle.xml.treeviewer.XMLTreeView.setXMLDocument(org.w3c.dom.Document), 4-121, 4-196, 4-203
setXSLT(Reader, String), 9-17
setXSLT(String, String), 9-17
showWarnings(boolean) -
oracle.xml.async.DOMBuilder.showWarnings(boolean), 4-17, 4-79, 4-154
showWarnings(boolean) -
oracle.xml.async.XSLTransformer.showWarnings(boolean), 4-38, 4-95, 4-170
showWarnings(boolean) -
oracle.xml.parser.v2.DOMParser.showWarnings(boolean), 1-21
showWarnings(boolean) -
oracle.xml.parser.v2.XSLProcessor.showWarnings(boolean), 1-152
SKIPROWS_ALL -
oracle.xml.sql.query.OracleXMLQuery.SKIPROWS_ALL, 9-12
SKIPROWS_DEFAULT -
oracle.xml.sql.query.OracleXMLQuery.SKIPROWS_DEFAULT, 9-12
sleep(int) -
oracle.xml.async.ResourceManager.sleep(int), 4-32, 4-90, 4-165
splitText(int) -
oracle.xml.parser.v2.XMLText.splitText(int), 1-132
STag - oracle.xml.parser.v2.XMLToken.STag, 1-138
STagName -
oracle.xml.parser.v2.XMLToken.STagName, 1-138
startElement(NSName, SAXAttrList) -
oracle.xml.parser.v2.DefaultXMLDocumentHandler.startElement(oracle.xml.parser.v2.NSName, oracle.xml.parser.v2.SAXAttrList), 1-14
startElement(NSName, SAXAttrList) -
oracle.xml.parser.v2.XMLDocumentHandler.startElement(oracle.xml.parser.v2.NSName, oracle.xml.parser.v2.SAXAttrList), 1-88
stringParamValue(Object) -
oracle.xml.xsql.XSQLUtil.stringParamValue(java.lang.Object), 3-55

T

TextDecl -
oracle.xml.parser.v2.XMLToken.TextDecl, 1-138
token(int, String) -
oracle.xml.parser.v2.XMLToken.token(int, java.lang.String), 1-139

tokenize(InputSource) - oracle.xml.parser.v2.XMLTokenizer.tokenize(org.xml.sax.InputSource), 1-143
tokenize(InputStream) - oracle.xml.parser.v2.XMLTokenizer.tokenize(java.io.InputStream), 1-143
tokenize(Reader) - oracle.xml.parser.v2.XMLTokenizer.tokenize(java.io.Reader), 1-144
tokenize(String) - oracle.xml.parser.v2.XMLTokenizer.tokenize(java.lang.String), 1-144
tokenize(URL) - oracle.xml.parser.v2.XMLTokenizer.tokenize(java.net.URL), 1-145
transformNode(XSLStylesheet) - oracle.xml.parser.v2.XMLNode.transformNode(oracle.xml.parser.v2.XSLStylesheet), 1-115
translate(String, String) - oracle.xml.xsql.XSQLUtil.translate(java.lang.String, java.lang.String), 3-55
translate(URL, String) - oracle.xml.xsql.XSQLUtil.translate(java.net.URL, java.lang.String), 3-55
translateURL(String) - oracle.xml.xsql.XSQLPageRequestImpl.translateURL(java.lang.String), 3-36
translateURL(String) - oracle.xml.xsql.XSQLPageRequest.translateURL(java.lang.String), 3-28
translateURL(String) - oracle.xml.xsql.XSQLServletPageRequest.translateURL(java.lang.String), 3-50

U

UNHANDLED_ERR - oracle.xml.xsql.Res.UNHANDLED_ERR, 3-9
UNHANDLED_ERR_MSG - oracle.xml.xsql.Res.UNHANDLED_ERR_MSG, 3-9
UNHANDLED_ERR_XSL_PR - oracle.xml.xsql.Res.UNHANDLED_ERR_XSL_PR, 3-10
UNHANDLED_ERR_XSL_PR_MSG - oracle.xml.xsql.Res.UNHANDLED_ERR_XSL_PR_MSG, 3-10
UNHANDLED_ERR_XSL_RD - oracle.xml.xsql.Res.UNHANDLED_ERR_XSL_RD, 3-10
UNHANDLED_ERR_XSL_RD_MSG - oracle.xml.xsql.Res.UNHANDLED_ERR_XSL_RD_MSG, 3-10
updateUI() - oracle.xml.treeviewer.XMLTreeView.updateUI(), 4-121, 4-196, 4-203
updateXML(Document) - oracle.xml.sql.dml.OracleXMLSave.updateXML(org.w3c.dom.Document), 9-7
updateXML(InputStream) - oracle.xml.sql.dml.OracleXMLSave.updateXML(java.io.InputStream), 9-7
updateXML(String) - oracle.xml.sql.dml.OracleXMLSave.updateXML(java.lang.String), 9-7
updateXML(URL) - oracle.xml.sql.dml.OracleXMLSave.updateXML(java.net.URL), 9-7
useConnectionPooling() - oracle.xml.xsql.XSQLPageRequestImpl.useConnectionPooling(), 3-36
useConnectionPooling() - oracle.xml.xsql.XSQLPageRequest.useConnectionPooling(), 3-28
useHTMLErrors() - oracle.xml.xsql.XSQLPageRequestImpl.useHTMLErrors(), 3-36
useHTMLErrors() - oracle.xml.xsql.XSQLPageRequest.useHTMLErrors(), 3-28
useHTMLErrors() - oracle.xml.xsql.XSQLServletPageRequest.useHTMLErrors(), 3-50
useLowerCaseTagNames() - oracle.xml.sql.query.OracleXMLQuery.useLowerCaseTagNames(), 9-17
useNullAttributeIndicator(boolean) - oracle.xml.sql.query.OracleXMLQuery.useNullAttributeIndicator(boolean), 9-17
useUpperCaseTagNames() -

oracle.xml.sql.query.OracleXMLQuery.useUpperCaseTagNames(), 9-17

V

validateContent(Element) - oracle.xml.parser.v2.ElementDecl.validateContent(org.w3c.dom.Element), 1-35
validateElementContent(Element) - oracle.xml.parser.v2.XMLDocument.validateElementContent(org.w3c.dom.Element), 1-81
valueOf(Element, String) - oracle.xml.xsql.XSQLUtil.valueOf(oracle.xml.xsql.Element, java.lang.String), 3-55
valueOf(Node, String) - oracle.xml.xsql.XSQLUtil.valueOf(oracle.xml.xsql.Node, java.lang.String), 3-56
valueOf(String, NSResolver) - oracle.xml.parser.v2.XMLNode.valueOf(java.lang.String, oracle.xml.parser.v2.NSResolver), 1-115
valueOf(XMLElement, String) - oracle.xml.xsql.XSQLUtil.valueOf(oracle.xml.xsql.XMLElement, java.lang.String), 3-56
valueOf(XMLNode, String) - oracle.xml.xsql.XSQLUtil.valueOf(oracle.xml.xsql.XMLNode, java.lang.String), 3-56

W

WARNING - oracle.xml.parser.v2.XMLParseException.WARNING, 1-118

X

XL(String, String) - oracle.xml.xsql.XSQLHttpUtil.XL(java.lang.String, java.lang.String), 3-22
XL(String, String) - oracle.xml.xsql.XSQLUtil.XL(java.lang.String, java.lang.String), 3-56
XML_INS_ERR - oracle.xml.xsql.Res.XML_INS_ERR, 3-10
XML_INS_ERR_MSG - oracle.xml.xsql.Res.XML_INS_ERR_MSG, 3-10
XML_PARSE - oracle.xml.xsql.Res.XML_PARSE, 3-10
XML_PARSE_MSG - oracle.xml.xsql.Res.XML_PARSE_MSG, 3-10
XML_SQL_ERR - oracle.xml.xsql.Res.XML_SQL_ERR, 3-10
XML_SQL_ERR_MSG - oracle.xml.xsql.Res.XML_SQL_ERR_MSG, 3-10
XMLAttr(String, String) - oracle.xml.parser.v2.XMLAttr.XMLAttr(java.lang.String, java.lang.String), 1-56
XMLAttr(String, String, String, String) - oracle.xml.parser.v2.XMLAttr.XMLAttr(java.lang.String, java.lang.String, java.lang.String, java.lang.String), 1-56
XMLCDATA - oracle.xml.parser.v2.XMLCDATA, 1-62
XMLCDATA(String) - oracle.xml.parser.v2.XMLCDATA.XMLCDATA(java.lang.String), 1-64
XMLComment - oracle.xml.parser.v2.XMLComment, 1-65
XMLComment(String) - oracle.xml.parser.v2.XMLComment.XMLComment(java.lang.String), 1-67
XMLDecl - oracle.xml.parser.v2.XMLToken.XMLDecl, 1-138
XMLDocument - oracle.xml.parser.v2.XMLDocument, 1-68
XMLDocument() - oracle.xml.parser.v2.XMLDocument.XMLDocument(), 1-71
XMLDocumentFragment - oracle.xml.parser.v2.XMLDocumentFragment, 1-82
XMLDocumentFragment() - oracle.xml.parser.v2.XMLDocumentFragment.XMLDocumentFragment(), 1-84
XMLDocumentHandler - oracle.xml.parser.v2.XMLDocumentHandler, 1-85

XMLElement - oracle.xml.parser.v2.XMLElement, 1-90
XMLElement(String) - oracle.xml.parser.v2.XMLElement.XMLElement(java.lang.String), 1-93
XMLElement(String, String, String) - oracle.xml.parser.v2.XMLElement.XMLElement(java.lang.String, java.lang.String, java.lang.String), 1-93
XMLEntityReference - oracle.xml.parser.v2.XMLEntityReference, 1-102
XMLEntityReference(String) - oracle.xml.parser.v2.XMLEntityReference.XMLEntityReference(java.lang.String), 1-103
XMLNode - oracle.xml.parser.v2.XMLNode, 1-104
XMLParseException - oracle.xml.parser.v2.XMLParseException, 1-117
XMLParseException(String, String, String, int, int, int) - oracle.xml.parser.v2.XMLParseException.XMLParseException(java.lang.String, java.lang.String, java.lang.String, int, int, int), 1-119
XMLParser - oracle.xml.parser.v2.XMLParser, 1-121
XMLPI - oracle.xml.parser.v2.XMLPI, 1-127
XMLPI(String, String) - oracle.xml.parser.v2.XMLPI.XMLPI(java.lang.String, java.lang.String), 1-129
XMLSourceView - oracle.xml.srcviewer.XMLSourceView, 4-49, 4-124
XMLSourceView() - oracle.xml.srcviewer.XMLSourceView.XMLSourceView(), 4-53, 4-128
XMLSourceViewBeanInfo - oracle.xml.srcviewer.XMLSourceViewBeanInfo, 4-65, 4-139
XMLSourceViewBeanInfo() - oracle.xml.srcviewer.XMLSourceViewBeanInfo.XMLSourceViewBeanInfo(), 4-66, 4-140
xmlTableExists(Connection, String) - oracle.xml.transviewer.DBAccess.xmlTableExists(java.sql.Connection, java.lang.String), 4-110, 4-185
XMLText - oracle.xml.parser.v2.XMLText, 1-130
XMLText(String) - oracle.xml.parser.v2.XMLText.XMLText(java.lang.String), 1-132
XMLToken - oracle.xml.parser.v2.XMLToken, 1-134
XMLTokenizer - oracle.xml.parser.v2.XMLTokenizer, 1-140
XMLTokenizer() - oracle.xml.parser.v2.XMLTokenizer.XMLTokenizer(), 1-142
XMLTokenizer(XMLToken) - oracle.xml.parser.v2.XMLTokenizer.XMLTokenizer(oracle.xml.parser.v2.XMLToken), 1-142
XMLTransformPanel - oracle.xml.transviewer.XMLTransformPanel, 4-111, 4-186
XMLTransformPanel() - oracle.xml.transviewer.XMLTransformPanel.XMLTransformPanel(), 4-113, 4-188
XMLTransformPanelBeanInfo - oracle.xml.transviewer.XMLTransformPanelBeanInfo, 4-114, 4-189
XMLTransformPanelBeanInfo() - oracle.xml.transviewer.XMLTransformPanelBeanInfo.XMLTransformPanelBeanInfo(), 4-115, 4-190
XMLTransViewer - oracle.xml.transviewer.XMLTransViewer, 4-116, 4-191
XMLTransViewer() - oracle.xml.transviewer.XMLTransViewer.XMLTransViewer(), 4-116, 4-191
XMLTreeView - oracle.xml.treeviewer.XMLTreeView, 4-118, 4-193, 4-200
XMLTreeView() - oracle.xml.treeviewer.XMLTreeView.XMLTreeView(), 4-120, 4-195, 4-202
XMLTreeViewBeanInfo - oracle.xml.treeviewer.XMLTreeViewBeanInfo, 4-122, 4-197, 4-204
XMLTreeViewBeanInfo() - oracle.xml.treeviewer.XMLTreeViewBeanInfo.X

MLTreeViewBeanInfo(), 4-123, 4-198, 4-205
XSL_ERRORS -
oracle.xml.xsql.Res.XSL_ERRORS, 3-10
XSL_ERRORS_MSG -
oracle.xml.xsql.Res.XSL_ERRORS_MSG, 3-10
XSL_NOFILE -
oracle.xml.xsql.Res.XSL_NOFILE, 3-11
XSL_NOFILE_MSG -
oracle.xml.xsql.Res.XSL_NOFILE_MSG, 3-11
XSL_PARSE -
oracle.xml.xsql.Res.XSL_PARSE, 3-11
XSL_PARSE_MSG -
oracle.xml.xsql.Res.XSL_PARSE_MSG, 3-11
XSLException -
oracle.xml.parser.v2.XSLException, 1-146
XSLNOTFOUND -
oracle.xml.xsql.Res.XSLNOTFOUND, 3-11
XSLNOTFOUND_MSG -
oracle.xml.xsql.Res.XSLNOTFOUND_MSG, 3-11
XSLProcessor -
oracle.xml.parser.v2.XSLProcessor, 1-147
XSLProcessor() -
oracle.xml.parser.v2.XSLProcessor.XSLProcessor(), 1-148
XSLStylesheet -
oracle.xml.parser.v2.XSLStylesheet, 1-153
XSLStylesheet(InputStream, URL) -
oracle.xml.parser.v2.XSLStylesheet.XSLStylesheet(java.io.InputStream, java.net.URL), 1-154
XSLStylesheet(Reader, URL) -
oracle.xml.parser.v2.XSLStylesheet.XSLStylesheet(java.io.Reader, java.net.URL), 1-154
XSLStylesheet(URL, URL) -
oracle.xml.parser.v2.XSLStylesheet.XSLStylesheet(java.net.URL, java.net.URL), 1-154
XSLStylesheet(XMLDocument, URL) -
oracle.xml.parser.v2.XSLStylesheet.XSLStylesheet(oracle.xml.parser.v2.XMLDocument, java.net.URL), 1-155
XSLTransformer -
oracle.xml.async.XSLTransformer, 4-33, 4-91, 4-166
XSLTransformer() -
oracle.xml.async.XSLTransformer.XSLTransformer(), 4-35, 4-92, 4-167
XSLTransformer(int) -
oracle.xml.async.XSLTransformer.XSLTransformer(int), 4-35, 4-92, 4-167
XSLTransformerBeanInfo -
oracle.xml.async.XSLTransformerBeanInfo, 4-40, 4-96, 4-171
XSLTransformerBeanInfo() -
oracle.xml.async.XSLTransformerBeanInfo.XSLTransformerBeanInfo(), 4-41, 4-97, 4-172
xslTransformerError(XSLTransformerEvent) -
oracle.xml.async.XSLTransformerListener.xslTransformerError(oracle.xml.async.XSLTransformerEvent), 4-47, 4-103, 4-178
xslTransformerErrorCalled(XSLTransformerErrorEvent) -
oracle.xml.async.XSLTransformerErrorListener.xslTransformerErrorCalled(oracle.xml.async.XSLTransformerErrorEvent), 4-44, 4-100, 4-175
XSLTransformerErrorEvent -
oracle.xml.async.XSLTransformerErrorEvent, 4-42, 4-98, 4-173
XSLTransformerErrorEvent(Object, Exception) -
oracle.xml.async.XSLTransformerErrorEvent.XSLTransformerErrorEvent(java.lang.Object, java.lang.Exception), 4-43, 4-99, 4-174
XSLTransformerErrorListener -
oracle.xml.async.XSLTransformerErrorListener, 4-44, 4-100, 4-175
XSLTransformerEvent -
oracle.xml.async.XSLTransformerEvent, 4-45, 4-101, 4-176
XSLTransformerEvent(Object, int) -
oracle.xml.async.XSLTransformerEvent.XSLTransformerEvent(java.lang.Object, int), 4-46, 4-102, 4-177
XSLTransformerListener -
oracle.xml.async.XSLTransformerListener, 4-47, 4-103, 4-178
xslTransformerOver(XSLTransformerEvent) -
oracle.xml.async.XSLTransformerListener.xslTransformerOver(oracle.xml.async.XSLTransformerEvent), 4-48, 4-104, 4-179
xslTransformerStarted(XSLTransformerEvent) -
oracle.xml.async.XSLTransformerListener.xslTr

ansformerStarted(oracle.xml.async.XSLTransformerEvent), 4-48, 4-104, 4-179
XSQLActionHandler - oracle.xml.xsql.XSQLActionHandler, 3-12
XSQLActionHandlerImpl - oracle.xml.xsql.XSQLActionHandlerImpl, 3-15
XSQLActionHandlerImpl() - oracle.xml.xsql.XSQLActionHandlerImpl.XSQLActionHandlerImpl(), 3-16
XSQLCommandLine - oracle.xml.xsql.XSQLCommandLine, 3-17
XSQLCommandLine() - oracle.xml.xsql.XSQLCommandLine.XSQLCommandLine(), 3-17
XSQLDiagnostic - oracle.xml.xsql.XSQLDiagnostic, 3-19
XSQLDiagnostic(String) - oracle.xml.xsql.XSQLDiagnostic.XSQLDiagnostic(java.lang.String), 3-19
XSQLHttpUtil - oracle.xml.xsql.XSQLHttpUtil, 3-21
XSQLHttpUtil() - oracle.xml.xsql.XSQLHttpUtil.XSQLHttpUtil(), 3-21
XSQLPageRequest - oracle.xml.xsql.XSQLPageRequest, 3-23
XSQLPageRequestImpl - oracle.xml.xsql.XSQLPageRequestImpl, 3-29
XSQLPageRequestImpl() - oracle.xml.xsql.XSQLPageRequestImpl.XSQLPageRequestImpl(), 3-31
XSQLPageRequestImpl(Hashtable) - oracle.xml.xsql.XSQLPageRequestImpl.XSQLPageRequestImpl(java.util.Hashtable), 3-31
XSQLPageRequestImpl(String, Hashtable) - oracle.xml.xsql.XSQLPageRequestImpl.XSQLPageRequestImpl(java.lang.String, java.util.Hashtable), 3-31
XSQLParserHelper - oracle.xml.xsql.XSQLParserHelper, 3-38
XSQLParserHelper() - oracle.xml.xsql.XSQLParserHelper.XSQLParserHelper(), 3-39
XSQLRequest - oracle.xml.xsql.XSQLRequest, 3-40
XSQLRequest(String) - oracle.xml.xsql.XSQLRequest.XSQLRequest(java.lang.String), 3-41
XSQLRequest(String, XSQLPageRequest) - oracle.xml.xsql.XSQLRequest.XSQLRequest(java.lang.String, oracle.xml.xsql.XSQLPageRequest), 3-41
XSQLRequest(URL) - oracle.xml.xsql.XSQLRequest.XSQLRequest(java.net.URL), 3-41
XSQLRequest(URL, XSQLPageRequest) - oracle.xml.xsql.XSQLRequest.XSQLRequest(java.net.URL, oracle.xml.xsql.XSQLPageRequest), 3-42
XSQLServlet - oracle.xml.xsql.XSQLServlet, 3-45
XSQLServlet() - oracle.xml.xsql.XSQLServlet.XSQLServlet(), 3-45
XSQLServletPageRequest - oracle.xml.xsql.XSQLServletPageRequest, 3-47
XSQLServletPageRequest(HttpServletRequest, HttpServletResponse) - oracle.xml.xsql.XSQLServletPageRequest.XSQLServletPageRequest(oracle.xml.xsql.HttpServletRequest, oracle.xml.xsql.HttpServletResponse), 3-48
XSQLStylesheetProcessor - oracle.xml.xsql.XSQLStylesheetProcessor, 3-51
XSQLStylesheetProcessor() - oracle.xml.xsql.XSQLStylesheetProcessor.XSQLStylesheetProcessor(), 3-51
XSQLUtil - oracle.xml.xsql.XSQLUtil, 3-53
XSQLUtil() - oracle.xml.xsql.XSQLUtil.XSQLUtil(), 3-54