Xerces-C++ Documentation

Copyright 2003 The Apache Software Foundation. All Rights Reserved.

Table of Contents

1 Xerces C++ Parser

Xerces-C++ Version 2.5.0

Xerces-C++ is a validating XML parser written in a portable subset of C++. Xerces-C++ makes it easy to give your application the ability to read and write XML [1] data. A shared library is provided for parsing, generating, manipulating, and validating XML documents.

Xerces-C++ is faithful to the XML 1.0 [2] recommendation and many associated standards (see Features below).

The parser provides high performance, modularity, and scalability. Source code, samples and API documentation are provided with the parser. For portability, care has been taken to make minimal use of templates, no RTTI, and minimal use of #ifdefs.

Applications of the Xerces Parser

Xerces has rich generating and validating capabilities. The parser is used for:

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

Features

- Conforms to
  - XML 1.0 (Second Edition) [3] , W3C Recommendation of October 6, 2000
  - DOM Level 1 Specification [4] , W3C Recommendation of October 1, 1998
  - DOM Level 2 Core Specification [5] , W3C Recommendation of November 13, 2000
  - DOM Level 2 Traversal and Range Specification [6] , W3C Recommendation of November 13, 2000
  - SAX 1.0 and SAX 2.0 [7]
  - Namespaces in XML [8] , W3C Recommendation of January 14, 1999
  - XML Schema Part 1: Structure [9] , W3C Recommendation 2 May 2001
  - XML Schema Part 2: Datatypes [10] , W3C Recommendation 2 May 2001
- Full experimental implementation of XML 1.1 [11] , W3C Candidate Recommendation of October 15, 2002 (Note: section 2.13 Normalization Checking has not been implemented)
- Full experimental implementation of Namespaces in XML 1.1 [12] , W3C Candidate Recommendation of 18 December 2002
- Contains a partial implementation of the DOM Level 3.0 Core Specification [13] , Version 1.0 W3C Working Draft 26 February 2003 and Document Object Model (DOM) Level 3 Load and Save

Specification [14] , Version 1.0 W3C Working Draft 26 February 2003. This implementation is experimental. See DOM Level 3 Support for detail.

- Source code, samples, and documentation is provided
- Programmatic generation and validation of XML
- Pluggable catalogs, validators and encodings
- High performance
- Customizable error handling

Platforms with Binaries

Operating System	**Compiler**
32-bit binary	
Windows NT 4.0 SP5	MSVC 6.0 SP3
Redhat Linux 7.2	Intel C++ Compiler v7, icc
Redhat Linux 8.0	gcc Compiler v3.2
AIX 5.1	xlC_r 6.0.0
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads
SuSE Linux 7.2 (S390)	g++ 2.95.3
64-bit binary	
Windows XP, IA64	Intel C++ Compiler v7, ecl
Redhat Linux 7.2, IA64	Intel C++ Compiler v7, ecc
AIX 5.1	xlC_r 6.0.0
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads

Other ports...

- OS/390
- AS/400
- FreeBSD
- SGI IRIX
- Macintosh
- OS/2
- PTX
- UnixWare
- and more!

License Information

The Xerces-C++ Version 2.5.0 is available in both source distribution and binary distribution. Both Xerces-C++ are made available under the Apache Software License [15] .

2 Releases

Releases Plan

For future release plan about Xerces-C++, please refer to Releases Plan.

Releases Archive

For release information about Xerces-C++ 2.4.0 or earlier, please refer to Releases Archive.

Release Information of Xerces-C++ 2.5.0: Feb. 16, 2004

Date	Contributor	Description
2004-02-15	Neil Graham	make first parameter of BinOutputStream::writeBytes const * const; bug 26936
2004-02-13	Khaled Noaman	Remove the limitation on providing PSVI information
2004-02-13	David Cargill	Update threadtest to accept -init option instead of using compiler directive.
2004-02-13	David Cargill	Remove unnecessary if statement
2004-02-13	David Cargill	Bug#26900 fix, remove virtual on destructor
2004-02-12	Alberto Massari	Xercesc2_5_0 Updates
2004-02-12	PeiYong Zhang	Xercesc2_5_0 Updates
2004-02-12	David Cargill	PSVIWriter documentation updates
2004-02-12	Merlin	Bug#26607 fix
2004-02-12	Alberto Massari	Bug#21965: A substitution group with no type is always valid
2004-02-12	David Cargill	SCMPrint build error fix
2004-02-12	Alberto Massari	Xercesc2_5_0: com updates
2004-02-12	Erik Rydgren	Implemented setTextContent
2004-02-11	PeiYong Zhang	Project SUMPrint
2004-02-11	PeiYong Zhang	PSVIWriter to build with intel
2004-02-11	David Cargill	PSVIWriter to build on AIX
2004-02-11	David Bertoni	Bug#26648 fix
2004-02-11	David Cargill	Bug#26131fix.
2004-02-11	David Cargill	Bug#25541fix

2004-02-10	David Cargill	PSVIWriter build fix
2004-02-09	C-J Berg	Bug#20684 fix
2004-02-09	David Cargill	PSVIWriter build fix and usage update
2004-02-06	PeiYong Zhang	Project PSVIWriter
2004-02-06	David Cargill	Misc 390 changes.
2004-02-06	David Cargill	Intrinsic transcoding support for 390.
2004-02-05	David Cargill	Fix a seg fault with PSVI and set basetype of anysimpletype to be anytype.
2004-02-05	David Cargill	Code cleanup changes to get rid of various compiler diagnostic messages.
2004-02-04	Alberto Massari	Added support for the Interix platform (Windows Services for Unix 3.5)
2004-02-04	Berin Lautenbach	Bug#26426 fix
2004-02-03	PeiYong Zhang	Bug#26315 fix
2004-02-03	PeiYong Zhang	put back the parameter to build debug
2004-01-31	David Cargill	Update script to flush buffer
2004-01-29	David Cargill	Update sanity tests so that each test result can be uniquely identified
2004-01-29	David Cargill	Code cleanup changes to get rid of various compiler diagnostic messages.
2004-01-28	James Berry	Add include for unistd.h
2004-01-28	James Berry	Define away some gcc garbage so that /usr/include/unistd.h will compile with the CodeWarrior MachO target
2004-01-26	James Berry	Add a check for a corner-case buffer condition
2004-01-25	James Berry	Bug #26419 fix
2004-01-25	James Berry	Step around CodeWarrior compiler warning
2004-01-25	James Berry	Update Mac OS Xcode project to reflect recent file additions
2004-01-25	James Berry	Update Mac OS CodeWarrior project to reflect recent file additions
2004-01-21	PeiYong Zhang	Bug#25751fix
2004-01-19	Alberto Massari	WideCharToMultiByte and MultiByteToWideChar return 0 on failure, not -1
2004-01-16	PeiYong Zhang	maintain the same size on both 32/64 bit architecture
2004-01-16	PeiYong Zhang	Project XSerializerTest
2004-01-16	Alberto Massari	In the Win32LCPTranscoder, don't use wcstombs or mbstowcs, as they don't pick up the correct local code page; use the Win32 API using CP_ACP as the code page

2004-01-16	Alberto Massari	Removed usage of undeclared macro MIN
2004-01-15	PeiYong Zhang	proper allignment for built-in datatype read/write
2004-01-15	Khaled Noaman	HP compiler (after upgrade) is no longer complaining about placement delete
2004-01-15	Alberto Massari	Bug#18341 fix
2004-01-15	Michael Wuschek	Bug#24929 fix
2004-01-13	PeiYong Zhang	revert code back to previous version
2004-01-13	Kahled Noaman	Remove unnecessary local static data
2004-01-13	PeiYong Zhang	set optimization level#2 on hp aCC
2004-01-13	David Cargill	Misc build updates
2004-01-13	David Cargill	Undo previous change memory management changes.
2004-01-13	David Cargill	Misc memory management changes
2004-01-13	Khaled Noaman	For sanity, use class name to qualify method
2004-01-13	Khaled Noaman	Fix wrong size of allocation
2004-01-12	Neil Graham	remove unused static member
2004-01-12	Neil Graham	update Copyright year
2004-01-12	David Cargill	Minor performance change for handling reserved and unreserved characters.
2004-01-12	David Cargill	Fix 390 compilation errors.
2004-01-12	Khaled Noaman	Use a global static mutex for locking when creating local static mutexes instead of compareAndSwap
2004-01-12	Neil Graham	remove use of static buffers
2004-01-12	David Cargill	Avoid throwing malformedurl exceptions in XMLURL to avoid a threading problem on AIX.
2004-01-06	Khaled Noaman	PSVI: inherit facets from base type
2004-01-06	Neil Graham	Bug#25660 fix
2004-01-06	Joanne Bogart, Neil Graham	Bug#25542 fix
2004-01-06	Alberto Massari	Bug#25768 fix: Replaced the call to wcstombs using a NULL target buffer with the equivalent call to WideCharToMultiByte ()
2004-01-06	Jeroen Witmond, Neil Graham	Bug#25412 fix
2004-01-06	Khaled Noaman	Fix segfault when adding S4S
2004-01-06	Khaled Noaman	Reset list of grammars after building XSModel
2004-01-06	PeiYong Zhang	using the no-exception-thrown ctor
2004-01-06	Reid Spencer, Neil Graham	Bug#28517 static initialization problems
2004-01-06	Neil Graham	make sure locally-declared attributes have declarations in the PSVI

2004-01-06	Neil Graham	On some platforms, it is problematic to throw a different exception from inside the catch block of another exception
2004-01-05	Khaled Noaman	Various PSVI fixes
2004-01-03	PeiYong Zhang	using ctor/parseContent to avoid exception thrown from ctor
2003-12-31	David Cargill	Update AIX build to generate .a libraries as -brtl is no longer used so that a build generated without using packageBinaires.pl will be clean.
2003-12-31	David Cargill	Release memory when an error is encountered.
2003-12-31	Alberto Massari	Updated project for BCC551
2003-12-31	Alberto Massari	Made virtual function checkAdditionalFacet 'const', so that it matches the declaration in a derived class
2003-12-30	Neil Graham	Even if the resolver has no grammars, since all schema processors are aware of the schema-for-schemas, an XSModel should be produced.
2003-12-30	Neil Graham	fix one more buffer overrun, affecting boolean lists
2003-12-30	Neil Graham	enable production of canonical representations for dates with negative years, or years >9999
2003-12-30	Neil Graham	ensure an XSModel
2003-12-30	Neil Graham	even if there are no grammars to add to an XSModel, the S4S grammar must be included
2003-12-30	Neil Graham	do not report anything about default/fixed values for non-global attribute declarations
2003-12-30	Neil Graham	use a null-terminated string when tokenizing pattern facets
2003-12-30	Neil Graham	more PSVI bug fixes
2003-12-30	Neil Graham	some indices in the PSVIAttributeList were 1 off
2003-12-30	Neil Graham	fix segfault when validation of a union type fails
2003-12-30	Neil Graham	initialize undeclared attribute registry appropriately for its local use in scanStartTag
2003-12-30	Neil Graham	allow schema normalized values to be associated with a PSVIAttribute after it is reset
2003-12-29	PeiYong Zhang	use the original memory manager to deallocate in assignment operator
2003-12-29	Khaled Noaman	PSVI: return value constraint only if global declaration

2003-12-29	Khaled Noaman	PSVI: add whitespace facet if missing
2003-12-29	Khaled Noaman	More PSVI updates
2003-12-29	Alberto Massari	When parsing a new element, clear the maps holding the unparsed attribute we have seen. This because these maps keep pointers to the name of the attributes object that gets recycled for every element (and their name is deallocated when recycled)
2003-12-24	David Cargill	Memory management update.
2003-12-24	David Cargill	Improved algorithm for finding derivedFrom.
2003-12-24	David Cargill	More updates to memory management so that the static memory manager.
2003-12-23	PeiYong Zhang	Absorb exception thrown in getCanonicalRepresentation and return 0, only validate when required
2003-12-22	Michael Glavassevich	Bug#18611 fix.
2003-12-22	Gareth Reakes	Bug #25699 fix: made getRootElemID const.
2003-12-22	Jeroen N. Witmond, Gareth Reakes	Bug#25164: Patch for doc enhancement
2003-12-20	Neil Graham	add attribute names to PSVIAttributeList and fix some problems with calculation of canonical values in element content
2003-12-20	Neil Graham	fix canonical representation production
2003-12-20	Neil Graham	store name/namespace of corresponding attribute in PSVIAttributeList; not all PSVIAttributes have XSAttributeDeclarations
2003-12-19	David Cargill	Fix compiler messages on OS390.
2003-12-19	David Cargill	More memory management updates.
2003-12-19	Khaled Noaman	PSVI: process 'final' information
2003-12-19	Neil Graham	when validating a skipped element or attribute, we should not look for a declaration.
2003-12-19	Neil Graham	remove a throw clause inserted during debugging
2003-12-18	PeiYong Zhang	do not assert memorymanager in placement delete.
2003-12-17	David Cargill	Fix AIX compiler error.
2003-12-17	Khaled Noaman	PSVI: Use complex type info if present, otherwise use datatype validator
2003-12-17	Khaled Noaman	PSVI: fix for annotation of attributes in attributeGroup/derived types
2003-12-17	Neil Graham	fix two overflow conditions

2003-12-17	Neil Graham	fix a segfault and a possible buffer overflow condition
2003-12-17	David Cargill	Update for memory management so that the static memory manager (one used to call Initialize) is only for static data.
2003-12-17	Khaled Noaman	Check for NULL when building XSParticle
2003-12-16	Neil Graham	fix compilation error
2003-12-16	Neil Graham	ensure all uses of ArrayJanitor use a memory manager
2003-12-16	Neil Graham	add default memory manager parameter to loadMsg method that uses char * parameters
2003-12-16	David Cargill	Fix memhandlertest failure (memory not deleted).
2003-12-16	PeiYong Zhang	XSerializerTest updates
2003-12-16	PeiYong Zhang	exception thrown upon invalid number, thanks Gareth Reakes.
2003-12-16	Khaled Noaman	Make IC_Field stateless, fMayMatch is no longer a data member of IC_Field
2003-12-16	Khaled Noaman	Add nextElementKey method
2003-12-		BinMemOutputStream
2003-12-16	Alberto Massari	The DOMTypeInfo should have a NULL namespace and type name when DTD validation is used, not empty strings
2003-12-16	PeiYong Zhang	don't expand ContextSpecNode when deserilized
2003-12-16	Steve Dulin, Neil Graham	update
2003-12-16	David Cargill	Change a conditional expression to an if-else to avoid a compiler problem.
2003-12-15	Neil Graham	fix segfault when a writeAnnotation() method was called
2003-12-15	David Cargill	psvi updates; cleanup revisits and bug fixes
2003-12-14	Neil Graham	make use of XMLDocumentHandler::elementTypeInfo instead of non-thread-safe XMLElementDecl methods
2003-12-13	Han Ming, Neil Graham	Bug#25494 fix
2003-12-13	Neil Graham	fix compilation errors under gcc
2003-12-13	Neil Graham	configure scripts need to be told about XSerializerTests before they can build its Makefile
2003-12-12	Michael Glavassevich, Neil Graham	fix small bugs that made sanity tests fails
2003-12-12	PeiYong Zhang	Project XSerializerTest

2003-12-11	PeiYong Zhang	trailing zeros for double/float w/o decimal point
2003-12-	Michael Glavassevich, Neil Graham	fixes for the URI implementation to take registry names into account
2003-12-11	PeiYong Zhang	Canonical Representation Support
2003-12-11	Khaled Noaman	Store non schema attributes from parent in XSAnnotation
2003-12-10	Steve Dulin, Neil Graham	OS/390 updates
2003-12-10	Steve Dulin, Neil Graham	make documentation accord with what Xerces-C supports
2003-12-10	Steve Dulin, Neil Graham	ICU has deprecated the -s390 encoding suffix. This patch uses the new convention for XML documents that make use of this suffix
2003-12-10	Steve Dulin, Neil Graham	make CreateDOMDocument sample more robust
2003-12-10	Neil Graham	fixes for canonical value production; memory management was not implemented correctly
2003-12-10	Stephen Dulin	Eliminate the preparsing stage
2003-12-10	Neil Graham	change some hash constants
2003-12-10	Neil Graham	fix seg fault caused when a complex type had simple content; we were not processing the complex type itself, only its base
2003-12-09	James Berry	Remove GCC2 build styles from xcode samples
2003-12-09	Han Ming Ong	Bug #25343 Add xcode project for SEnumVal sample
2003-12-09	Han Ming Ong	Bug#25337: Enable DEPLOYMENT_POSTPROCESSING for Mac OS X GCC3 Deployment Build.
2003-12-07	Neil Graham	fix bug in PSVI where a segfault would occur if an attribute was not defined
2003-12-07	Neil Graham	fix duplicate attribute detection when namespaces are disabled
2003-12-03	Neil Graham	PSVI fix: cannot allow validator to reset its element content buffer before exposing it to the application
2003-12-03	Neil Graham	uninitialize panic handlers so they will be ready for subsequent initalizations
2003-12-03	Pete Lloyd, Neil Graham	when an empty element is valid, it will not have a datatype validator
2003-12-02	Jeroen Whitmond, Neil Graham	Bug#25118, additional fix once again

2003-12-02	Neil Graham	since there are certain things, such as schemaLocation attributes, that have a datatype and which we nonetheless do not validate, make canonical-value production dependent on validity being valid
2003-12-02	Pete Lloyd, Neil Graham	fix for ArrayIndexOutOfBoundsException in PSVIAttributeList
2003-12-02	Alberto Massari	Bug#20169, openFile was opening the file for read and write, while only read was required
2003-12-02	Alberto Massari	Removed /version option from the linker
2003-12-02	Alberto Massari	Don't use the word "exception" as a variable name, as VC 7.1 complains about it
2003-12-02	Alberto Massari	Bug#16055 fix, Make the code compilable on Windows when UNICODE is defined
2003-12-01	Khaled Noaman	Properly set fAnnotation data member

3
Releases Archive

Release Information of earlier releases

- Release Information of Xerces-C++ 2.4.0
- Release Information of Xerces-C++ 2.3.0
- Release Information of Xerces-C++ 2.2.0
- Release Information of Xerces-C++ 2.1.0
- Release Information of Xerces-C++ 2.0.0
- Release Information of Xerces-C++ 1.7.0
- Release Information of Xerces-C++ 1.6.0
- Release Information of Xerces-C++ 1.5.2
- Release Information of Xerces-C++ 1.5.1
- Release Information of Xerces-C++ 1.5.0
- Release Information of Xerces-C++ 1.4.0
- Release Information of Xerces-C++ 1.3.0
- Release Information of Xerces-C++ 1.2.0
- Release Information of Xerces-C++ 1.1.0
- Release Information of Xerces-C++ 1.0.1
- Release Information of Xerces-C++ 1.0.0
- Release Information of Xerces-C++ BETA

Release Information of Xerces-C++ 2.4.0 Dec. 2, 2003

Date	Contributor	Description
2003-12-01	Jeroen Witmond/Neil Graham	fix Doxygen warnings; bug 25118
2003-12-01	Neil Graham	fix bug 28084
2003-11-28	Khaled Noaman	Set root element if not previpusly set, Use memory manager when creating root element name
2003-11-27	Neil Graham	in preparation for stateless DOMTypeInfo for elements
2003-11-27	Neil Graham	Fix state-ful duplicate attribute detection when the integrated scanner is in use and namespaces are off. Also, implement change to PSVIHandler interface to remove prefix passing.
2003-11-27	Khaled Noaman	PSVIElement implementation
2003-11-27	David Cargill	implement writeAnnotation

2003-11-27	Neil Graham	create XSModels if a PSVIHandler has been set on the scanner. Make PSVI production more robust
2003-11-27	Pete Lloyd	implement isSpecified
2003-11-27	David Cargill	fixes for segfaults and infinite loops in schema component model implementation;
2003-11-26	Neil Graham	mark DOMTypeInfo-related methods on XMLElementDecl deprecated since they are not thread-safe
2003-11-26	Vitaly Prapirny	Bug#24983: Proposed changes of bcc.551 and bcb6 project files for forthcoming 2.4.0 release
2003-11-26	PeiYong Zhang	DOMPrint run result updated.
2003-11-26	Neil Graham	more complete implementation of PSVIAttributeList; remove some problematic const-ness
2003-11-26	Khaled Noaman	Store XSModel.
2003-11-25	Neil Graham	remove XMLIBM1047Transcoder's dependence on iostream
2003-11-25	Khaled Noaman	Add a method to return the XSObject mapped to a schema grammar component
2003-11-25	James Berry	Update Mac OS Codewarrior project, Cleanup build errors/warnings from CodeWarrior
2003-11-25	David Cargill	Serialize enclosing complex type, Check for out of memory exception and document updates
2003-11-25	David Cargill	Make XSObjectFactory inherit from XMemory
2003-11-25	David Cargill	Misc. PSVI updates
2003-11-25	David Cargill	Update expected result
2003-11-25	Khaled Noaman	Fix AIX linking error
2003-11-25	James Berry	Add Mac OS project for Xcode, Revise build instructions to reflect deprecation of Project Builder projects, and elevation of the Xcode projects.
2003-11-24	James Berry	Eliminate some compiler warnings concerning comments inside of comments
2003-11-24	Hongguo He	add IBM1047 to the list of intrinsic transcoders
2003-11-24	Neil Graham	allow classes derived from XMLTransService to tailor the intrinsic maps to their taste.
2003-11-24	Khaled Noaman	Reset memory manager in Terminate
2003-11-24	Khaled Noaman	PSVI: finish construction of XSSimpleTypeDefinition

2003-11-24	Gareth Reakes	added in support for xml-declaration feature.
2003-11-24	Adam Heinz	Fix for bug 22917
2003-11-23	Neil Graham	update method documentation
2003-11-23	Neil Graham	DatatypeValidator support for PSVI
2003-11-23	Khaled Noaman	PSVI updates
2003-11-	David Cargill	Enable grammar pools and grammar resolvers to manufacture XSModels. This also cleans up handling in the parser classes by eliminating the need to tell the grammar pool that schema compoments need to be produced.
2003-11-21	David Cargill	Setting the stage for PSVI (element and attribute) implementation: pointing out all places in scanners where validation updates need to occur
2003-11-21	David Cargill	More schema component model implementation, In particular, this cleans up and completes the XSModel, XSNamespaceItem, XSAttributeDeclaration and XSAttributeGroup implementations.
2003-11-21	Khaled Noaman	PSVI: Use XSObjectFactory to create various components.
2003-11-21	Alberto Massari	insertElementAt was not checking if there was room for the new element (bug#24714)
2003-11-21	Jeroen Witmond	Wrong filename in error messages (bug#24883)
2003-11-21	Alberto Massari	Protect getEntityDeclPool from invoking a method on a NULL pointer (bug#24881)
2003-11-21	Alberto Massari	Updated COM for xerces-c_2_4_0
2003-11-21	Alberto Massari	Updated project to copy xerces-c_2_4_0, not xerces-c_2_3_0
2003-11-20	Khaled Noaman	PSVI: element declaration, content model, PSVIutil
2003-11-20	Alberto Massari	Updated Borland makefile
2003-11-20	Neil Graham	PSVI: store name and namespace information
2003-11-20	PeiYong Zhang	build xercesc2_4_0 with icu2.6.1
2003-11-19	Neil Graham	increment version to 2.4.0 in docs
2003-11-19	PeiYong Zhang	build xercesc2_4_0
2003-11-17	Pete Lloyd, Neil Graham	PSVIAttributeList needs to be included by PSVIHandler
2003-11-17	PeiYong Zhang	Fix to #4556
2003-11-17	Ronald Landheer-Cieslak	Fix for bug 23930
2003-11-17	Alberto Massari	Fixed documentation bug#24746

2003-11-14	David Cargill	changes in support of second phase of XSModel implementation
2003-11-14	David Cargill	removed methods made unnecessary by new XSModel implementation design
2003-11-14	Neil Graham	PSVI updates
2003-11-14	Alberto Massari	When invoking resolveEntity, specify the current document as the base URI
2003-11-14	Graham Bennett	Fix to bug #4556
2003-11-13	PeiYong Zhang	Pass correct initSize to container during deserialization
2003-11-13	PeiYong Zhang	Solve Compilation/Linkage error on AIX/Solaris/HP/Linux
2003-11-12	PeiYong Zhang	Stateless Grammar: Validation Context
2003-11-11	Khaled Noaman	Serialization of XSAnnotation.
2003-11-10	Neil Graham	implementation for new stateless means of traversing attribute definition lists
2003-11-10	Alberto Massari	Fixed memory leak
2003-11-08	Abe Backus	fix for bug 24287
2003-11-07	David Cargill	PSVI/schema component model implementation
2003-11-07	David Cargill	fix compilation errors on AIX and HPUX
2003-11-07	Khaled Noaman	For PSVI support, distinguish wildcard elements with namespace lists.
2003-11-06	Neil Graham	update KEYS file with public key as newly-signed by two other Apache-ites
2003-11-06	James Berry	Add Mac OS X DYLD_LIBRARY_PATH notes to unix build instructions
2003-11-06	Neil Graham	update grammar pool interface so that cacheGrammar(Grammar) can tell the caller whether the grammar was accepted. Also fix some documentation errors.
2003-11-06	PeiYong Zhang	Patch to Solaris compiler error
2003-11-06	Khaled Noaman	PSVI support for annotations.
2003-11-06	David Cargill	first part of PSVI/schema component model implementation.
2003-11-05	PeiYong Zhang	Grammar Pool Specification updates
2003-11-05	PeiYong Zhang	
2003-11-05	PeiYong Zhang	don't serialize built-in baseValidator, and don't serialize localName/uriName
2003-11-04	Alberto Massari	When invoking resolveEntity, specify the base URI

2003-11-04	Alberto Massari	When loading a grammar that's going to be cached, re-use the grammars already in the cache
2003-11-03	Alberto Massari	A version of lastIndexOf would crash the application if the character to be searched was not found in the string
2003-11-01	Alberto Massari	Updated BCB6 project
2003-10-31	PeiYong Zhang	Serialization test fix
2003-10-30	David Cargill	Enhanced Entity Resolver Support.
2003-10-29	PeiYong Zhang	GrammarPool serialization/deserialization
2003-10-29	PeiYong Zhang	Support for Template serialization/deserialization added
2003-10-29	PeiYong Zhang	XObjectComparator/XTemplateComparator
2003-10-27	James Berry	Add comment regarding permissible values for XML_PLATFORM_NEW_BLOCK_ALIGNMENT.
2003-10-24	David Cargill.	Fix for bug #24207
2003-10-23	Khaled Noaman	Fix memory leak
2003-10-22	Khaled Noaman	Annotation support
2003-10-21	Alberto Massari	Fixed memory leak [bug 23073]
2003-10-21	Alberto Massari	Update COM project files
2003-10-21	PeiYong Zhang	update XercesLib.mak
2003-10-21	Alberto Massari	Inside a schema, the properties "fixed" and "default" for a reference to an attribute were ignored unless the "required" property were also present [bug 11767]
2003-10-20	Khaled Noaman	Fix multithreading problem
2003-10-20	Gareth Reakes	Pass in memory manager to constructors and use for creation of enumerators.
2003-10-18	James Berry	Open files for reading as "r", not "r+". on MacOS
2003-10-18	PeiYong Zhang	Support for Template class serialization/deserialization
2003-10-17	Khaled Noaman	Fix multithreading problem for regular expression.
2003-10-15	PeiYong Zhang	Implementation of Serialization/Deserialization for Schema components
2003-10-10	Neil Graham	update XSModel and XSObject interface so that IDs can be used to query components in XSModels, and so that those IDs can be recovered from components
2003-10-10	PeiYong Zhang	Implementation of Serialization/Deserialization for Grammar components

2003-10-09	David Cargill	fix for bug 21780
2003-10-09	Neil Graham	Synchronized StringPool for thread-safe updates.
2003-10-08	PeiYong Zhang	Synchronize ContentSpec/ContentModel/FormattedModel
2003-10-07	PeiYong Zhang	API for Template_Class Object Serialization/Deserialization
2003-10-07	David Cargill.	Fix #23413
2003-10-06	Mike Pawlowski	Rewrite packageSources
2003-10-04	Neil Graham	Stateless Grammar
2003-10-02	PeiYong Zhang	Implementation of Serialization/Deserialization for Datatype Validators
2003-10-02	Gareth Reakes	Removed ^Z from end of files which was preventing compilation under gcc 2.96.
2003-10-01	David Cargill.	improve handling of out of memory conditions, bug #23415
2003-10-01	Khaled Noaman	Refactoring of some code to improve performance.
2003-09-26	PeiYong Zhang	Synchronize ContentSpecNode and formattedModel
2003-09-26	David Cargill	fix for bug #23427
2003-09-25	PeiYong Zhang	Loose the assert condition so that Serializable class need NOT to check the actual string length before read/write.
2003-09-24	Alby	useImplementation should use memory maneger.
2003-09-23	PeiYong Zhang	Inplementation for Serialization/Deserialization
2003-09-23	PeiYong Zhang	Macro re-organized: provide create/nocreate macros for abstract and nonabstract classes
2003-09-22	Neil Graham	change Grammar::putElemDecl(XMLElementDecl, bool) so that it does not require the Grammar object to be const. Also, mark findOrAddGrammar as being dangerous in multithreaded situations
2003-09-22	Gareth Reakes	doc fix
2003-09-18	PeiYong Zhang	OSU: Object Serialization Utilities
2003-09-18	Gareth Reakes	updated the distribution directory.
2003-09-16	Neil Graham	make Grammar pool be responsible for creating and owning URI string pools. This is one more step towards having grammars be independent of the parsers involved in their creation
2003-09-16	Neil Graham	PSVI/schema component model classes

2003-09-12	Jay Hansen	enable MemParse to work on OS400.
2003-09-10	Neil Graham	fix compiler warnings on ISeries; add Apache copyright notice
2003-09-08	PeiYong Zhang	Restore pre2.3 constructors
2003-09-06	Dave Bertoni	Fix bug #22938. Deletion of void* is illegal.
2003-09-04	Gareth Reakes	Fix for bug #22008. Removed the ability to adopt the DOMObject.
2003-09-04	David Cargill.	Fix bug #19605. Problem with CDATA END TAG
2003-09-01	Gareth Reakes	added API to get an enumerator for the cached grammars.
2003-08-31	Shin'ya Morino.	Fix for bug 21990
2003-08-29	Gareth Reakes	If a type was explicitly declared as anyType that now gets set in DOMTypeInfo. Added test cases.
2003-08-27	James Berry	Add new static global that always points to array-allocating memory manager
2003-08-27	Gareth Reakes	Fixed a bug where multiple invalid elements with the same name/uri were not being set with appropriate PSVI info. Added a test case to expose the problem.
2003-08-26	James Berry	Add new memory allocator that allocates using new[], for use where returned memory must be able to be deleted using delete []. This saves duplicated code in cases where a routine is optionally called with a specific memory manager, such as in the case of transcode
2003-08-26	Neil Graham	fix compilation errors on HPUX and Solaris
2003-08-26	benoit.blaquiere@ign.fr	Fix bug #22697; transcodeFrom incorrectly throws on kTECOutputBufferFullStatus.
2003-08-25	Alberto Massari	fix for bug 22178
2003-08-22	Alberto Massari	keep the fGrammarFromPool in sync to avoid problems when parseing multiple times.
2003-08-22	Gareth Reakes	Not all unknown attributes are faulted in. In these cases the DOMTypeINfo should report AnySimpleType, not AnyURI as they were.
2003-08-21	Neil Graham	add the Apache license to various Perl scripts that did not have it
2003-08-21	PeiYong Zhang	use PlatformUtils::panic()

2003-08-20	Neil Graham	Added a method for use in XercesDOMParser (and others derived from AbstractDOMParser) and a feature in DOMBuilder that allows the creation of the document during parse to be from an DOMImplementation other than the default.
2003-08-20	Gareth Reakes	Changed constuctors to protected to be derivatable
2003-08-20	Gareth Reakes	Reorderd initializer list to prevent compiler warning.
2003-08-20	Steven White	A basic perl script that takes a DOM header file and creates the format used in HTML bindings file.
2003-08-20	Gareth Reakes	Added Level 3 XPath interfaces.
2003-08-20	David Cargill	fix for bug 22565
2003-08-19	David Cargill	fixing bug 21001
2003-08-19	Neil Graham	fix for bug 22537
2003-08-16	Neil Graham	fix for bug 22457. Union types that are restrictions of other union types were previously considered not to inherit their parents member types. This is at variance with the behaviour of the Java parser and apparently with the spec.
2003-08-14	Gareth Reakes	Method added to allow serilization of custom nodes from derived classes.
2003-08-14	Vitaly Prapirny	patch for bug 16933
2003-08-13	Khaled Noaman	Code refactoring to improve performance of validation.
2003-08-13	Alberto Massari	fix to bug 22177
2003-08-13	David Cargill	fix for bug 20058
2003-08-12	Caroline Rioux.	Added serialization for attribute nodes
2003-08-08	Steve Dulin.	fixes to make OS390PlatformUtils.cpp compile
2003-08-07	Neil Graham	fix segmentation faults that may arise when the parser throws exceptions during document parsing. In general, XMLPlatformUtils::Terminate() should not be called from within a catch statement.
2003-08-04	Zeid Derhally	Update Win32 CodeWarrior project for recent file additions/deletions;
2003-08-04	James Berry	Update Mac OS ProjectBuilder project for recent file additions/deletions
2003-07-31	James Berry	Resolve bug #21623; document that XMLParsePath... routines may fail if the file doesn't exist
2003-07-31	PeiYong Zhang	GrammarPool

2003-07-28	Steve Dulin	fix to permit the samples source to be copied from a non-writable to a writable part of the filesystem, then compiled. Binaries will also be dropped in a writable directory. This should make experimentation easier on multi-user systems.
2003-07-25	Michael Glavassevich	The patch fixes Bugzilla #19787, #20006, #20009, #20010 and #20287, and several other issues.
2003-07-24	Michael Glavassevich	Fix for bug #20005
2003-07-24	David Cargill	Patch for bug #20530 - Attributes which have the same expanded name are not considered duplicates.
2003-07-24	Erik Rydgren	getTextContent fix
2003-07-22	Steven White.	Fix build under VC7
2003-07-21	June Ng	fixing bug 21573
2003-07-17	Pedro Lopes and Vitaly Prapirny	fix for bug 18860
2003-07-16	PeiYong Zhang	Documentation on system call, strtod
2003-07-14	Vitaly Prapirny and Anthon Pang	patch to bug 20353
2003-07-14	Abe Backus	patch to bug 21527
2003-07-10	PeiYong Zhang	Stateless Grammar: create grammar components with grammarPool's memory Manager
2003-07-10	PeiYong Zhang	Stateless Grammar: Initialize scanner with grammarResolver
2003-07-04	PeiYong Zhang	specify library with version on AIX
2003-06-26	PeiYong Zhang	GrammarPool
2003-06-23	PeiYong Zhang	to solve unresolved symbol on Solaris
2003-06-23	PeiYong Zhang	clean up temporary XMLGrammarDescription to make MemoryTest happy
2003-06-20	PeiYong Zhang	Stateless Grammar Pool :: Part I
2003-06-16	Tuan Hoang	update xerces-c.spec file
2003-06-10	James Berry	Add support to threadtest for Mac OS X
2003-06-09	James Berry	Add DYLIB_LIBRARY_PATH directions for running samples under Mac OS X.
2003-06-06	Robort Buck	Bug#20552 Updated VC7 Project files.
2003-06-03	PeiYong Zhang	for build on WinXP.NET and Intel Electron
2003-06-02	Neil Graham	new test for the pluggable memory management mechanism.
2003-06-02	Berin Lautenbach	fix for bug #20092
2003-06-02	Berin Lautenbach	Bug 20413 Xerces 2.3 does not compile under NetBSD 1.6
2003-05-30	Alberto Massari	Fixes so we compile under VC7.1.

2003-05-30	Gareth Reakes	Use new macros for iostream.h and std:: issues.
2003-05-30	Sean McInerney	fix to bug #20350. Fix 2 typos.
2003-05-29	Gareth Reakes	fixed typo for version number
2003-05-29	Khaled Noaman	Fix memory leak when using deprecated dom.
2003-05-29	Nathan Codding	Fix to bug #16817. Non leaf nodes and attributes now get notified of release
2003-05-29	Gareth Reakes	fix to bug #20325. Removed unused file and updated Projects.
2003-05-29	Gareth Reakes	Added macros in so we can determine whether to do things like iostream as opposed to iostream.h and whether to use std:: or not.
2003-05-28	Neil Graham	update copyright notice
2003-05-27	Neil Graham	upload public key used for signing releases
2003-05-27	Michael Glavassevich	fix typo that could have been impacting correct operation of reference counting.
2003-05-26	PeiYong Zhang	Use memory manager embedded in rather than the one passed in to de-allocate memory.
2003-05-24	Neil Graham	fix segfault on GCC 2.9x. The depreacted DOM attribute implementation had a cute trick where a void * field could be either a NodeChild pointer or a DOMString; the latter played havoc with the new memory management paradigm. Now a union of a DOMString * and a ChildNode * is used.
2003-05-22	Neil Graham	make GCC happy and make it clearer what we actually use alignPointerForNewBlockAllocation() for in our code.
2003-05-22	James Berry	Move pointer alignment functionality into XMLPlatform header; revise XMemory and DOMDocumentImpl to return blocks aligned by this function
2003-05-22	PeiYong Zhang	removal of isOwnerDocSingleton
2003-05-22	PeiYong Zhang	Build memory manager on hp
2003-05-22	Neil Graham	PanicHandler interface should not inherit from XMemory
2003-05-22	Gareth Reakes	Removed usage of std to compile under gcc and other platforms
2003-05-22	Magnus Strand	Fix another case where use of fallback characters during transcode could cause undesired failure of transcode

2003-05-21	James Berry	Ensure proper block alignment for blocks allocated with XMemory new operators
2003-05-21	Khaled Noaman	Handle allocation of document types not created by a DOM document
2003-05-21	Khaled Noaman	Fix to HP-UX compiler's complaint about the duplicate overload of delete
2003-05-21	Khaled Noaman	fix to gcc 2.95.x internal error for some template definitions
2003-05-21	PeiYong Zhang	release document
2003-05-21	James Berry	Mac OS LCP transcoder fix
2003-05-21	James Berry	CodeWarrior Mac OS Project File updates
2003-05-21	Zeid Derhally	CodeWarror Win32 Project File updates
2003-05-21	James Berry	First cut at Mac OS X Project Builder changes
2003-05-20	PeiYong Zhang	Apply Memory Manager to Base64
2003-05-20	Khaled Noaman	Initialize ValueVectorOf
2003-05-19	Gareth Reakes	NetBSD compilation fix

Release Information of Xerces-C++ 2.3.0: May 23, 2003

Date	Contributor	Description
2003-05-16	Khaled Noaman	Configurable Memory Management
2003-05-15	Gareth Reakes	Partial Document::normalizeDocument() Implementaion
2003-05-15	Nathan Codding	Optimization. We now resize the hash when appropriate.
2003-05-14	Alberto Massari	Fix to problem with multiple default namespace attributes being serialized
2003-05-14	Hiramatsu Yoshifumi	port to NetBSD
2003-05-13	Neil Graham	Fix a bug that caused ComplexTypeInfo#elementCount() to report values including references to global elements only when the schema-full-checking flag was true
2003-05-12	Alberto Massari	[Bug 18832] Corrected serilization with regards to namespace nodes
2003-05-10	Zeid Derhally	Fix bugs 19816, 19817, 19818;
2003-05-06	Neil Graham	Fix GCC compilation problem and incorrect #include
2003-05-05	Urs Muff/Neil Graham	Adding optional support for reference counting of nodes within the DOM
2003-05-01	James Devries	Socket support added on OS400
2003-04-30	Khaled Noaman	MemoryManager and XMemory
2003-04-30	Andrew Hefford	[Bug 19472]Spelling mistake correction.
2003-04-29	Khaled Noaman	Cut link to XMLBigInteger
2003-04-28	Neil Graham	Implement namespaces 1.1
2003-04-28	Ailian Ding	[Bug 19402] OS2PlatformUtils.cpp compareAndSwap() need to return retVal.

2003-04-28	James Berry	Add function prototype to eliminate compiler warning
2003-04-27	James Berry	Add new files to Mac OS CodeWarrior project
2003-04-27	James Berry	Add include for stdlib to pull in size_t declaration
2003-04-27	James Berry	Add new files to Mac OS ProjectBuilder projects
2003-04-27	James Berry	PanicHandler, GetCurrentDir() and isAnySlash() on MAC
2003-04-25	Khaled Noaman	Replicate key2 and key3 when putting an item in the list
2003-04-25	PeiYong Zhang	throw exception if getcwd() fails
2003-04-25	Neil Graham	Win32PlatformUtils: use WIN API to make it compilable on Windows with both cygwin and MSVC++
2003-04-24	PeiYong Zhang	Logical Path Resolution
2003-04-22	Khaled Noaman	Initialize security manager in Scanner constructor
2003-04-22	Neil Graham	change const static member (in SecurityManager) to an enum to make MSVC happy
2003-04-21	Khaled Noaman	Use XMLString::release to prepare for configurable memory manager.
2003-04-21	Khaled Noaman	Performance: use memcpy in moveChars and replicate.
2003-04-21	Khaled Noaman	MemoryManager and MemoryManagerImpl
2003-04-21	PeiYong Zhang	Performance tuning to XMLPlatformUtils::getFullPath()
2003-04-17	Neil Graham	new property, http://apache.org/xml/properties/security-manager
2003-04-15	Berin Lautenbach	[Bug 17096] XMLUri relative path calculation badly broken
2003-04-09	Pedro Lopes	[Bug 18860] Samples on Borland C++ 6 - access violations and build errors
2003-04-09	Guido Gagliardi	[Bug 18856] Example code do not compile
2003-04-07	Vasily Tchekalkin	[Bug 18672] IconvGNUTranscoder can't be build when namespaces is on.
2003-04-04	Neil Graham	Update to project file: DOMConfigurationImpl
2003-04-03	PeiYong Zhang	Revised Implementation of getTextContent() to use castToNodeImpl()
2003-04-02	Erik Rydgren	Implementation of getTextContent().
2003-04-02	Neil Graham	Fix to personal.xsd to permit xml:base on on elements
2003-04-01	PeiYong Zhang	[Bug 18594] DOMWriter does not recognize Document Fragment
2003-04-01	PeiYong Zhang	Link in version numbered ICU on AIX
2003-03-31	Gareth Reakes	Changed the API for document normalization to the new Level 3 WD
2003-03-31	Caroline Rioux	DOMConfiguration
2003-03-27	Tinny Ng	use __IBMCPP__ instead of __xlC__ to determine xlC compiler
2003-03-25	Khaled Noaman	Fix typo in program-others.xml
2003-03-24	Tinny Ng	Link in version numbered ICU so that multiple version of XML4C can coexist
2003-03-23	PeiYong Zhang	Invalid second values in XMLDateTime

2003-03-21	Khaled Noaman	Should reset reader manager before returning loaded grammar
2003-03-20	PeiYong Zhang	Fix to 'genrb' on Linux
2003-03-20	Neil Graham	[Bug 12436] Add detection of invalid UTF-8 byte sequences
2003-03-19	Vinayak	Added flag (p) and code to the countChildElements fn to enable printing of nodes and all associated attributes
2003-03-18	PeiYong Zhang	Build versioned shared library, libXercesMessages on UNIX
2003-03-18	Khaled Noaman	Schema Errata E2-18.
2003-03-18	Alberto Massari	[Bug 18063] References to attributeGroup/group definition are not allowed to have annotations
2003-03-17	PeiYong Zhang	Build versioned Message on Windows
2003-03-16	PeiYong Zhang	[Bug 18051] Memory leak in Formatter
2003-03-15	PeiYong	[Bug 17983] Formatter does not escape control characters
2003-03-14	PeiYong Zhang	Copy non-versioned libXercesMessages to target directory
2003-03-14	PeiYong Zhang	Enable to locate libXercesMessage
2003-03-14	Tinny Ng	Change to 2.3
2003-03-14	Tinny Ng	[Bug 17147] C++ namespace breaks build of XercesCOM DLL.
2003-03-13	Chris McKillop	[Bug 17858] Support for QNX/Neutrino.
2003-03-13	Vitaly Prapirny	[Bug 11974] mak-files for bcc32 v.5.5.1 (free or from BCB5).
2003-03-11	PeiYong Zhang	Build versioned dll for ICU message files.
2003-03-11	Khaled Noaman	Schema Fix: Check that target namespace of global/local attribute declarations is not the xsi uri.
2003-03-11	Khaled Noaman	Schema Fix for circular substitution group check.
2003-03-10	Khaled Noaman	Schema Fix for complex type declarations with mixed content.
2003-03-10	PeiYong Zhang	Schema Errata E2-40 double/float.
2003-03-10	Khaled Noaman	Schema Fix for types referred to without explicitly specifying its namespace.
2003-03-10	Tinny Ng	XML1.0 Errata E38.
2003-03-09	PeiYong Zhang	Pluggable PanicHandler.
2003-03-07	Jacques Legare	[Bug 17589] Refactoring ....
2003-03-07	Peter Crozier	[Bug 17774] Unixware platform utils not implemented.
2003-03-07	Tinny Ng	[Bug 11692] Unimplement the hidden constructors and assignment operator to remove warnings from gcc.
2003-03-07	Tinny Ng	Return a reference instead of void for operator=.
2003-03-07	Bjoern A. Zeeb	[Bug 17571] fix building IconvFBSD (namespaces).
2003-03-07	Bjoern A. Zeeb	[Bug 17570] IconvFBSD build on alpha,sparc.
2003-03-06	Alberto Massari	[Bug 17633] Empty complex type definition is always non-mixed even if declaration says otherwise.

2003-03-04	Khaled Noaman	[Bug 17516] Thread safety problems in ../util/ and ../util/regx.
2003-03-04	Khaled Noaman	RegEx: fix for character category escape.
2003-03-01	PeiYong Zhang	Schema Fix: TotalDigits value must be a positiveInteger.
2003-02-26	Khaled Noaman	[Bug 17425] Schema using cyclic import fails validation.
2003-02-26	PeiYong Zhang	Schema Errata E2-43: disallow trailing decimal point and a new pattern added to the Integer definition.
2003-02-25	Tinny Ng	[Bug 12192] File named 'CVS' should be deleted.
2003-02-25	Steve Dulin	Modify UnixHTTPURLInputStream for it to work on ebcdic platform.
2003-02-25	James Berry	Fixes to runConfigure which was worrying overmuch if no C++ compiler was specified. and fix the test for TRU64
2003-02-25	PeiYong Zhang	Schema Errata: E2-44 totalDigits/fractDigits.
2003-02-25	Tinny Ng	[Bug 13491] avoid deleting through void* in DOMDocumentImpl.cpp.
2003-02-25	Tinny Ng	[Bug 7072] Documentation for XMLString::transcode states invalid return value.
2003-02-25	Tinny Ng	[Bug 13493] Use const on static data in DOMWriterImpl.cpp.
2003-02-25	Duncan Stodart	[Bug 12350] Xerces compilation problems on Tandem (HP Nonstop).
2003-02-25	Dan Gohman	[Bug 13492] Unintended comma expression in DOMRangeImpl.cpp.
2003-02-25	Michael Cahill	[Bug 17358] C++ namespace support in IconvFBSD doesn't compile.
2003-02-22	James Berry	Improvements to Mac OS port: - Refactor Mac OS file handling into distinct files per file type. - Add Posix file handling to use posix file apis directly where possible. - Carbon file access is now used only where posix files aren't available. - Tweaks to FSSpec/FSRef routines to handle directories better.
2003-02-22	PeiYong Zhang	Schema Errata E2-35 Length, minLength and maxLength in different derivation steps.
2003-02-21	Neil Graham	Fix packageBinaries.pl so that it pays attention when you set the compiler to gcc under cygwin.
2003-02-21	Neil Graham	[Bug 13429] Text in part of the Programming/Parsing FAQ is truncated/missing.
2003-02-20	PeiYong Zhang	[Bug 7077] build error message shared library for ICUMsgLoader.
2003-02-19	Khaled Noaman	Schema errata E2-38.
2003-02-17	Dan Egnor	[Bug 17131] File writing on Win32 very very very slow.
2003/02/17	PeiYong Zhang	Allow set user specified error message file location in PlatformUtils::Initialize().

2003-02-10	PeiYong Zhang	Remove -weol from the command line option list of sample DOMPrint.

Release Information of Xerces-C++ 2.2.0: February 7, 2003

Date	Contributor	Description
2003-02-06	Khaled Noaman	Schema Errata: 1. E1-2 2. E1-10 3. E1-15 4. E1-16 5. E1-20 6. E1-21 7. E1-22 8. E1-23 9. E1-27
2003-02-06	PeiYong Zhang	Schema Errata: 1. E2-9 Base64. 2. E2-12 gMonth. 3. E2-16 maxExclusive. 4. E2-23 seconds part shall have at least one digit after the dot if it appears. 5. E2-24 Duration 'T': allow SchemaDateTimeException be propogated to client. 6. E2-25 language.
2003-02-06	Khaled Noaman	Performance: Scanner Reorganization. Create XMLScannerResolver, WFXMLScanner, IGXMLScanner, DGXMLScanner, and SGXMLScanner.

2003-02-06	Khaled Noaman	Performance: 1. [Bug 13695] Performance problem with large text nodes and XMLFormatter.cpp. 2. Make getNextChar/peekNextChar inline. 3. Reduce instruction counts in XMLReader. 4. Do not call XMLString::stringLen in XMLString::indexOf. 5. Use existing QName in XMLElementDecl instead of creating a new one everytime. 6. Allow creating/setting of XMLAttr using a rawname (i.e. 'prefix:localpart'). 7. Enable/disable calculation of src offset. 8. No need to use temporary buffer to hold namespace value in SAX2XMLReaderImpl. 9. Eliminate unnecessary condition in compareNString. 10. Use global buffer to eliminate repetitive memory creation/deletion

2003-02-06	Tinny Ng	Performance: 1. DOM: call fParent.fOwnerDocument directly instead of fNode.getOwnerDocument. 2. Check for null string directly isntead of calling XMLString::stringLen. 3. New inline function XMLString::equals that simply returns true or false, use it instead of XMLString::compareString wherever applicable. 4. XERCES_XMLCH should not be classified as XMLRecognizer::OtherEncodings. 5. Pre uppercase the encodingString before calling encodingForName to avoid calling compareIString. 6. Use XMLRecognizer::Encodings enum to make new transcode, faster than comparing the encoding string every time. 7. Reduce some instruction counts in XMLUTF8Transcoder. 8. [Bug 13447] Using LocalFileFormatTarget with DOMWriter is very slow. 9. Define fGlobalDeclarations as an array of ValueVectorOf to avoid string comparison.
2003-02-06	David Bertoni	[Bug 16826] RefVectorOf.c has errors in strict ANSI mode.
2003-02-06	Gareth Reakes	Schema Fix: bug with multiple attributes being validated by the same union type.
2003-02-05	Tinny Ng	[Bug 7592] XMLURL::lookupByName() should be static.
2003-02-05	Tinny Ng	[Bug 3111] Problem with LexicalHandler::startDTD() and LexicalHandler::endDTD().
2003-02-05	Tinny Ng	[Bug 16322] DOMDocumentImpl::replaceChild should honor fDocElement.
2003-02-05	Tinny Ng	[Bug 11915] Utility for freeing memory.
2003-02-05	Tinny Ng	[Bug 13437] Incorrect memory management in XXXPlatformUtils.cpp.
2003-02-05	Zeid Derhally	[Bug 14599] Metrowerks in support of CodeWarrior for Windows.

2003-02-05	PeiYong Zhang	[Bug 16796] Possible out of bounds memory read in XMLRecognizer::basicEncodingProbe.
2003-02-05	Khaled Noaman	[Bug 16747] Parser loses ValidationScheme setting between parse attempts.
2003-02-04	PeiYong Zhang	[Bug 16784] Obsolete documentation on XMLTranscoder
2003-02-04	PeiYong Zhang	[Bug 16652] data from CDATA section is not passed for validation.
2003-01-30	Tinny Ng	[Bug 3041] wrong PLATFORM_IMPORT in MVSCPPDefs.hpp.
2003-01-29	Gareth Reakes	Partial PSVI Support.
2003-01-29	Gareth Reakes	DOM L3: DOMTypeInfo and an associated test case.
2003-01-29	Lenny Hoffman	[Bug 6271] Invalid Precondition Test.
2003-01-29	Khaled Noaman	[Bug 15787] Reduce array size to reduce memory footprint.
2003-01-28	PeiYong Zhang	[Bug 13694]: Allow Xerces to write the BOM to XML files.
2003-01-23	Tinny Ng	[Bug 16188] Consistent crashes with BCB6.
2003-01-23	Tinny Ng	[Bug 16277] Readme should make note of threaded library problems in BCB6.
2003-01-16	David Bertoni	[Bug 16151] Memory leak in DTDScanner with ill-formed DTD declaration.
2003-01-13	Khaled Noaman	[Bug 16024] SchemaSymbols.hpp conflicts C++ Builder 6 dir.h.
2003-01-13	Khaled Noaman	[Bug 14390] C++ Indentifier collision with Python.
2003-01-13	Khaled Noaman	[Bug 14469] Validator doesn't enforce xsd:key.
2003-01-10	Tinny Ng	[Bug 13909] Use of non standard mbstowcs feature.
2003-01-10	Tinny Ng	[Bug 14545] samples/Makefile.incl has bad -L for linux.
2003-01-10	Alberto Massari	[Bug 14912] crashes inside UnionDatatypeValidator::isSubstitutableBy.
2003-01-10	Albert Strasheim	[Bug 5854] Patches and .spec file for rpm creation of 2.2.0.
2003-01-09	Tinny Ng	[Bug 14955] error validating parser.
2003-01-09	Tinny Ng	[Bug 15928] Output with LocalFileFormatTarget fails silently.
2003-01-09	Tinny Ng	[Bug 15371] Fix documentation. The default of schema processing shoud be false.

2003-01-09	Tinny Ng	[Bug 15372] DOMBuilder::parseFromURI ignores result of handleErrors.
2003-01-09	Tinny Ng	[Bug 15802] Add "const" qualifier to getURIText.
2003-01-09	Tinny Ng	[Bug 15427] DOMWriter dose not flush the output stream.
2003-01-09	Colin Adams	[Bug 15796] surroundContents seg-faults.
2003-01-03	Tinny Ng	New feature StandardUriConformant to force strict standard uri conformance.
2002-12-31	Tinny Ng	[Bug 15590] BeOSDefs.hpp has wrong case in CVS.
2002-12-31	Tinny Ng	[Bug 15608] IconvLCPTranscoder::transcode() is wrong at wcstombs() usage.
2002-12-30	Gareth Reakes	Added isDocumentAdopted API and recognize feature fgXercesUserAdoptsDOMDocument in DOMBuilder::getFeature/canSetFeature.
2002-12-24	Tinny Ng	Build with ICU 2.4.
2002-12-24	Tinny Ng	[Bug 15160] TrueCoverage build fails in Window.
2002-12-23	Khaled Noaman	New public api to various parsers to return the src offset within the input source.
2002-12-20	Tinny Ng	XML 1.1
2002-12-19	Peter A. Volchek	Schema: get/set methods to see if the represented type is anonymous.
2002-12-18	Gareth Reakes	[Bug 13438] Mismatched new[]/delete in template vector class. Added new abstract base class BaseRefVectorOf from which both RefVectorOf and the new class RefArrayVectorOf inherit from it.
2002-12-18	Jennifer Schachter	New Regx functionality - tokenize and replace.
2002-12-16	James Berry	[Bug 14805] Mac OS transcoder should return pointer to zero length string (rather than NULL) on receipt of zero length input.
2002-12-10	PeiYong Zhang	Validating Schema Float/Double in value space. Converting out-of-bound value into special values.
2002-12-10	Tinny Ng	NLS: DOMWriter should use message loader to load message instead of using hardcoded static stirng.

2002-12-06	Kevin King	[Bug 13840] DOMWriter: more pretty-print format feature.
2002-12-06	Tinny Ng	Fix: for file protocol, need to manually replace any character reference %xx first.
2002-12-06	Tinny Ng	[Bug 9083] Make some classes be exportable.
2002-12-06	Tinny Ng	[Bug 9697] Make GrammarResolver to be exportable.
2002-12-02	Andrew Bachmann	[Bug 12490] Patches required to build Xerces-C++ on BeOS R5.
2002-12-02	Adam Zell	[Bug 14723] Memory leak in atomicOpsMutex.
2002-12-02	Abe Backus	[Bug 13804] Update build and installation docs for cygwin.
2002-12-02	Peter A. Volchek	[Bug 14960] Opened up interface to expose user defined and built in registries.
2002-12-02	Gareth Reakes and Peter A. Volchek	[Bug 12188] Create NMTOKEN, ID, IDREF, ENTITY, NAME, NCNAME with appropriate base types. Some reordering of creation was required where dependencies resulted.
2002-12-02	Peter A. Volchek	[Bug 12238] Attributes without type declarations should be validated using AnySimpleTypeValidator, not the string validator.
2002-11-26	Tinny Ng	Namespace Check: 1. xmlns:a="" where namespace URI is null is not valid. 2. xmlns:doc where xmlns is used as element prefix is not valid. 3. xmlns:xmlns where xmlns is used as prefix is not valid. 4. xmlns:xml="a" where xml is used as prefix but URI does not match the xml uri (http://www.w3.org/XML/1999/namespace) is not valid. 5. if validation is on, attribute values declared to be of types ID, IDREF(S), ENTITY(IES), and NOTATION are also Names, and thus should be colon-free.
2002-11-25	Tinny Ng	Thread-safe the static variable TransService::gMappings.
2002-11-22	Robert Buck	Add autodetection of MSVC++ version in packageBinaries.pl.

2002-11-22	Chris Larsson and Stephen Dulin	390: support record-oriented MVS datasets with the DOM Level 3 serialization APIs.
2002-11-22	Chris Larsson and Stephen Dulin	390: Uniconv390 support.
2002-11-21	Jennifer Schachter	Fixed bug in Token::analyzeFirstCharacter so that . matches new line with head character optimisation enabled.
2002-11-21	Gareth Reakes and Jennifer Schachter	DOM L3: isId, setIdAttribute, setIdAttributeNS and setIdAttributeNode.
2002-11-21	PeiYong Zhang	Schema Fix: validate content as a whole against pattern.
2002-11-20	PeiYong Zhang	Update ThreadTest to use DOMWriter to dump DOM.
2002-11-19	Tinny Ng	[Bug 13487] Linux runs on many non-i386 platforms.
2002-11-19	Cameron Dorrat	[Bug 14661] Caldera implemented openFileToWrite and writeBufferToFile.
2002-11-18	Steven White	Problems using make tarball under linux.
2002-11-18	Abe Backus	[Bug 14612] GCCDefs clashes with cygwin's string.h for stricmp and strnicmp.
2002-11-15	Abe Backus	[Bug 13801] cygwin libxerces-c.dll symlinks misleading.
2002-11-15	Tinny Ng	[Bug 13751] Documentation for DOMNamedNodeMap incorrect.
2002-11-15	Richard Balint	[Bug 14598] IRIX 6.5 / g++ 3.0.4 compilation bugs.
2002-11-14	Tinny Ng	[Bug 14265] Access violation with Null systemId/publicId in DTDScanner.
2002-11-14	Tinny Ng	[Bug 14479] XMLString::subString failure when len(source)==0.
2002-11-14	Tinny Ng	[Bug 14389] DOMPrint - gDoCreate - wrong default value.
2002-11-13	PeiYong Zhang	[Bug 14528] Encounters of the end tag "]]>" are ignored.
2002-11-13	James Berry	[Bug 14260] MacOSUnicodeConverter::upperCase() passes wrong arguments to Carbon function.
2002-11-13	James Berry	Update Mac OS build for compatiblity with namespace additions.
2002-11-12	Tinny Ng	DOM Message: make use of the non-standard extension DOMImplementation::loadDOMExceptionMsg to load the default error text message for the correspond Exception Code.

2002-11-12	Tinny Ng	DOM Message: introduce a new message domain, XMLDOMMsg, for DOM Messages.
2002-11-04	PeiYong Zhang	New feature XMLPlatformUtils::Initialize(const char* const locale) to set the locale for message loader.
2002-11-04	Tinny Ng	C++ Namespace Support
2002-10-30	Tinny Ng	[Bug 13641] compiler-generated copy-constructor for QName doesn't do the right thing.
2002-10-29	Chris Larsson	Modify DOMPrint to accept a file name as a parameter.
2002-10-29	Tinny Ng	Support for Linux/390 which is big endian.
2002-10-23	PeiYong Zhang	[Bug 13213] DOMImplementation::hasFeature() should be const.
2002-10-17	PeiYong Zhang	[Bug 13640] Getter methods not public in DecimalDatatypeValidator.
2002-10-16	Khaled Noaman	[Bug 13293] Schema ID validation can fail depending on declaration ordering.
2002-10-15	Khaled Noaman	[Bug 13604] while loop never terminates.
2002-10-15	Khaled Noaman	[Bug 13639] Failure to parse xsi:schemaLocation attribute value correctly.
2002-10-15	Khaled Noaman	[Bug 13494] use unsigned instead of signed in TraverseSchema.cpp.
2002-10-15	Khaled Noaman	[Bug 13490] new[]/delete mismatch in RangeToken.cpp.
2002-10-15	Khaled Noaman	[Bug 13489] missing 'return' in Token.cpp.
2002-10-15	Khaled Noaman	[Bug 13485] incorrect return from getWSstring.
2002-10-04	Duncan Stodart	[Bug 12560] Use const in DOMWriter.
2002-10-01	Tinny Ng	[Bug 13139] Building Promblems on HP-UX.
2002-09-30	PeiYong Zhang	Support ICU Message Loader.
2002-09-30	PeiYong Zhang	Xlat: To generate icu resource file (in text) for error message.
2002-09-30	Tinny Ng	[Bug 13109] DOMRange::toString eventually cycles forever.
2002-09-27	Guillaume Morin	[Bug 12547] Xerces C++ 2.1 fails to build on Linux 64 bits arch with -tlinux.
2002-09-27	Tinny Ng	[Bug 13073] GeneralAttributeCheck.cpp : compilation fails with Sun C++ 4.2 on Solaris2.7 system.
2002-09-27	Peter Volchek	[Bug 12740] Extra include.

2002-09-27	Gareth Reakes	[Bug 12847] bulid warning for non-virtual constuctor.
2002-09-27	Gareth Reakes	[Bug 12848] newline warning whist building.
2002-09-26	Gareth Reakes	[Bug 12849] comparison is always false warning.
2002-09-26	Erik Rydgren	[Bug 12914] Bug in AbstractDOMParser::resetPool().
2002-09-26	Gareth Reakes	DOM L3: Add const to isSameNode, isEqualNode, compareTreePosition.
2002-09-23	PeiYong Zhang	Issue Panic_CantLoadMsgDomain if loadAMsgSet() fails.
2002-09-23	PeiYong Zhang	Support MsgCatalog Message Loader.
2002-09-23	Gareth Reakes, Thomas Ford and Tinny Ng	DOM L3: Support baseURI.
2002-09-18	Stephen Dulin	OS390 Performance Enhancement: instead of calling isPosixOn everytime, store the information in a static flag during initialization.
2002-09-17	Thomas Woerner	RPM for Linux.
2002-09-16	Tinny Ng	Infinite loop for malformed xml (e.g. simple has "XXXX") w/ setexitonfirstfatal(false).
2002-09-16	Tinny Ng	[Bug 12442] Fix typo: "Mode:" which should be "Model".
2002-09-09	PeiYong Zhang	[Bug 12369] invalid output from DOMWriter using MemBufFormatTarget.
2002-09-05	James Berry	Add export directives for Mac OS path utility routines .
2002-09-05	Tinny Ng	[Bug 12232] Make operator to be constant.
2002-09-05	Tinny Ng	[Bug 12290] example on webpage won't compile.
2002-09-05	Tinny Ng	[Bug 12279] Makefiles contain tabs causing "commands commence" error.
2002-09-05	Tinny Ng	[Bug 12275] DOMCount -n gives DOM Error.
2002-09-03	Tinny Ng	[Bug 12897] System ID is missing inside DOCTYPE.
2002/08/27	Khaled Noaman	Identity Constraint: handle case of recursive elements.
2002-08-27	Tom Ford	[Bug 12087] XMLString::patternMatch() is not accurate.

Release Information of Xerces-C++ 2.1.0: August 26, 2002

Date	Contributor	Description

2002-08-26	Abe Backus	[Bug 12004] Samples/Tests don't build under cygwin.
2002-08-23	Tinny Ng	[Bug 11981] inproper "AND" operator in AutoSense.hpp
2002-08-23	Tinny Ng	Memory leak fix: enums is not deleted if an error occurred.
2002-08-23	Tinny Ng	Memory leak fix: XMLUri data not deleted if constructor failed.
2002-08-23	James Berry	Begin addition of support for Codewarrior MachO build of Xerces framework.
2002-08-23	James Berry	[Bug 11776] MacOSUnicodeConvertor::upperCase doesn't work correctly.
2002-08-23	Robert Buck	[Bug 11975] Update to XercesLib VC7 Project File.
2002-08-22	Tinny Ng	[Bug 7512] Wrong error message created.
2002-08-22	Tinny Ng	[Bug 11448] DomCount has problems with XHTML1.1 DTD.
2002-08-22	Robert Buck	[Bug 11946] Updated VC7 Project Files for Xerces-C 2.1.
2002-08-22	Khaled Noaman	[Bug 11906] Wrong comparison in TraverseSchema.
2002-08-22	PeiYong Zhang	[Bug 10653] XMLString::parseInt possible overflow.
2002-08-21	Tinny Ng	[Bug 11869] Add the const modifier (XMLBuffer.hpp).
2002-08-21	Tinny Ng	[Bug 7087] compiler warnings when using gcc.
2002-08-20	Benjamin Piwowarski	[Bug 11515] Exponential time using DOMTreeWalker.
2002-08-20	Tinny Ng	[Bug 6251] Info during compilation.
2002-08-19	Vasily Tchekalkin	[Bug 11771] Linux specific IconvGNU transcoder.
2002-08-19	Derek Harmon and Abe Backus	[Bug 6467] Installing Xerces C++ on cygwin environment.
2002-08-19	Tinny Ng	[Bug 11229] bogus -I statements order in CXXFLAGS.
2002-08-19	Tinny Ng	[Bug 1471] getInternalSubset returns NDATA with quote. Also fix internalsubset to include notation.
2002-08-19	Khaled Noaman	[Bug 11770] - Xerces does not validate the XMLSchema's root element name.
2002-08-16	Khaled Noaman	[Bug 7698] Filenames with embedded spaces in schemaLocation strings not handled properly.
2002-08-16	PeiYong Zhang	New Configure: Win64 Debug for samples Project / Makefiles.

2002-08-16	Gareth Reakes	DOM L3: support lookupNamespacePrefix, lookupNamespaceURL, isDefaultNamespace.
2002-08-16	Tinny Ng	[Bug 11360] Release user data using handler.
2002-08-14	Khaled Noaman	[Bug 3111] Problem with LexicalHandler::startDTD() and LexicalHandler::endDTD().
2002-08-13	PeiYong Zhang	[Bug 9442] minInclusive factet validation alters value.
2002-08-13	Khaled Noaman	Recognize UTF16.
2002-08-12	PeiYong Zhang	[Bug 11462] MemBufFormatTarget issues (2) - const-ness, thread-safety.
2002-08-12	Tinny Ng	Support Intel IA32 C++ Compiler, icc.
2002-08-09	Gareth Reakes	DOM L3: support compareTreePosition.
2002-08-08	Stephen Dulin	DOMWriter support on z/OS.
2002-08-08	Tinny Ng	DOM Fix: Recycle node value buffer to avoid memory growth.
2002-08-07	PeiYong Zhang	[Bug 11534] Wrong CDATA Terminator in DOMWriterImpl.
2002-08-07	Khaled Noaman	Pass proper value of actual encoding to XMLDecl callback.
2002-08-01	Khaled Noaman	If the NamespaceURI, qualifiedName, and doctype are null, the returned Document is empty with no document element.
2002-08-01	Khaled Noaman	Ensure that we add only DOM Attr nodes to the attributes NamedNodeMap.
2002-08-01	Khaled Noaman	DOM L2 does not support editing DocumentType nodes.
2002-07-31	Tinny Ng	[Bug 11338] missing const keyword for DOMNodeList methods.
2002-07-31	Tinny Ng	[Bug 6227] Make method getLastExtLocation() constant.
2002-07-31	Tinny Ng	[Bug 3788] very long lines in CppErrMsgs_EN_US.hpp causes problems for OS390 compiler.
2002-07-31	Eric Zurcher	[Bug 11099] BCB6 project for Xerceslib 2.0 has wrong files.
2002-07-31	Tinny Ng	[Bug 6321] gmake error in regx/Makefile.in.
2002-07-30	Tinny Ng	[Bug 8550] No explanation of XMLFormatter escape options.
2002-07-30	Khaled Noaman	Create default attributes with the namespace URI mapped to the attributes' prefixes.

2002-07-29	PeiYong Zhang	Build Xerces with ICU -- Itanium/WinXP/IntelC++Compiler.
2002-07-29	Tinny Ng	[Bug 9084] scripts/packageBinaries -j option not well documented.
2002-07-29	Tom Keane	[Bug 9533] Win32TransService does not recognize aliases for encodings.
2002-07-26	Tinny Ng	Memory Leak in DOMDocumentTypeImpl.
2002-07-26	Joé St-Germain	[Bug 10337] XMLString::patternMatch doesn't find pattern in particular context.
2002-07-26	David Bertoni	[Bug 11189] Tru64 utilities missing implementation of new functions.
2002-07-26	Jonathan Lennox	[Bug 2681] Can't build with gcc/g++ not named 'gcc'/'g++'.
2002-07-26	Khaled Noaman	For a given DOM Element and DOM DocumentType node, explicitly cast to the implementation of that DOM node when calling setReadOnly.
2002-07-26	Khaled Noaman	Public/System id for notations should be stored as NULL if missing.
2002-07-25	Robert Buck	[Bug 11141] Fix To Broken VC7 Builds.
2002-07-25	Khaled Noaman	[Bug 11153] getOwnerDocument() on PI that's child of Document returns NULL.
2002-07-24	Khaled Noaman	Remove check for disallowed encodings - not needed anymore.
2002/07/23	Tinny Ng	Build with ICU 2.2.

Release Information of Xerces-C++ 2.0.0: July 23, 2002

Date	Contributor	Description
2002-07-19	Tinny Ng	[Bug 10968] Default attributes from Schema not restored by removeAttribute.
2002-07-18	Khaled Noaman	Feature to control strict IANA encoding name.
2002-07-18	Tinny Ng	[Bug 9707] config.guess out of date. From AutoConf dated July 18, 2002, CVS Tag AUTOCONF-2_53b.
2002-07-17	PeiYong Zhang	Add Win64 to Windows VC6 Project files
2002-07-16	Tinny Ng	[Bug 6070] warning unused variable in HandlerBase.hpp.
2002-07-16	Tinny Ng	[Bug 6576] Exception on processing UTF-16 InputSource buffer with set encoding.
2002-07-16	Tinny Ng	[Bug 6590] Improper Internal subset filling.
2002-07-16	Alberto Massari	[Bug 7458] Schema validator does not automatically associate the xml prefix to the "http://www.w3.org/XML/1998/namespace" URI.
2002-07-16	Case Larsen	[Bug 9502] purify UMR in DocumentImpl::DocumentImpl.
2002-07-16	Case Larsen	[Bug 9553] purify UMR in XMLRecognizer::basicEncodingProbe.

2002-07-16	Tinny Ng	[Bug 10651] CMStateSet.hpp includes both memory.h and string.h.
2002-07-16	Tinny Ng	[Bug 10648] DOMDocumentImpl misaligned allocations on machines with a 64 bits 'long' type.
2002-07-15	Tinny Ng	DOM Level 3 C++ Binding.
2002-07-15	Tinny Ng	DOM L3: DOMText::getIsWhitespaceInElementContent, DOMDocument::set/getStrictErrorChecking.
2002-07-15	Robert Buck	[Bug 10834] Update version header to handle two digit revision and patch levels.
2002-07-12	Khaled Noaman	Grammar caching/preparsing.
2002-07-12	Khaled Noaman	Add getRootGrammar and modify SEnumVal.
2002-07-12	James Berry	Add some support for testing of Mac OS X builds with GCC3 compiler.
2002-07-12	James Berry	[Bug 10649] XercesDefs.hpp and AutoSense.hpp assume CodeWarrior is MacOS.
2002-07-10	Tinny Ng	Enable embedded path link option in HP.
2002-07-10	Robert Buck	[Bug 9154] Requesting Xerces Version Macro.
2002-07-08	PeiYong Zhang	[Bug 10525] runConfigure fails to recognize '-d' flag.
2002-07-05	Max Gotlib	[Bug 10250]: Implementation of new platform methods in FreeBSD.
2002-07-05	Tinny Ng	[Bug 9788] VecAttrListImpl::getValue skips prefix if SAX namespace validation is on.
2002-07-05	Robert Buck	[Bug 10065] xml4com bugs found when porting to Visual Studio .NET project files.
2002-07-05	Tinny Ng	[Bug 10105] Exception in parse() despite setErrorHandler().
2002-07-05	Tinny Ng	[Bug 10119] Grammar::getGrammarType need a const modifier.
2002-07-05	Max Gotlib	[Bug 10252] Modify FreeBSD build environment for the samples.
2002-07-04	PeiYong Zhang	[Bug 10482] XMLUri crashes with empty fragment.
2002-07-04	Max Gotlib	[Bug 10253] Bugfix for the IconvFBSD transcoder.
2002-07-04	Tinny Ng	[Bug 10336] Error in Error Message (set 3, #56, English).
2002-07-04	Tinny Ng	DOM L3: Add DOMDocument::renameNode.
2002-06-27	Tinny Ng	DOM L3: Add DOMNode::isSameNode and DOMNode::isEqualNode.
2002-06-25	Tinny Ng	Add "adoptDocument" to XercesDOMParser so that document can optionally live outside the parser.
2002-06-25	Tinny Ng	[Bug 7675] IDOM memory management problem.
2002-06-25	Tinny Ng	DOM C++ Binding: add function release().
2002-06-24	James Berry	Support CodeWarrior 8. (Important Note: Since Codewarrior 8 at long last supports HFS+ long file names, these projects now directly reference the src/xercesc files instead of the previously shortened file names in the MacSrc directory. With CodeWarrior 8 and these projects it is no longer necessary to run the perl script ShortenNames.pl to generate that MacSrc directory.)
2002-06-24	Robert Buck	[Bug 10067] SEnumVal bugs found when porting to Visual Studio .NET.

2002-06-24	Robert Buck	[Bug 10180] New Visual Studio .NET Project Files.
2002-06-18	Khaled Noaman	DOM L3: Add Wrapper4DOMInputSource and Wrapper4InputSource.
2002-06-18	Khaled Noaman	DOM L3: Modify DOMCount to modify DOMBuilder.
2002-06-18	Peter A. Volchek	Bug#9950: Compilation error on MSVC5.
2002-06-17	Tinny Ng	Add feature "http://apache.org/xml/features/validation-error-as-fatal", and users should use setFeature instead of setValidationConstraintFatal in SAX2XMLReader.
2002-06-17	Tinny Ng	Add feature "http://apache.org/xml/features/continue-after-fatal-error", and users should use setFeature instead of setExitOnFirstFatalError in SAX2XMLReader.
2002-06-17	Tinny Ng	Name Xerces features as XMLUni::fgXercesXXXX instead of XMLUni::fgSAX2XercesXXXX so that they can be shared with DOM parser.
2002-06-14	PeiYong Zhang	Build 64bit production on Itaniums platform (Windows and Linux) using Intel Compiler.
2002-06-12	Tinny Ng	DOM L3: Add DOMUserDataHandler, DOMNode::set/getUserData.
2002-06-12	Tinny Ng	Fix: Thread-safety in DOMString. The reference or update to DOMString::gLiveStringHandleCount should be synchronized (locked).
2002-06-07	Tinny Ng	DOM L3: Add Entity::get/setActualEncoding, get/setEncoding, get/setVersion.
2002-06-07	Tinny Ng	DOM L3: Add Document::get/setActualEncoding, get/setEncoding, get/setVersion, get/setStandalone, get/setDocumentURI.
2002-06-03	Tinny Ng	DOM L3: Add DOMImplementationRegistry and DOMImplementationSource.
2002-05-30	Tinny Ng	Add feature http://apache.org/xml/features/nonvalidating/load-external-dtd to optionally ignore external DTD.
2002-05-29	Khaled Noaman	DOM L3: Add DOMInputSource, DOMEntityResolver, DOMImplementationLS and DOMBuilder.
2002-05-29	Gereon Steffens	[Bug 9489] Malformed HTTP GET Requests in UnixHTTPUrlInputStream.
2002-05-28	PeiYong Zhang	DOM L3: Modify DOMPrint to use DOMWriter.
2002-05-28	PeiYong Zhang	DOM L3: Add DOMWriter, DOMWriterFilter, LocalFileFormatTarget, StdOutFormatTarget, and MemBufFormatTarget.
2002-05-28	Tinny Ng	[Bug 9104] prefixes dissapearing when schema validation turned on.
2002-05-27	Tinny Ng	Add DOMDocumentRange and DOMDocumentTraversal.
2002-05-27	Khaled Noaman	Performance: Lazily store top-level components to eliminate unnecessary traversal of DOM tree when looking up for a top level component.
2002-05-27	Khaled Noaman	Performance: Use pre-built element-attribute map table.
2002-05-27	Tinny Ng	To get ready for 64 bit large file, use XMLSSize_t to represent line and column number.

2002-05-27	Tinny Ng	Define XMLSize, XMLSSize_t and their associate MAX.
2002-05-24	Khaled Noaman	Performance: Eliminate mulitple calls to addRange and sort in regx.
2002-05-23	Khaled Noaman	Performance: Use XMLBufferMgr instead of local creation of XMLBuffer(s).
2002-05-22	Khaled Noaman	DOM L3: Add AbstractDOMParser, DOMError, DOMErrorHandler, and DOMLocator.
2002-05-21	Tinny Ng	DOM Reorganization (rename IDOM and deprecate old DOM) and other documentation update.
2002-05-19	James Berry	[Bug 9237] Encoding spec in lower case (DTD/XML) not recognized.
2002-05-10	Tinny Ng	[Bug 8967] Default element behaviour is incorrect (schema only).
2002-05-08	Martin Kalen	[Bug 7701] NameIdPoolEnumerator copy constructor should call base class.
2002-05-08	PeiYong Zhang	[Bug 8899] Missing implementation of Op::Op(const Op&) causes Intel C++ Win32 link to fail.
2002-05-08	Khaled Noaman	[Bug 8301] INFINITY used as enum member.
2002-05-08	David Bertoni	[Bug 8381] XMLScanner performance fixes.
2002-05-08	PeiYong Zhang	[Bug 8898] SchemaElementDecl doesn't compile with Intel C++ for IA32.
2002-05-07	Tinny Ng	Schema Fix: re-add the ID, IDREF ... datatype validators only if they were not there.
2002-05-07	Khaled Noaman	Update SAX2 documentation to tell users it is necessary to delete the parser returned by XMLReaderFactory::createXMLReader.
2002-05-07	David Bertoni	[Bug 8852] UnixHTTPURLInputStream.cpp includes unneeded file.
2002-05-06	David Bertoni	[Bug 8492] Incorrect HP link options.
2002-05-03	Martin Kalen	[Bug 7341] Missing newline at end of util and DOM source files.
2002-05-03	Martin Kalen	[Bug 7261] Remove obsolete define in UnixWareDefs.hpp.
2002-05-03	PeiYong Zhang	[Bug 8769] UMR (uninitialized memory read) detected by memory tool.
2002-05-01	Tinny Ng	[Bug 7265] UnixWare port broken in platformTerm().
2002-04-24	Jason Stewart	[Bug 8495] URLInputSource constructor initializes fURL member incorrectly.
2002-04-22	PeiYong Zhang	Build AIX 64 bit binary.
2002-04-19	Khaled Noaman	[Bug 8236] Problem with recursive and derived elements.
2002-04-18	PeiYong Zhang	[Bug 7301] Redundant range-check in HexBin.cpp.
2002-04-17	Tinny Ng	[Bug 7583] Build warnings with MS Visual Studio .NET.
2002-04-17	Tinny Ng	[Bug 7493] The word "occured" is misspelled and it is a global error.
2002-04-17	PeiYong Zhang	[Bug 8195] Invalid path to build 'samples' target.
2002-04-16	PeiYong Zhang	[Bug 8156] Bad path name breaks build.
2002-04-16	PeiYong Zhang	[Bug 8168] Error when attempting to build NetAccessors.
2002-04-09	Khaled Noaman	[Bug 7706] XMLString::lowerCase() does not work.
2002-04-09	PeiYong Zhang	[Bug 6095] Modify .so name to have version number.

2002-04-08	Tinny Ng	ICU 2.0.2 Update.
2002-04-04	Khaled Noaman	Change min/maxOccurs from unsigned int to int.
2002-04-03	Khaled Noaman	[Bug 7565] Attributes in different namespaces produce a Fatal Exception.
2002-04-03	Tinny Ng	check null string first in isWSCollapsed and fix [Bug 6902] Typo in XMLString.cpp.
2002-04-02	Martin Kalen	[Bug 7555] Enable AIX build with newer xlC versions.
2002-04-02	Khaled Noaman	Modiy QName comparison (operator==).
2002-04-01	Tinny Ng	According to DOM spec, setNodeValue by default is no-op.
2002-04-01	Tinny Ng	Do not issue DOM_DOMException::INUSE_ATTRIBUTE_ERR if the owner is the same.
2002-04-01	Tinny Ng	DOMString problem with Asian codepages.
2002-04-01	PeiYong Zhang	[Bug 7551] Exceptions are caught by value, rather than by reference.
2002-04-01	Tinny Ng	[Bug 7585] xml4com.dsp - Cannot open source file.
2002-04-01	Khaled Noaman	[Bug 7297] Validation of schema included in document fails with improper error.
2002-04-01	Khaled Noaman	Move Element Consistency checking (ref to global declarations) to SchemaValidator.
2002-03-27	Tinny Ng	[Bug 1173] DOMParser entity resolution property is messed about.
2002-03-27	Tinny Ng	Fix: not all the children of EntityReference Node is set to readOnly.
2002-03-27	Tinny Ng	Should call setReadOnly instead of isReadOnly to populate the flag.
2002-03-27	Tinny Ng	[Bug 3010] DocumentImpl::importNode - Missed Readonly Flag Restore.
2002-03-27	Tinny Ng	Correct count element routine in IDOMCount.
2002-03-26	Khaled Noaman	[Bug 7471] Failed to validate correctly when schema has xsd:extension> and the base has an attribute.
2002-03-25	Khaled Noaman	Move particle derivation checking from TraverseSchema to SchemaValidator.
2002-03-22	Khaled Noaman	[Bug 7358] About TraverseSchema::traverseSimpleTypeDecl member function.
2002-03-21	Khaled Noaman	Add support for reporting line/column numbers of schema errors.
2002-03-19	PeiYong Zhang	[Bug 7164] DOMParser with a DTD leak!
2002-03-19	Khaled Noaman	Fix for declarations referenced from a different NS in the case of a circular import.
2002-03-19	PeiYong Zhang	[Bug 7243] Base64 encoding is not working.
2002-03-19	Khaled Noaman	[Bug 7074] Unwarranted error regarding "no circular definitions allowed".
2002-03-18	Khaled Noaman	[Regx Fix] Change constant names to eliminate possible conflict with user defined ones.
2002-03-18	Tinny Ng	[Bug 7162] IconvFreeBSDTransService.cpp needs an #include statement fixed to use xercesc.

2002-03-15	Tinny Ng	[Bug 6888] NodeIterator. Retrofit this typo that was applied to Xerces-J, although this function "matchNodeOrParent" is not used in Xerces-C++.
2002-03-15	Tinny Ng	DOMString Thread safe Fix: should lock the entire deleter function where freeListPtr and blockListPtr are modified.
2002-03-15	Tinny Ng	Issue DOMException::INDEX_SIZE_ERR if count is greater than length, equal to length is ok.
2002-03-14	Tinny Ng	IDOM Fix: Issue IDOM_DOMException::INDEX_SIZE_ERR if count or offset is negative.
2002-03-14	Tinny Ng	IDOM Fix: Initialize fPublic/fSystemId to zero in IDNotationImpl.
2002-03-14	Tinny Ng	Certain IDOM Node should call fParent.normalize().
2002-03-14	Tinny Ng	Run methods test[NodeType] in the IDOMTest and other fixes.
2002-03-12	Mark Russell	[Bug 1687] resValue not always updated when making a transcoder.
2002/03/11	PeiYong Zhang	[Bug 7000] The URL is corrupted in UnixHTTPURLInputStream.cpp.

Release Information of Xerces-C++ 1.7.0: March 8, 2002

Date	Contributor	Description
2002-03-07	Tinny Ng	Add a keys file to store public key of committers who sign and upload packages to Apache.
2002-03-07	PeiYong Zhang	Call Terminate() to avoid memory tools reporting memory leak in Traversal test cases.
2002-03-06	PeiYong Zhang	Schema: Allow [+]? [0]* '.'? [0]* and normalize the input to positive zero string. And similarly input conforming to '-' [0]* '.'? [0]* is normalized to negative zero.
2002-03-04	Tinny Ng	[Bug 2869] AIX 4.3.3 mutex/atomic-operation changes for build.
2002-03-04	Khaled Noaman	[Bug 6834] apparently correct schema/instance not validating.
2002-03-01	Tinny Ng	NodeIDMap informational message about growing only be printed if debug is on. Besides the throw message should be encap in the XMLErrList_EN_US.Xml, not hardcoded in the code.
2002-02-28	PeiYong Zhang	[Bug 2717] Unterminated INCLUDE section causes infinite loop with setExitOnFirstFatalError(false)
2002-02-28	Tinny Ng	Fix: ReaderMgr Should check if XMLReader is created successfully.
2002-02-28	Tinny Ng	[Bug 1368] improper DOMStringHandle locking.
2002-02-28	Martin Kalen	[Bug 6445] Caldera (SCO) OpenServer Port.
2002-02-27	Tinny Ng	Fix: default attribute are not added when namespace is on and validation is off.
2002-02-27	Tinny Ng	Fix: SAX AttributeList::getName should attach prefix if present

2002-02-26	Tinny Ng	[Bug 6672] SAXValidator results in an access violation when validating against schema with empty element that has default value.
2002-02-26	Khaled Noaman	Fix: Create ZeroOrOne node for PCDATA only if needed.
2002-02-25	Tinny Ng	Schema Fix: Thread-safe the built-in datatype validator registry.
2002-02-25	Tinny Ng	Schema Fix: Ensure no invalid uri index for UPA checking.
2002-02-25	Tinny Ng	Merge IThreadTest and ThreadTest. Modify ThreadTest to do schema processing, and add ThreadTest to sanityTest.pl
2002-02-20	Don Mastrovito	Project files for BCB6.
2002-02-20	Tinny Ng	[Bug 2845] HP-UX 10.20 with CC A.10.40 needs +Z instead of +z.
2002-02-20	Tinny Ng	[Bug 5977] Warnings on generating apiDocs.
2002-02-18	PeiYong Zhang	Fix: Add code for ContentSpecNode::All in formatNode.
2002-02-18	James Berry	Add support for building with new MacOSURLAccessCF NetAccessor that doesn't require Carbon but can allow Xerces to live solely within CoreServices layer.
2002-02-17	James Berry	[Bug 6092] stricmp and strnicmp not present in FreeBSD.
2002-02-17	James Berry	Update Mac OS projects to reflect "sane includes" changes.
2002-02-15	Tinny Ng	Add IDOM to API documentation.
2002-02-15	PeiYong Zhang	Base64 interface redefined for conversion in XMLByte.
2002-02-14	PeiYong Zhang	Add getEnumString to DatatypeValidator.
2002-02-14	Khaled Noaman	[Bug 6461] Unexpected recursion errors reported against schema.
2002-02-13	Khaled Noaman	Add constraint checking for the extension of an 'all' content model.
2002-02-13	Khaled Noaman	[Bug 4581] erroneous static cast in programming examples.
2002-02-13	Khaled Noaman	[Bug 6336] Output of XMLString::transcode not freed?
2002-02-13	Khaled Noaman	Update samples to use SAX2 features/properties constants from XMLUni.
2002-02-11	PeiYong Zhang	[Bug 6330] Base64::encode does not work.
2002-02-11	Tinny Ng	[Bug 2715] Build recursion suppresses make failures.
2002-02-11	Tinny Ng	[Bug 2496] libxerces-c1_5_0 fails to build correctly on Solaris.
2002-02-06	Khaled Noaman	Added a new flag '-p' to SAX2 samples to set the 'namespace-prefixes' feature.
2002-02-06	Khaled Noaman	Use IDOM for schema processing.
2002-02-05	Tinny Ng	Remove 3rd party jar style-apachexmljar, expand into physical files.
2002-02-05	Tinny Ng	Add IDOMMemTest.
2002-02-05	Tinny Ng	Modify InitTermTest to take option flag like -s, -f , and -n.
2002-02-05	Tinny Ng	[Bug 5716] Can't parse with Validation more than one file.
2002-02-05	Tinny Ng	Recognize IBM01140 (IANA encoding) as alias of intrinsic encoding IBM1140.

2002-02-04	Tinny Ng	[Bug 6114] Memory leaks on IDOM getElementsByTagName().
2002-02-04	Tinny Ng	Add DOM Level2 missing functions: NodeIterator::getRoot, TreeWalker::getRoot Element::hasAttribute, Element::hasAttributeNS and Node::hasAttribute
2002-02-04	Tinny Ng	Memory leak fix in samples / test cases.
2002-02-01	PeiYong Zhang	src and include folder reorganization for "sane_include".
2002-01-28	Khaled Noaman	Fix: some SAX calls were not passed to the LexicalHandler.
2002-01-29	Tinny Ng	Remove those jar files that are not clear in license issue.
2002-01-28	Khaled Noaman	Add a 'null' string constant in XMLUni.
2002-01-28	Khaled Noaman	Add SAX2-ext's DeclHandler support.
2002-01-28	Khaled Noaman	The namespace-prefixes feature in SAX2 should be off by default.
2002-01-24	Tinny Ng	[Bug 3111] Problem with LexicalHandler::startDTD() and LexicalHandler::endDTD().
2002-01-23	Tinny Ng	Progressive parse does not do post-validation and thus ID/IDREF are not checked.
2002-01-23	Tinny Ng	[Bug 5545] Progressive Parse trashes when encountering "<! ... "
2002-01-23	Tinny Ng	Update DOM/IDOM hasFeature method to correctly reflect current status. And add more hasFeature test to DOMMemTest.
2002-01-21	Tinny Ng	Some intrinsic encodings support (e.g. UTF-16) only work as input encoding while reading in XML data; but do not work as output encoding in XMLformatter.
2002-01-21	Tinny Ng	Document encoding alias for intrinsic encoding support.
2002-01-21	Tinny Ng	[Bug 5847] ICUMsgLoader can't be compiled with gcc 3.0.3 and ICU2. And also fix the memory leak introduced by Bug 2730 fix.
2002-01-18	Max Gotlib	Adds the capability to compile the tests under FreeBSD and either ICU or IconvFBSD transservice (the transcoding service is automatically detected during configuration stage), with or without pthreads.
2002-01-18	Tinny Ng	[Bug 5371] runConfigure extra linker options ignored in Makefiles for tests and samples.
2002-01-18	Tinny Ng	Break program.xml which takes too long to load, into program-sax.xml, program-sax2.xml program-dom.xml, program-idom.xml.
2002-01-18	Tinny Ng	Break faq-parse.xml which becomes longer and longer into faq-parse.xml and faq-build.xml to better categorize the FAQ, and update the FAQ
2002-01-18	Tinny Ng	Create symbolic link to those duplicate ICU libraries, instead of physical duplicate copies.
2002-01-15	Khaled Noaman	[Bug 5807] Parser produces unexpected errors from 'Good' document.
2002-01-14	PeiYong Zhang	XMLURi bug fix: related to Authority and wellformedAddress
2002-01-14	Max Gotlib	[Bug 5570] DOM_Range lacks the copy constructor.

2002-01-14	Max Gotlib	Support IconvFBSD in multi-threading environment with all the possible combinations of threading and transcoding options.
2002-01-10	Khaled Noaman	[Bug 5786] Unexpected Schema errors.
2002-01-03	Khaled Noaman	Fix for identity constraints - union operation.
2002-01-03	Khaled Noaman	Resolve namespace first before resolving the schema location in <import>.
2002-01-03	Khaled Noaman	[Bug 5675] Use of setExternalSchemaLocation() yields inconsistent behavior.
2002-01-02	Khaled Noaman	Fix for validity constraint check for standalone documents.
2002-01-02	Khaled Noaman	Fix for regular expression patterns that begin with ".".
2002-01-02	Khaled Noaman	Fix for error message when checking for attributes with a namespace prefix.
2002-01-02	Khaled Noaman	[Bug 5569] <extension> does not work -- ancestor elements not recognized.
2002-01-02	Tinny Ng	Schema Fix: should not store a temp value as the key in the element pool and the attribute pool.
2001-12-22	Jason Stewart	[Bug 4953] Propagate existing CFLAGS and CXXFLAGS.
2001-12-21	Jason Stewart	[Bug 5514] XMLEnumerator needs virtual destructor.
2001-12-21	Tinny Ng	[Bug 2680] Remove '-instances=static' from the compile step.
2001-12-21	Tinny Ng	[Bug 1833] LexicalHandler::startDTD not called correctly.
2001-12-21	Frank Balluffi	[Bug 5466] Memory Leak: ElementImpl.cpp's ElementImpl::ElementImpl copy constructor does not cleanup attributes before assignment.
2001-12-21	Frank Balluffi	[Bug 5464] Memory Leak: DocumentImpl::importNode does not delete old attribute if its reference count equals zero.
2001-12-21	Tinny Ng	Schema fix: leading whitespace should be preserved for CData type.
2001-12-14	Khaled Noaman	Add surrogate support to comments and processing instructions.
2001-12-14	Tinny Ng	Performance: Do not transcode twice in DOMString constructor.
2001-12-14	Tinny Ng	update BUILDINSTRUCTIONS.TXT to be in sync with build instruction in build*.xml.
2001-12-13	PeiYong Zhang	Fix: Invalid Argument to FreeLibrary (Hint: 0x0000000).
2001-12-13	Linda Swan	iSeries (AS/400) documentation update and other iSeries related fixes.
2001-12-13	Khaled Noaman	[Bug 5410] non-schema <attribute> attributes cause error.
2001-12-12	Tinny Ng	Fix typos in messages.
2001-12-12	PeiYong Zhang	Memory leak: fRedefineList.
2001-12-12	Tinny Ng	[Bug 5367] Progressive parse does not throw error when file is empty.
2001-12-12	Tinny Ng	Performance: Remove obsolete code in ElemStack.
2001-12-11	Max Gotlib	More changes to IconvFBSDTransService. Allow using "old" TransServece implementation (via '-t native' option to runConfigure) or to employ libiconv (it is a part of FreeBSD ports-collection) services.

2001-12-11	Christopher Just	[Bug 5320] 1.5.2 Build fails on IRIX. The variable "atomicOpsMutex" has been defined twice.
2001/12/10	PeiYong Zhang	Swap checking to avoid "dangling pointer" reported by BoundsChecker.
2001-12-10	PeiYong Zhang	Memory Leak: fLeafNameTypeVector.

Release Information of Xerces-C++ 1.6.0: December 6, 2001

Date	Contributor	Description
2001-12-06	Khaled Noaman	Schema: Add Identity Constraint(Key, KeyRef, Unique, Selector, Field, and Partial XPath Support). Add XPathSymbols, XPathMatcherStack, XPathMatcher, XPathException, XercesXPath, ValueStoreCache, ValueStore, IdentityConstraint, IC_Unique, IC_Selector, IC_KeyRef, IC_Key, IC_Field, FieldValueMap, FieldActivator. Support Particle Derivation Constraint Checking.
2001-12-06	PeiYong Zhang	DatatypeValidator: Support DateTimeValidator, DateTimeDatatypeValidator, DateDatatypeValidator, TimeDatatypeValidator, DayDatatypeValidator, MonthDatatypeValidator, MonthDayDatatypeValidator, YearDatatypeValidator, YearMonthDatatypeValidator, DurationDatatypeValidator. Add SchemaDataTimeException, XMLAbstractDoubleFloat, XMLDateTime.
2001-12-06	Tinny Ng	[Bug 1959] setNodeValue throws exception when spec specifies NOP.
2001-12-06	Erik Rydgren	[Bug 2174] Bug in NamedNodeMapImpl.

2001-12-06	Henry Zongaro	Performance Enhancement. Added setNPrefix and setNLocalPart methods in QName that allow code to take advantage of the fact that it knows the length of the prefix and local name, when possible.
2001-12-06	Henry Zongaro	Performance Enhancement. Added a second ContentSpecNode constructor that allows the QName to be just assigned, not copied.
2001-12-06	Henry Zongaro	Performance Enhancement. Added a second CMLeaf constructor that indicated the QName passed in was to be adopted.
2001-12-06	Henry Zongaro	Performance Enhancement. Modify the handling of the fNEL option so that it results in fgCharCharsTable being modified, instead of having all of the low-level routines check the option.
2001-12-06	Tinny Ng	Make the runConfigure and associated config*, Makefile* in folders tests, samples and src more consistent.
2001-12-05	Khaled Noaman	[Bug 1236] Incorrect NMTOKENS attribute normalization.
2001-12-05	Khaled Noaman	[Bug 2752] Surrogate support incomplete.
2001-12-05	Edward Avis	Fix runConfigure which can run into infinite loop with invalid argument
2001-12-05	Tinny Ng	Generate linker map for certain platforms
2001-12-03	Tinny Ng	[Bug 5237] PATH_MAX undefined during build without threading support.
2001-12-03	Tinny Ng	[Bug 5179] Misprint in downcasting description.
2001-12-03	Max Gotlib	Add FreeBSD native transcoder (IconvFBSD).
2001-11-30	PeiYong Zhang	Build all tests on HP-UX 11.
2001-11-29	Michael Huedepohl	Add FreeBSD Support.
2001-11-28	PeiYong Zhang	DOMMemTest: delete compiler generated temporary DOMString object "Hello Goodbye".
2001-11-28	Tinny Ng	Fix broken ParserTest.
2001-11-28	Tinny Ng	Do not increment the error count if it is a warning.
2001-11-28	Phil Brown	[Bug 4019] XMLReader::getNextChar can over read (UTF-16).
2001-11-28	Tinny Ng	[Bug 4544] DOM_NodeList::getLength incorrect when called twice for empty list.

2001-11-28	Artur Klauser	[Bug 2238]libWWW problems with broken proxys and range requests.
2001-11-28	Artur Klauser	[Bug 2237] libWWW redirect error.
2001-11-28	Matt Lovett	[Bug 4422] BinMemInputStream::readBytes is inefficient.
2001-11-28	Tinny Ng	[Bug 3683] Access Violations when performing custom schema validation.
2001-11-28	Tinny Ng	Check tohash pointer before accessing content in XMLString::hash.
2001-11-27	Tinny Ng	Fix packageBinaries.pl to correctly strip the zip file name from the target directory which has "." dot in it.
2001-11-26	Don Mastrovito	BCB4 can use wchar_t.
2001-11-23	Tinny Ng	Support ICU 2.0.
2001-11-23	Tinny Ng	Eliminate Warning from Solaris Forte C++: Warning (Anachronism): Formal argument start_routine of type extern "C".
2001-11-23	Tinny Ng	Eliminate Warning from Solaris Forte C++: Warning: String literal converted to char* in initialization.
2001-11-23	Tinny Ng	Eliminate Warning from AIX xlC 3.6:1540-399.
2001-11-23	Tinny Ng	[Bug 4655] config.status be included in all future binary releases.
2001-11-23	Tinny Ng	[Bug 4873] ICU 2.0 breaks Xerces 1.5.2 build.
2001-11-22	PeiYong Zhang	Eliminate Visual C++ compiler warning C4273.
2001-11-22	PeiYong Zhang	Schema: Allow "0.0" to be a valid lexcial representation of ZERO.
2001-11-21	Peter A. Volchek and PeiYong Zhang	Add sample SEnumVal.
2001-11-21	Tinny Ng	New method InputSource::get/setIssueFatalErrorIfNotFound to tell the parser whether to issue fatal error or not if cannot find it (the InputSource). This is required for schema processing as it shouldn't be a fatal error if the schema is not found.
2001-11-20	Tinny Ng	Allow schemaLocation and noNamespaceSchemaLocation to be specified outside the instance document. New methods setExternalSchemaLocation and setExternalNoNamespaceSchemaLocation are added (for SAX2, two new properties are added).
2001-11-19	PeiYong Zhang	XMLFloat and XMLDouble boundary Values updated.

2001-11-16	Tinny Ng	Add test case InitTermTest to test XMLPlatformUtils:Initialize/Terminate() pair.
2001-11-16	Khaled Noaman	Design change: GeneralAttributeCheck is not longer a singleton class.
2001-11-15	Khaled Noaman	Re-organize constant values in XMLAttDef.
2001-11-13	Tinny Ng	Move root element check from XMLValidator to XMLScanner and deprecate XMLValidator::checkRootElement().
2001-11-13	Tinny Ng	Update documentation for SAX2XMLReader, DefaultHandler and DOMParser.
2001-11-09	Tinny Ng	Regular Expression: Update the Block Names and Block Range to comply to the latest standard.
2001-11-09	Carolyn Weiss	DOMIDTest/MemParse fix: Pulled the hardcoded encoding out of the document itself and made it a #define to make it easier to support other encodings.
2001-11-09	Carolyn Weiss	DOMMemTest fix: Changed some literal values to their equivalent hex values so they work correctly on both ASCII and EBCDIC systems.
2001-11-09	Linda Swan	Bug Fix: maxChars in XMLString::copyNString is more related to the target than the src.
2001-11-07	Tinny Ng	Performance: Create QName in ContentSpecNode only if it is a leaf/Any/PCDataNode.
2001-11-07	Tinny Ng	Performance: move getRawName() to outer loop in DFAContentModel so that it is called only once per outer loop.
2001-11-06	Khaled Noaman	[Bug 4644] Memory leak in schema traverser.
2001-11-02	Jason Stewart	[Bug 4133] --prefix not used properly in configure.
2001-11-01	Jason Stewart	[Bug 2730] Can't build xerces-c-1.5.1 with ICUMsgLoader.
2001-11-01	Jason Stewart	[Bug 4578] No documentation for XMLTranscoder.
2001-11-01	Tinny Ng	IDOM: Leak: should allocate the fNodeListPool with the overloaded new.

2001/10/29	Tinny Ng	Update samples doc to reflect the latest changes. Also update runConfigure usage in build doc to reflect the latest changes.
2001-10-26	PeiYong Zhang	Thread safe XMLFloat and XMLDouble.
2001-10-26	Tinny Ng	Update SAX standard web link.

Release Information of Xerces-C++ 1.5.2: October 26, 2001

Date	Contributor	Description
2001-10-26	Khaled Noaman	Schema: Support group, attributeGroup, all, any, anyAttribute, annotation, notation, redefine, circular import. Add AnySimpleTypeDatatypeValidator. Add XercesGroupInfo. More complex type constraint checking.
2001-10-26	PeiYong Zhang	DatatypeValidator: Support DoubleDatatypeValidator, FloatDatatypeValidator, AnyURIDatatypeValidator, AbstractStringValidator, AbstractNumericValidator, AbstractNumericFacetValidator, NCNameDatatypeValidator, NameDatatypeValidator. Add XMLDouble, XMLFloat, XMLInteger, XMLNumber, XMLUri.
2001-10-26	Tinny Ng	Schema: Support xsi:type, Unique Particle Attribution Constraint Checking, anyAttribute in Scanner and Validator. Add XercesElementWildCard, AllContentModel, XMLInternalErrorHandler.
2001-10-25	PeiYong Zhang	XMLDeleterFor related functions and data are removed. Replace with XMLRegisterCleanup.

2001-10-25	Henry Zongaro	[Bug 2924] runConfigure script to accept multiple linker options.
2001-10-25	John Warrier	[Bug 2924] runConfigure script to accept multiple compiler options.
2001-10-25	Mark Weaver	[Bug 4213] BinHTTPURLInputStream initialization not thread safe.
2001-10-25	John Clayton	[Bug 4121] BinHTTPUrlInputStream needs to read entire HTTP header.
2001-10-25	Tinny Ng	[Bug 4318] Single threaded build fails due to obsolete #define.
2001-10-25	Tinny Ng	[Bug 2860] gAtomicMutex should be used when APP_NO_THREADS is not defined in both Tru64 and OS400.
2001-10-25	Tinny Ng	Comment outside root element should also be reported.
2001-10-24	PeiYong Zhang	[Bug 4342] Validator mutex is not deleted.
2001-10-24	PeiYong Zhang	[Bug 3975] XMLPlatformUtils::Initialize() leaks memory after thousands of calls.
2001-10-24	Kevin Philips	[Bug 3813] BinHTTPURLInputStream has weak HTTP request capabilities.
2001-10-24	Peter A. Volchek	[Bug 2305] Include stdlib.h to BinHTTPURLInputStream.cpp.
2001-10-24	Sean Bright	[Bug 2456] loadXML gives an exception.
2001-10-24	Curt Arnold	Fixed xml4com.idl which attempts to set the version of the type library to 1.5.2 when only major.minor format is allowed.
2001-10-23	Mark Weaver	[Bug 4060] XMLPlatformUtils leaks a mutex on Solaris, Linux and others.
2001-10-23	Mark Weaver	[Bug 880] XMLPlatformUtils::Terminate cannot be called more than once.
2001-10-22	Tinny Ng	[Bug 3660] Off-by-one error in DOMString.cpp.
2001-10-22	Tinny Ng	Check that memory has been acquired successfully after memory acquisition requests in DOMString.
2001-10-22	Tinny Ng	[Bug 3361] "String pool id was not legal" error in Attributes::getURI().
2001-10-22	Linda Swan	castToNodeImpl is inconsistent with other cast routines in IDCasts.
2001-10-19	James Berry	Add new file name shortening hints; chmod +x.
2001-10-19	James Berry	Cleanup handling of transcoder failure to transcode a character; implement canTranscodeTo; thanks to Geoff Coffey.
2001-10-19	James Berry	Correctly swap / and : in classic environment MacOS pathnames; thanks to Geoff Coffey.
2001-10-19	James Berry	Update MacOS projects for CodeWarrior 7 and ProjectBuilder 1.1, new files.
2001-10-19	Tinny Ng	[Bug 3909] return non-zero an exit code when error was encountered.
2001-10-19	Tinny Ng	Modify PParse not to hardcode the number of expected elements as this may vary.

2001-10-19	David McCreedy	Fixed the binary search in XML256TableTranscoder.cpp which fails for the last item in whichever table it is searching.
2001-10-19	David McCreedy	Added U+0110 to XMLEBCDICTranscoder.cpp's "Unicode to IBM037" translation table.
2001-10-19	David McCreedy	Modified DOMPrint and IDOMPrint not to use "endl" method which puts out a newline in the local code page to generate output.
2001-10-18	Jerry Carter	[Bug 3666] Win32MsgLoader unable to retrieve error text if DLL is renamed.
2001-10-18	Tinny Ng	Use opt2 on AIX platform.
2001-10-18	Tinny Ng	[Bug 1699] Redirect "delete this" to a temp ptr to bypass AIX xlC v5 optimization memory leak problem.
2001-10-18	Tinny Ng	[Bug 4015] IDDOMImplementation::createDocumentType hopelessly broken.
2001-10-16	Khaled Noaman	[Bug 3750] GeneralAttributeCheck threading bug.
2001-10-15	Khaled Noaman	[Bug 4177] setupRange uses non-portable code.
2001-10-13	Jason Stewart	[Bug 2409] undocumented XMLException in LocalFileInputSource::new().
2001-10-13	Jason Stewart	[Bug 4133] --prefix not used properly in configure.
2001-10-10	Jason Stewart	XMLURL::parse now throws an exception if it sees a an http URL without two forward slashes ('//') following the protocol.
2001-10-10	Petr Gotthard	Add "Base64::encode" for encoding binary data.
2001-10-09	Tinny Ng	[Bug 1685] memory leak after parsing document with validation error. And other miscellaneous memory leak.
2001-10-05	PeiYong Zhang	[Bug 3831] -1 returned from getIndex() needs to be checked.
2001-10-03	Tinny Ng	[Bug 3867] IDOM_Element::getElementsByTagName() threading problem.
2001-10-02	Tinny Ng	Memory leak in IDOM, need to delete the fDocument created.
2001-09-13	Artur Klauser	Patch: Xerces 1.5 w/ libWWW for Tru64.
2001-09-13	Artur Klauser	Patch: Xerces 1.5 samples with g++ compiler.
2001-09-12	PeiYong Zhang	[Bug 3565] Stream leaked in ReaderMgr.
2001-09-12	Tinny Ng	[Bug 3155] SAX2 does not offer progressive parse.
2001-09-11	Tinny Ng	[Bug 3523] SchemaElementDecl.cpp(242) : error C2202 : not all control paths return a value.
2001-09-10	Tinny Ng	Performance: Store the fGrammarType instead of calling getGrammarType all the time for faster performance.
2001-09-04	Christopher Just	Support IRIX's sproc().
2001-09-04	Kevin Philips	[Bug 3170] URLs with ? type fragments in them don't work.
2001-08-29	Henry Zongaro	Allowing -p as argument to -z or -l in runConfigure.
2001-08-29	Tinny Ng	Performance: Use XMLBufBid instead of XMLBuffer directly for better performance.
2001-08-29	Tinny Ng	Performance: No need to new the child QName in ElemStack addChild. Remove it for performance gain.
2001-08-22	Don Mastrovito	Project files for BCB5.

2001-08-21	PeiYong Zhang	[Bug 2816]Numerous datatype headers cause CC error 1144.
2001-08-21	PeiYong Zhang	[Bug 3017] MSVC5.0: C2202: 'compareSpecial' : not all control paths return a value.
2001-08-17	Nick Chiang	Fix to memory leak in buildDFA().
2001-08-16	PeiYong Zhang	Performance: stateTable created to optimize the identification of new state created.
2001-08-10	PeiYong Zhang	Add isHex(), isAlphaNum(), isAllWhiteSpace() and patternMatch() in XMLString.
2001-08-09	Tinny Ng	[Bug 2947]IDOM segfault calling getElementsByTagName() using a DOM_Document().
2001-08-09	Tinny Ng	Port test case DOMTest to IDOMTest.
2001-08-07	Tinny Ng	[Bug 2676] IDOM: pure virtual called in IDDeepNodeListImpl::item().
2001-08-07	Kari Whitcomb	IDOM: Unaligned Access warnings in IDOM samples.
2001-08-02	Tinny Ng	[Bug 1329] SAX2XMLReaderImpl leaks XMLBuffers.
2001-08-02	Tinny Ng	Allow DOMCount/SAXCount/IDOMCount/SAX2Count to take a file that has a list of xml file as input.
2001-07-31	PeiYong Zhang	Fix: memory leak in DFAContentModel::postTreeBuildInit().
2001/07/27	Tinny Ng	Fix bug in 'transcode' functions reported by Evgeniy Gabrilovich.
2001-07-27	Tinny Ng	put getScanner() back as they were there before, not to break existing apps.
2001-07-26	Tinny Ng	[Bug 2751] Several NameChar characters missing from internal tables.
2001-07-26	Khaled Noaman	[Bug 2815] util/regx/RegxParser.cpp compile fails on HP-UX 10.20 with CC A.10.40.
2001-07-24	PeiYong Zhang	[Bug 2707] DFAContentModel memory leaks.
2001-07-19	Tinny Ng	Add IDOMCount, IDOMPrint, SAX2Count, and SAX2Print to samples.dsw.
2001-07-19	Tinny Ng	Add more tests in sanityTest.pl.

Release Information of Xerces-C++ 1.5.1: July 18, 2001

Date	Contributor	Description
2001-07-17	Khaled Noaman	[Bug 2643] - derivation by extension of complex types does not permit addition of ONLY element content.
2001-07-16	Tinny Ng	[Bug 2410] DOMParser::parse() throws undocumented exceptions.
2001-07-16	Tinny Ng	[Bug 2512] typing mistake in code example of chapter "Constructing an XML Reader".
2001-07-16	Tinny Ng	APIDocs fix: default for schema processing in DOMParser, IDOMParser, and SAXParser should be false.

2001-07-15	James Berry	Add new files to UnionTypeValidator and ListDataTypeValidator to MacOS Project files.
2001-07-09	Khaled Noaman	Add constraint checking for simple types.
2001-07-11	PeiYong Zhang	Fix to normalizeWhiteSpace: synchronize fDatatypeBuffer with toFill.
2001-07-05	PeiYong Zhang	Add ListDatatypeValidator and UnionDatatypeValidator.
2001-07-10	Tinny Ng	Give proper error message when scanning external id.
2001-07-10	Tinny Ng	The first char of PI Target Name should be checked.
2001-07-09	Khaled Noaman	Add <any> declaration.
2001-07-09	Khaled Noaman	Fixes for import/include declarations.
2001-07-09	Tinny Ng	Partial Markup in Parameter Entity is validity constraint and thus should be just error, not fatal error.
2001-07-08	James Berry	Add new samples projects: IDOMPPrint and SAX2Print for ProjectBuilder
2001-07-08	James Berry	Update ProjectBuilder Xerces project for latest file additions.
2001-07-08	James Berry	[Bug 2486] Files missing from XercesLib.mcp.
2001-07-08	James Berry	Add new samples for CodeWarrior build: IDOMPrint and SAX2Print.
2001-07-08	James Berry	New file for use in building Carbon samples.
2001-07-08	James Berry	Simplify file existence checks.
2001-07-08	James Berry	[Bug 2495] Missing (in xerces-c-src1_5_0/obj/Makefile.in.
2001-07-08	James Berry	Fix clean and distclean targets; broken because rm fails if passed no files.
2001-07-06	Tinny Ng	[Bug 2472] Linker options ignored on IRIX.
2001-07-06	Martin Kalen	Automatic build of single-threaded library.
2001-07-05	Tinny Ng	Encoding String must present for external entity text decl.
2001-07-05	Tinny Ng	Standalone checking is validity constraint and thus should be just error, not fatal error.
2001-07-05	PeiYong Zhang	Add NotationDatatypeValidator, QNameDatatypeValidator and ENTITYDatatypeValidator.
2001-07-04	PeiYong Zhang	Add IDREFDatatypeValidator and IDDatatypeValidator.

2001-07-04	PeiYong Zhang	XMLString:isValidName(): to validate Name (XML [4][5]).
2001-07-03	Tinny Ng	Some compilers (e.g. the HP compiler) has mistaken the parameter 'std', which is short for standalone as the special prefix used by the standard libraries.
2001-07-03	Miroslaw Dobrzanski-Neumann	Supporting dce threading on AIX and Solaris.
2001-06-27	David Bertoni	[Bug 2365] Huge performance problem with the parser in XMLScanner::sendCharData().
2001-06-27	David Bertoni	[Bug 2363] XMLScanner::sendCharData() can send the wrong length to the handler.
2001-06-27	Khaled Noaman	[Bug 2353] Validating Parser parses after validation failed.
2001-06-27	Murray Cumming	[Bug 1147] Headers install in wrong directory.
2001-06-26	Tinny Ng	[Bug 2119] DOMString::print() should use DOMString::transcode() for transcoding.
2001-06-25	Stephen Dulin	OS390 updates.
2001-06-25	Linda Swan	AS400 updates.
2001-06-25	PeiYong Zhang	[Bug 1393] Converting from Unicode to iso8859.
2001-06-25	Matt Lovett	[Bug 965] scanDocTypeDecl messes up the source offsets.
2001-06-25	Khaled Noaman	Add constraint checking on elements in complex types.
2001-06-22	James Berry	[Bug 2277] Bad argument to ConvertFromUnicodeToText.
2001-06-22	PeiYong Zhang	[Bug 2263] 'SIZE' : redefinition (BooleanDatatypeValidator.cpp).
2001-06-22	Khaled Noaman	[Bug 2258] Bug in Iconv and Iconv390.
2001-06-22	Tinny Ng	[Bug 2225] assignment vs. comparison in if clause.
2001-06-22	Tinny Ng	[Bug 2257] 1.5 thinks a ?xml-stylesheet ...> tag is a <?xml ...> tag.
2001-06-21	Khaled Noaman	[Bug 1946] Standalone validity check only for external decl.
2001-06-21	Tinny Ng	[Bug 2262] Duplicated header guard.
2001-06-20	PeiYong Zhang	Proper Debug Guard: Reported by Dean.
2001-06-19	Tinny Ng	Namespace should be off by default in XMLScanner.

2001/06/19	Tinny Ng	Add installAdvDocHandler to SAX2XMLReader as the code is there already.
2001-06-19	Khaled Noaman	Handle maxChars > length(toTranscode).
2001-06-18	Erik Rydgren	Memory leak fix: to addlevel().
2001-06-18	Khaled Noaman and PeiYong Zhang	Add support for 'fixed' facet.
2001-06-15	Khaled Noaman	Added constraint checking for ref on elements.
2001-06-15	Tinny Ng	ICU 1.8.1 update.

Release Information of Xerces-C++ 1.5.0: June 15, 2001

Date	Contributor	Description
2001-06-15	Tinny Ng	Schema: Add Schema support in XMLParsers (DOM/SAX/SAX2), XMLScanner. Create SchemaValidator. Add Grammar Model. Support xsi:nil. Support xsi:schemaLocation and xsi:noNamespaceSchemaLocation. Update samples to enable schema.
2001-06-15	Tinny Ng	Break DTDValidator into DTDGrammar, DTDScanner, and DTDValidator.
2001-06-15	Tinny Ng	IDOM: Complete the Range, TreeWalker, NodeIterator, and other memory fixes. Support IDOM on UNIX platform. Add samples IDOMPrint, and IDOMCount. Add test cases IRangeTest and ITraversal.
2001-06-15	Khaled Noaman	Schema: Add Regular Expression. Add Schema Messages. Add Schema Simple Type Support. Add Schema Complex Type Support (Except Group). Add Schema Attribute Declarations support. Add Schema Element Declarations support. Support Simple Content and Complex Content. Support Element and attribute reuse using "ref". Support Schema Choice and Sequence. Support Schema Import and Include.
2001-06-15	Khaled Noaman	DatatypeValidator: Add DatatypeValidator and DatatypeValidatorFactory.

2001-06-15	PeiYong Zhang	Schema: Add Schema support in Content Model. Add Schema Exception Handling. Add Schema XUtil. Add QName Support. Support SubstitutionGroup.
2001-06-15	PeiYong Zhang	DatatypeValidator: Support Base64DatatypeValidator, BooleanDatatypeValidator, DecimalDatatypeValidator, HexBinDatatypeValidator, StringDatatypeValidator, InvalidDatatypeFacetException, InvalidDatatypeValueException.
2001-06-13	Erik Rydgren	[Bug 812] Memory leak with multiple !ATTLIST on single !ELEMENT.
2001-06-08	Tinny Ng	[Bug 2043] XMLFormatter unallocates arrays incorrectly.
2001-06-08	PeiYong Zhang	Documentation and project files update for Xerces 1.5.
2001-06-08	Khaled Noaman	IDOM Documentation.
2001-06-07	Khaled Noaman	Fix no error message for faulted-in attributes if reuse grammar for 3+ times.
2001-06-06	Peter A. Volchek	/Platforms/Win32/Win32PlatformUtils.cpp Include stdlib.h.
2001-06-06	James Berry	Update Mac OS ProjectBuilder projects.
2001-06-06	James Berry	Fix invalid file references in project.
2001-06-06	James Berry	/src/util XMLString.cpp Clean up compiler warning.
2001-06-06	James Berry	/src/util/regx RegxParser.cpp Fix two improper NULL tests.
2001-06-05	James Berry	Add support for Mac OS X command line configuration and build.
2001-06-5	Peter A. Volchek	Add 'const' to getGrammar.
2001-06-04	PeiYong Zhang	The start tag "<?xml" could be followed by (#x20 | #x9 | #xD | #xA)+.
2001-06-04	James Berry	Add support for tracking error count during parse; enables simple parse without requiring error handler.
2001-06-01	Tinny Ng	/scripts/packageSources.pl Keep the BCB4 project files in the source package.
2001-05-22	James Berry	Check for existence of MacOS Unicode Converter routines prior to instantiating our transcoder object; Xerces will thus panic, rather than crash, if they don't exist. Add support to check for existence of MacOS Unicode Converter to avoid calling through NULL pointer.
2001-05-16	Henry Zongaro	IDOM: Add DeepNodeList support.
2001-05-16	Henry Zongaro	IDOM: Add namespace support.

2001-05-10	Christian Schuhegger	[Bug 1158] built-in buffer limit could be smaller than system limit, use PATH_MAX instead.
2001-05-10	Arnaud LeHors	[Bug 1605] AttrNSImpl.cpp: fixed typo in constructor.
2001-05-09	Curt Arnold	[Bug 1500] The public id was set twice and the system id was not set on Notations.
2001-05-04	Tinny Ng	DOMPrint: Check error before continuing.
2001-05-03	Tinny Ng	ICU 1.8 update.
2001-05-03	Khaled Noaman	Added new option to the parsers so that the NEL (0x85) char can be treated as a newline character.
2001-04-23	Erik Rydgren	DTDScanner: Reuse grammar should allow users to use any stored element decl as root.
2001-04-19	William L Hopper	Win32PlatformUtils: InterlockedCompareExchange on different Windows.
2001-04-19	William L Hopper	BCB project changes.
2001-04-16	James Berry	MacOSUnicodeConverter: Fix include path, Updates to reflect changes for Mac OS X final and Update MacOS projects for Mac OS X final ProjectBuilder.
2001-04-11	Arnaud LeHors	[Bug 1303] AttrImpl: allow value to be set to null.
2001-04-11	Tinny Ng	DOMParser: Attribute default values not printed in document type internal subset interface.
2001-04-10	Tinny Ng	createdocs.bat: fix PDF generation.
2001-04-04	Alberto Massari	DTDElementDecl: Error checking for null content spec.
2001-04-02	Andy Heninger	IDOM: imported.
2001-04-02	Andy Heninger	IThreadTest: imported.
2001-03-30	Tinny Ng	[Bug 1150] Problems with Namespaces and validating parsing.
2001-03-27	Roman Sulzhyk	[Bug 1069] Explicit Makefile dependency for 'lib' build.
2001-03-26	PeiYong Zhang	When Standalone="yes", it is NOT supposed to accept element which is defined in external DTD with #FIXED attribute.
2001-03-26	Andy Heninger	Update packageBinaries.pl for ICU 1.8. ICU debug .lib file names and locations changed.
2001-03-23	Jeff Harrell	[Bug 1018] AutoSense looks for "IRIX" when it should look for "sgi" or "__sgi".
2001-03-22	Roman Sulzhyk	[Bug 1069] The Makefiles fail to locate .cpp - > .o dependency and rebuild .o all the time.
2001-03-22	John Rope	[Bug 1021] Accessing an XML file using the file "protocol" and a UNC path fails to open the file.
2001-03-09	Tinny Ng	[Bug 733] Seg fault when trying to parse empty filename.
2001-03-06	Tinny Ng	[Bug 677] Infinite loop caused by malformed XML. Happen when namespace is on.
2001-03-02	Martin Kalen	Enabling libWWW NetAccessor support under UNIX. Tested with latest tarball of libWWW (w3c-libwww-5.3.2) under RedHat Linux 6.1.
2001-02-27	Tinny Ng	[Bug 676] Linux for S/390 build requires -fPIC.
2001-02-22	Tinny Ng	[Bug 678] StdInParse doesn't output filename or duration.

2001-02-21	Matt Lovett	ICUTranscoder::transcodeFrom() expects ICU function ucnv_toUnicode to return an extra element in fSrcOffsets to allow us to figure out the last char size, which in fact it is not. The fix is to compute the last char size ourselves using the total bytes used.
2001/02/16	Andy Heninger	Change limit test to reduce spurious pointer assignment warnings from BoundsChecker.
2001-02-14	Bob Kline	Better FAQ for the checksum error.
2001-02-14	Mark Everline	Core dump when UTF-16 encoding contradicts actual encoding.
2001-02-13	Hiram Clawson	Update samples/tests files for on UnixWare 7.1.1 with gcc 2.95. Add UNIXWARE platform defines to Makefile.incl, add recognition of sysv5uw7 to configure.in, and add unixware as recognized platform to runConfigure.
2001-02-09	Martin Kalen	Update support for SCO UnixWare 7 (gcc). Tested under UnixWare 7.1.1 with gcc version 2.95.2 19991024 (release) with gmake 3.79.1.
2001-02-08	Martin Kalen	Enable COMPAQ Tru64 UNIX machines to build xerces-c with gcc (tested using COMPAQ gcc version2.95.2 19991024 (release) and Tru64 V5.0 1094).
2001-02-07	Bill Schindler	Rearranged statements in Initialize() so that platformInit() is called before an XMLMutex is created.
2001-02-07	Richard Ko	Storage overlay in ucnv_setFromUCallBack.
2001-02-05	Tinny Ng	[Bug 766] /src/util/Compilers/CSetDefs.hpp: define NO_NATIVE_BOOL macro only if not pre-defined/reserved.
2001-02-05	Jordan Naftolin	Add createPDF.jar and apachPDFStyle.xsl to convert documentation xml files to pdf format.

Release Information of Xerces-C++ 1.4.0: January 31, 2001

Date	Contributor	Description
2001-01-26	Walker Curtis	Undefined symbol error when building a single threaded version of the xerces lib on irix.
2001-01-25	Arnaud LeHors	Added a flag to turn off error checking in the DOM, this is primarily used while building the DOM from the parser to get better performance.
2001-01-25	Khaled Noaman	Let users add their encoding to the intrinsic mapping table.
2001-01-25	Khaled Noaman	const should be used instead of static const. And other clean up bug fixes.
2001-01-24	Arnaud LeHors	Fixed replaceChild to handle the case where a node is replaced by itself. Cleaned up insertBefore.
2001-01-24	Tinny Ng	Guard the use of '-ptr${OUTDIR}' in EnumVal/Makefile.in
2001-01-22	Curt Arnold.	Loads winsock dynamically.

2001-01-19	Curt Arnold.	COM various updates: updated the GUID's so both can coexist, better error reporting and fixed a new minor bugs.
2001-01-18	Bill Schindler	FAQ spell check, fix typos, fix grammar, readability editing, clean up formatting, re-organize so related topics appear together.
2001-01-18	Bill Schindler	Project file updated due to removal of ChildAndParentNode.cpp.
2001-01-17	Arnaud LeHors	DOM Implementation Optimization.
2001-01-17	Volker Krause	ElementImpl::getAttributeNS should check null pointer.
2001-01-17	Arnaud LeHors	Have a single counter global to the document. Removed node basis change counter.
2001-01-17	Arnaud LeHors	Removed unused field in NodeImpl that was left over.
2001-01-17	Tinny Ng	Access violations and stack overflows in insertBefore.
2001-01-15	David Bertoni	Performance Patches.
2001-01-12	Tinny Ng	Fix style-ibm.zip for documentation generation.
2001-01-12	Tinny Ng	Remove the two obsolete file: stylesheets\Copy of book2project.xsl and stylesheets\Copy of document2html.xsl in style-apachexml.jar
2001-01-12	Tinny Ng	Documentation Enhancement: explain values of Val_Scheme.
2001-01-12	Tinny Ng	Documentation Enhancement: Add list of SAX2 feature strings that are supported.
2001-01-04	Khaled Noaman	Assertion `size > 0' failure when cloning a node if the last attributes has been removed.
2000-12-28	James Berry	Omit include carbon.h in favor of specific include files.
2000-12-28	James Berry	Add or modify cvs header in various files.
2000-12-28	James Berry	Eliminate compiler warning in RangeImpl.cpp.
2000-12-28	James Berry	Replace include of Carbon.h with specific include files.
2000-12-28	James Berry	Move away from include of Carbon.h; include only needed files instead. Fix bug in parsing of upwardly relative paths under classic (thanks to Lawrence You).
2000-12-22	Tinny Ng	XMLUni::fgEmptyString which is defined as "EMPTY" is incorrectly used as an empty string; in fact XMLUni::fgZeroLenString should be used instead.
2000-12-22	Tinny Ng	Add the new header LexicalHandler.hpp to Makefile.in.
2000-12-22	Murray Cumming	removes '-instances=static' from the Linux link sections.
2000-12-22	David Bertoni	SAX2-ext's LexicalHandler support.
2000-12-14	Tinny Ng	Better instruction for using packageBinaries.pl. Use symbol XercesCInstallDir and XercesCSrcInstallDir instead of hardcoding the Xerces version number in the file.
2000-12-14	Tinny Ng	Fix API document generation warning: "Warning: end of member group without matching begin".
2000-12-14	Tinny Ng	Add RangeTest as part of the xerces-all MSVC++ workspace.
2000-12-12	Gareth Reakes	null pointer bug.
2000-12-08	Tinny Ng	Entity Reference cleanup dumping core if the last entity reference is deleted.

2000-12-06	Tinny Ng	fix the link to FAQ.
2000-12-06	Tinny Ng	further fixes to Range, and update RangeTest.cpp with more test coverage.
2000-11-30	Bill Schindler	Spell check, fix typos, fix grammar, readability editing, clean up formatting.
2000-11-30	Bill Schindler	Remove dead code (old StdOut and StdErr functions); minor clean-up.
2000-11-30	Tinny Ng	patch to fix a number of Range problems. See mail of 11/21/2000.
2000-11-30	Tinny Ng	DOM_Text::splitText(), fix off by one error in the test for index too big error.
2000-11-30	Tinny Ng	reuseValidator - fix bugs (spurious errors) that occurred on reuse due to pools already containing some items.
2000-11-08	Andrei Smirnov	Build updates for Solaris 2.8 64 bit.
2000/11/07	Tinny Ng	Bug fix for DTD entity reference problem reported by Tony Wuebben on 10/25.
2000-11-07	Tinny Ng	config.guess and config.sub updated to newer versions.
2000-11-07	Pieter Van-Dyck	Change InterlockedCompareExchange for compatibility with Borland BCB5
2000-11-07	Pieter Van-Dyck	Fix incorrect version number in gXercesMinVersion.
2000-11-01	Tinny Ng	SAX bug fix: Attribute lists were throwing exceptions rather than returning null when an attribute could not be found by name.
2000-11-01	Tinny Ng	Scanner bug fix: with progressive parsing, namespace and validation options were not being set correctly. Symptoms included failure to detect ignorable white space.
2000-10-31	Tinny Ng	DOM NodeIterator bug fix: iterators would sometimes continue beyond their starting (root) node.
2000-10-20	Andy Heninger	DOMParser bug fix - erroneous attempt to look up name space URIs while scanning default attribute values in DTD removed. Was a crashing bug when namespaces were enabled.
2000-10-20	Andy Heninger	DOM NodeFilter - define values for FilterAction enum to match those in the DOM spec.
2000-10-19	Andy Heninger	SAXCount sample, allow multiple files on command line. DOMCount sample, rename error handler class to say that it is an error handler.
2000-10-18	James Berry	MacOS project file updates. Small code optimization. Add comments to clarify and to reflect new fixed XMLCh size.
2000-10-17	Andy Heninger	Bug Fix - problems with multi-byte characters on input buffer boundaries.
2000-10-17	Andy Heninger	DOMPRintFormatTarget, bad override of writeChars fixed (missing const). XMLFormatTarget, removed version of writeChars with no length. Can not be safely used, and obscured other errors.
2000-10-16	Andy Heninger	Change XMLCh back to unsigned short on all platforms
2000-10-13	Devin Barnhart	COM: interpret BSTR as UTF-16 in documents

2000-10-13	Edward Bortner	Solaris: change detection for native support for type bool to defined(_BOOL).
2000-10-13	Nadav Aharoni	MXLString::trim() bug fix: failure to null terminate result.
2000-10-10	Bill Schindler	XMLFormatter: Fix problems with output to multi-byte encodings.
2000-10-10	Andy Heninger	From Janitor, remove the addition that is having compile problems in MSVC.
2000-10-10	James Berry	Fix a bug in returned length of transcoded string. Add a few comments.
2000-10-09	James Berry	ProjectBuilder project to build Xerces.
2000-10-09	James Berry	Numerous Changes: - Increase environmental sensitivity with hope of supporting pre OS 9 OS versions. - Enhanced path creation/interpretation to support proper unix style paths under Mac OS X instead of the volume rooted paths we previously used. Paths under Classic remain the same. - Better timer resolution. - Detect functionality via unresolved symbols rather than Gestalt where possible. - Softly back away from URLAccess...if it's not installed, we just don't support a net accessor. - Additional support for XMLCh/UniChar size differences under GCC on Mac OS X. - Fix Mac OS X support. GCC in this environment sets wchar_t to a 32 bit value which requires an additional transcoding stage (bleh...) - Improve sensitivity to environment in order to support a broader range of system versions. - Fix a few compiler sensitivities. - Carbon.h header support
2000-10-09	James Berry	Add some auto_ptr functionality to allow modification of monitored pointer value. This eases use of Janitor in some situations.
2000-10-09	James Berry	Autosense.hpp: modify sensing of Mac OS X.
2000-09-28	Andy Heninger	DOM_Document::putIdentifier() removed. There never was an implementation for this function.
2000-09-28	Curt Arnold	COM wrappers updated.
2000-09-28	Linda Swan	AS400 related changes.
2000-09-28	Andy Heninger	DOM_Document - remove the un-implemented function putIdentifier() from the header.
2000-09-28	Andy Heninger	DOMParser MemoryLeak fixed. Occurred when a document redefined the a builtin entity, e.g. <.
2000-09-28	Andy Heninger	DOMPrint sample: add deletes before exit so boundschecker runs cleanly.
2000-09-22	James Berry	Change file access permissions to fsRdPerm. Since we never write, there's no reason to request write access. Thanks to John Mostrom @ Adobe. Also nuke a few spaces and the entire defunct support for reading directly from MacOS resources.
2000-09-22	Arundhari Bhowmick	DOM Parser: internal subset entity printing update.

Release Information of Xerces-C++ 1.3.0: Sept 21, 2000

Date	Contributor	Description
2000-09-21	Torbjörn Bäckström	HPUX - Incorrect use of Array Janitor in Platform Utils removed.
2000-09-21	Arundhati Bhowmick	DOMPrint - DTD internal subset, printing of attribute value enumerations was broken.
2000/09/19	Arundhati Bhowmick	DOMPrint - output entity reference nodes as XML entity references, instead of just printing their children.
2000-09-19	Bill Schindler	OS/2 - port update
2000-09-18	Arundhati Bhowmick	DOM EntityReferences, fixed bugs with length() and hasChildNodes() methods.
2000-09-12	Arundhati Bhowmick	DOM: changed name of expandEntityReferences option to createEntityReferenceNodes. More accurately describes what it does. Fixed bugs that caused creation of Entity Reference nodes to fail.
2000-09-12	IBM	AS400 - transcoder updates.
2000-09-11	Shengkai Qu	OS390 - makefile updates
2000-09-11	Kirk Wylie	Alpha processor support update in config.sub.
2000-09-08	Kirk Wylie	Reordered member variables in ThrowEOEJanitor.
2000-09-08	Arnaud LeHors	DOM NamedNodeMap - because in many cases we may have to deal with both nodes with a namespace and nodes without any, NS methods through findNamePoint must handle both types of nodes.
2000-09-08	Kirk Wylie	Some destructors not virtual that should have been; some members of DOM_Entity virtual that should not have been.
2000-09-08	Andy Heninger	Removed incorrect detection of nested CDATA sections. Problem reported by Johannes Lipp.
2000-09-08	Andy Heninger	DOMPrint incorrectly handled DOCTYPE declarations containing both a public and system id. Problem reported by Jesse Pelton.
2000-09-08	Radovan Chytracek	MSVC: RangeTest project settings incorrect, build failed.
2000-09-07	Bob Kline	XMLReader::skippedString(), failed under certain rare circumstances.
2000-09-07	Andy Heninger	Fix SAXException assignment operator. Now non-virtual, and SAXParseException subclass invokes base class operator.
2000-09-06	William L. Hopper	Borland updates. It had fallen way behind.
2000-09-06	Andy Heninger	HPUX 11, packageBinaries build script, DCEThreads no longer default
2000-09-06	James Berry	Macintosh: Add support for new compile time options defined in prefix file. These control the selection of the msgloader, transcoder, and netaccessor. Add a tiny bit of robustness to the nasty panic method..
2000-09-06	Shengkai Qu	S390: socket related changes

2000-09-06	James Berry	Macintosh: Allow ShortenFiles to work even when destination directory already exists.
2000-09-06	Arundhati Bhowmick	HP compile options modified for ICU compatibility
2000-09-05	Michael Crawford	Macintosh: Fix atomic increment & decrement to return value after operation rather than before.
2000-09-05	Andy Heninger	Cleaned up various compiler warnings.
2000-09-05	Andy Heninger	SAX parser: added advanced callback support for XMLDecl
2000-09-01	Andy Heninger	Fix ICU transcoding service, crashing bug on Linux, Solaris
2000-08-30	Andy Heninger	Builds - clean up a number of compiler warnings.
2000-08-24	Andy Heninger	DOMPrint - fixed crash when input xml file was not found.
2000-08-23	Andy Heninger	Build Script updates and cleanups
2000-08-18	Andy Heninger	Version number bumped to 1.3 in preparation for the upcoming xerces 1.3 / xml4c 3.3 release
2000-08-17	Arnaud Lehors	DOM: Rewrote code updating the linked list on node addition and removal. I believe it is now easier to read and it uses fewer tests so it is also a little faster.
2000-08-17	Arnaud Lehors	DOM: small cleanup: renamed a set of [] boolean flag methods. yes, I know, I also wish I got them right in the first place...
2000-08-17	Sumit Chawla	PTX port updates
2000-08-16	Andy Heninger	Fixed crash when XML text content has very long lines. Bug pointed out by Simon Fell.
2000-08-14	Joe Polastre	SAX2 DefaultHandler, inconsistency in const parameters fixed.
2000-08-11	Arundhati Bhowmick	ICU Transcoding - updates to support ICU 1.6
2000-08-09	Arundhati Bhowmick	DOM Range: Add const to API where appropriate.

2000-08-09	Joe Polastre	Many conformance and stability changes: - ContentHandler::resetDocument() removed - attrs param of ContentHandler::startDocument() made const - SAXExceptions thrown now have msgs - removed duplicate function signatures that had 'const' [eg: getContentHander()] - changed getFeature and getProperty to apply to const objs - setProperty now takes a void* instead of const void* - SAX2XMLReaderImpl does not inherit from SAXParser anymore - Reuse Validator (http://apache.org/xml/features/reuse-validator) implemented - Features & Properties now read-only during parse
2000-08-09	Joe Polastre	Namespaces bug - bogus default namespace removed.
2000-08-09	Joe Polastre	SAXException enhanced, messages added.
2000-08-08	Joe Polastre	SAX2Count - new sample program for SAX2.
2000-08-07	Arundhati Bhowmick	Remove detach() method from TreeWalker.
2000-08-03	James Berry	Add Mac Codewarrior projects.
2000-08-01	Joe Polastre	SAX2 support added
2000-08-01	Gary Gale	Compaq Tru64 port added.
2000-07-31	Joe Polastre	bug fix in removeAll() to zero out all the pointers.
2000-07-31	Andy Heninger	utf-8 byte order mark recognition
2000-07-29	James Berry	Mac OS Port, general cleanups.
2000-07-28	James Berry	Addition of NetAccessor functionality for MacOS, built on URLAccess library.
2000-07-28	Arundhati Bhowmick	ICU Transcoding service: changes for move to ICU 1.6
2000-07-27	Arundhati Bhowmick	DOM Range added. (Major new feature)
2000-07-27	Murray Cumming	makefile fixes for SUNW_0.7
2000-07-25	Arundhati Bhowmick	XMLCh character constants definitions moved to XMLUniDefs.h. Removes name clashes with application defined symbols.
2000-07-25	Joe Polastre	allow nesting of PlatformUtils::Init() and Terminate()
2000-07-25	Gary Gale	ICU transcoding: fix off by one error.
2000-07-21	<check>	Change wcsupr to _wcsupr
2000-07-21	Eric Schroeder	Win32TransService - fix error in use of hashtables
2000-07-21	Joe Polastre	DOMPrint: fixed error in handling of null CDATA sections.

2000-07-20	Andy Heninger	Improved net access (parse of URLs). Still weak, though.
2000-07-20	Erik Schroeder	XMLScaner.cpp bugfix: call startDocument() at beginning of scan.
2000-07-20	Arundhati Bhowmick	DOMCount exception handling cleaned up.
2000-07-19	Todd Collins	runConfigure: modified to take "configureoptions"
2000-07-19	<check>	Add 'make install' target to src/util/Platforms/Makefile.in
2000-07-19	<check>	DOM: BugFix: DocumentType nodes can not have children.
2000-07-19	<check>	DOM: Bug in NodeIDMap constructor.
2000-07-18	Anupam Bagchi	Documentation generation tools updated.
2000-07-17	James Berry	Mac OS port brought up to date (was very old)
2000-07-17	Andy Heninger	Change windows project to link with ws2_32.lib instead of winsock32.lib
2000-07-17	Grace Yan, Joe Kesselman	DOM NodeIterator: bug fix for SHOW_ELEMENT flag incorrectly being retrieved.
2000-07-17	Joe Polastre	switched scanMisc() with endDoc() in scanNext. Pointed out by Dean Roddey.
2000-07-17	Jim Reitz	fix for uninitialized variable gotData bug in XMLScanner.cpp.
2000-07-12	Arundhati Bhowmick	DOM: fix bug in setting previous sibling pointer during insertNode
2000-07-07	Joe Polastre	Update to use of hashtables.
2000-07-07	Joe Polastre	DOM userdata: several bug fixes.
2000-07-06	Andy Heninger	Speedups in XMLScanner, XMLReader
2000-07-07	<check>	bug fixes in IXMLDOM*
2000-07-06	Joe Polastre	Performance tweaks, added more inlines.
2000-07-05	Anupam Bagchi	Documentation updates.
2000-07-05	Joe Polastre	DOM: Attribute node default value handling implemented.
2000-07-05	Joe Polastre	DOM Attr nodes - fixed setting of specified when cloning. (change may be in error)
2000-07-04	Dean Roddey	Fixed a memory leak when namespaces are enabled.
2000-06-28	Curt Arnold	COM object usage documentation update.
2000-06-28	Joe Polastre	DOM Userdata - put pointers in a hash table rather than having one pre-allocated per node. Memory footprint reduction.
2000-06-27	Joe Polastre	extended the (implementation) hash table classes.
2000-06-26	John Roper@iOra.com	Bug fix: check if initialized in Terminate() to stop access violations.
2000-06-26	<check>	Solaris build - template directory related changes.

Release Information of Xerces-C++ 1.2.0: June 22, 2000

Date	Contributor	Description

2000/06/22	<check>	OS/2 Port updated.
2000-06-22	Joe Polastre	DOM Attr nodes, specified flag not set correctly by parser. Fixed.
2000-06-20	Rahul, Joe, Arundhati	Many doc updates in preparation for release of 1.2
2000-06-19	Rahul Jain	Update Package Binaries script to build Xerces with ICU.
2000-06-19	Joe Polastre	Added help messages to PParse and StdInParse samples.
2000-06-19	Joe Polastre	Changed "XML4C" to "Xerces-C" in DOMPrint. (Missed in earlier mass name change.)
2000-06-19	Arundhati Bhowmick	Moved version.incl up one directory level.
2000-06-19	Curt Arnold	Improved Windows project file.
2000-06-16	John Smirl	Bug Fix: Document Handler was not called for PIs occurring before the document element. Bug identified by John Smirl and Rich Taylor
2000-06-16	Rahul Jain	DOMPrint, SAXPrint: remove extra space in printing PIs.
2000-06-16	Rahul Jain	Windows Debug Build: add 'D' suffix to DLL name in VCPPDefs.hpp
2000-06-16	Rahul Jain	Samples: added -v option (validate always). Needed for testing scripts.
2000-06-14	Joe Polastre	Fixed null ptr failures in DOM NamedNodeMap
2000-06-12	Andy Heninger	Fixed bug in XMLString::trim(), reported by Michele Laghi
2000-06-07	Joe Polastre	DOM: reduced memory usage for elements with no attributes.
2000-06-01	Andy Heninger	DOMString - add const to return type of const XMLCh *DOMString::rawBuffer()
2000-06-01	Arundhati Bhowmick	Fix crash with Solaris optimized build. Modified XMLURL.cpp to dodge compiler code generation error.
2000-06-01	Joe Polastre	Bug fix: DOM Attr Specified flag was incorrectly set when cloning or importing attributes.
2000-05-31	Andy Heninger	MSVC projects modified to produce separate debug and release versions of Xerces lib and dll.
2000-05-31	Rahul Jain	Bug fix: DOMPrint, SAXPrint produced garbage output on Solaris. Solaris library problem.
2000-05-31	Joe Polastre	Fixed incorrect error check for end of file in Win32 platform utils.
2000-05-31	Rahul Jain	DOMPrint enhancements. Add options for specifying character encoding of the output, better control over escaping of characters, better handling of CDATA sections. Default validation is now "auto"
2000-05-22	Dean Roddey	XMLFormatter now escapes characters, as reqd., occurring midway in strings. Reported by Hugo Duncan.
2000-05-22	Andy Heninger	Bug fix in implementation of DOM_Document::GetElementById()

2000-05-18	Anupam Bagchi	Documentation, DTD for source xml files moved into xerces-c project, sbk: prefixes removed, xml can now be validated locally.
2000-05-15	Dean	Fixed 'fatal error' when 'reusing the validator' problem reported by Rocky Raccoon (rrockey@bigfoot.com). Fix submitted by Dean Roddey (droddey@charmedquark.com).
2000-05-15	James Berry	Changed #include <memory.h> to <string.h> everywhere. <jberry@criticalpath.com>
2000-05-15	Andy H.	DOMTest: removed incorrectly failing entity tests
2000-05-12	Andy H.	Revised implementation of DOMDocument::getElementsById(), removed memory leaks, new test program for it.
2000-05-12	Dean	Bug fix - A PE ref appearing at the start of a skipped conditional section was incorrectly being processed rather than ignored. Fix from Dean Roddey.
2000-05-11	Rahul Jain	Start using the socket based netaccessor by default on most Unix platforms.
2000-05-11	Rahul Jain	Update ICUTransService to work with latest revision of ICU which provides a hard linked data DLL. i.e. icudata.dll will be loaded when xerces-c is loaded.
2000-05-05	Dean	Problem with progressive parsing. parseNext() would through an exception when the document contains entities, either or external.
2000-05-11	Sean MacRoibeaird	Add missing validity checks for stand-alone documents, character range and Well-formed parsed entities.
2000-05-10	Radovan Chytracek	Fix compilation problems on MSVC 5. Radovan.Chytracek@cern.ch>
2000-05-10	Dean	Fix XMLReader defect reported by SHOGO SAWAKI
2000-05-09	Andy H	Fix problem with Windows filenames containing '\' in Japanese and Korean encodings.
2000-05-08	Andy H	Memory Cleanup. XMLPlatformUtils::Terminate() deletes all lazily allocated memory
2000-05-05	Dean	Fixed defect in progressive parsing 'parseNext()' reported by Tim Johnston
2000-05-03	Tom Jordahl	Fixed Solaris build problems with static character constants. Tom Jordahl <tomj@allaire.com>
2000-04-28	Arnaud LeHors	Reduced memory usage for DOM Attributes.
2000-04-28	boercher@kidata.de	New runConfigure options -P and -C
2000-04-27	Andy H	Memory leaks in TransService. Joseph Chen JosephC@plumtree.com>
2000-04-27	Arnaud LeHors	DOM - storage requirements for nodes substantially reduced.
2000-04-27	Arundhati	Added DOM XMLDecl node type; provides access to XML declaration.

2000-04-20	Arundhati	Added DOM access to DTD subset (DOM Level 2 feature)
2000-04-19	Anupam Bagchi	API document generation changed to Doxygen from Doc++
2000-04-18	Arundhati	Full support for DOM_EntityReference, DOM_Entity and DOM_DocumentType introduced
2000-04-18	Dean Roddey	Don't allow spaces before PI target. Bug #42
2000-04-17	Anupam Bagchi	Follow the SMP/E procedures for the OS/390 BATCH install
2000-04-12	Dean Roddey	Auto-validate mode. Validate only when a DTD is present.
2000-04-11	Dean Roddey	If a SAX error handler is installed, then the resetErrors() event handler should call the one on the installed SAX error handler.
2000-04-10	Dean Roddey	Allow an empty DOCTYPE declaration, with just the root name.
2000-04-06	Dean Roddey	Add low level support for transcoding XML output to different character encodings.
2000-04-06	Arnaud Lehors	DOM node memory footprint reduction.
2000-04-06	Dean Roddey	Fixed hanging bug in character transcoding.
2000-04-05	Dean Roddey	Enable installation of DTDHandler on SAX parser.
2000-04-04	Anupam Bagchi	Support for PTX platform
2000-04-03		IRIX 6.5 port
2000-03-30		COM wrappers
2000-03-24	Jeff Lewis	DOM_Document::GetElementsByTagId() added.
2000-03-23	Chih Hsiang Chou	DOM: support for identifying "ignorable white space" text nodes.
2000-03-23	Rahul Jain	URL Net Accessor added.
2000-03-20	Dean Roddey	Fix null pointer exception with some bad documents.
2000-03-17	Dean Roddey	Initial support for two-way transcoding.
2000-03-17	Dean Roddey	Intrinsic transcoding table generation utility added.
2000-03-17	Anupam Bagchi	UNIX build: Now generates object files in platform-specific directories
2000-03-13	Anupam Bagchi	Fix GCC build problem: Changed XML_GNUG to XML_GCC
2000-03-13	Helmut Eiken	Fixed #54. Changed self-assignment to now use the parameter value. Reported by Helmut Eiken <H.Eiken@cli.de>
2000-03-10	Chih Hsiang Chou	Fix bug # 19, add const keyword to API. As a result, update test case.
2000-03-10	Chih Hsiang Chou	DOM: "specified" flag of attributes now set correctly.
2000-03-08	Dean Roddey	Some fixes for content models that have multiple, trailing, empty PE refs (for content model extension.)
2000-03-07	Dean Roddey	First cut for additions to Win32 xcode. Based very loosely on a prototype from Eric Ulevik.

2000-03-03	Dean Roddey	Fixed a bug in SimpleContentModel that allowed an <a/> to be taken as valid for a content model of (a,b).
2000-03-02	Dean Roddey	Added a scanReset()/parseReset() method to the scanner and parsers, to allow for reset after early exit from a progressive parse. Added calls to new Terminate() call to all of the samples. Improved documentation in SAX and DOM parsers.
2000-03-02	Dean Roddey	Change "XML4C" to "Xerces" in many places Add a cleanup method to XMLPlatformUtils. Implement the Locator scheme for SAX. Add a -n option to most of the samples, to enable namespaces Fix an error where XMLScanner::parseNext() was falling through on an exception instead of return a failure. Implement the specialized string loading for Win98, since LoadStringW() doesn't work on 98 and makes the loaded error text from the Win32 message loader come out junk fix error when two trailing entity references in a content model, like so: <!ELEMENT foo (a|b|c|d|e %one;%two;)*>

Release Information of Xerces-C++ 1.1.0: Feb 28, 2000

Date	Contributor	Description
2000/02/18	Dean Roddey	XMLCh defaults to wchar_t on platforms where wchar_t uses Unicode.
2000-02-18	Dean Roddey	Add Windows-1252 as a built in encoding
2000-02-17	Dean Roddey	Fixed an infinite loop caused while trying to trim leading white space from the raw URL during parsing.
2000-02-17	Rahul Jain	Add LibWWW based net accessor
2000-02-17	Chih Hsiang Chou	DOM: NodeIterator, TreeWalker added.
2000-02-16	Dean Roddey	Updates for EBCDIC code page issues.
2000-02-15	Chih Hsiang Chou	DOM: several namespace bugfixes
2000-02-14	Dean Roddey	Disallow EBCDIC documents without an encoding declaration
2000-02-10	Bill Schindler	Fixed defect in compare[N]IString function. Defect and fix reported by Bill Schindler from developer@bitranch.com
2000-02-10	Anupam Bagchi	Sample source code cleaned up.
2000-02-08	Dean Roddey	Fixed bug: xmlns:xxx="" should affect the mapping of the prefixes of sibling attributes

2000-02-07	Dean Roddey	Don't weave base and relative paths unless relative part is really relative.
2000-02-03	Dietrich Wolf	C++-Builder 4 support
2000-02-03	Robert Weir	DOMString enhancements
2000-01-31	Dean Roddey	Win32 mutex implementation was changed to use critical sections for speed.
2000-01-28	Dean Roddey	The API is not in place to allow client code to make sense of start/end entity ref calls from attribute values. So suppress them for now.
2000-01-28	Andy Heninger	Fix multi-threading problem in DOM.
2000-01-27	Dean Roddey	Fixed bug: If an entity ends on the last > of some markup, then the end of entity won't be sent because the end of entity is not sensed.
2000-01-24	Dean Roddey	Fixes a bogus error about]]> in char data.
2000-01-24	Dean Roddey	Exposed the APIs to get to the byte offset in the source XML buffer.
2000-01-21	Dean Roddey	Added a check for a broken pipe error on file read.
2000-01-18	Dean Roddey	Update to support new ICU 1.4 release
2000-01-18	Dean Roddey	Remove dependence on old utils standard streams
2000-01-18	Rahul Jain	Added CreateDOMDocument sample.
2000-01-13	Dean Roddey	Added a NetAccessorException for use by implementations of the NetAccessor abstraction, if they need to report errors during processing
2000-01-12	Dean Roddey	get the C++ and Java versions of error messages more into sync.
2000-01-11	Dean Roddey	Moved the input source classes from / to framework/.
2000-01-11	Dean Roddey	Changes to deal with multiply nested, relative paths, entities

Release Information of Xerces-C++ 1.0.1: December 15, 1999

- Port to Solaris.
- Improved error recovery and clarified error messages.
- Added DOMTest program.

Release Information of Xerces-C++ 1.0.0: December 7, 1999

- Released Xerces-C++ after incorporating ICU as a value-added plug-in.
- Has bug fixes, better conformance, better speed and cleaner internal architecture
- Three additional samples added: PParse, StdInParse and EnumVal
- Experimental DOM Level 2 support
- Support for namespaces
- Loadable message text enabling future translations to be easily plugged-in
- Pluggable validators
- Pluggable transcoders
- Reorganized the util directory to better manage different platforms and compilers

Release Information of Xerces-C++ BETA: November 5, 1999

- Created initial code base derived from IBM's XML4C Version 2.0
- Modified documentation to reflect new name (Xerces-C)

4
Future Releases Plan

Xerces-C++ Future Releases Plan

This document highlights the release plan for Xerces-C++.

Current Status

Xerces-C++ 2.3.0 - released on May 23, 2003.

Next Target Release

Xerces-C++ 2.4.0 - plan to be released by the Fall of 2003.

Xerces-C++ Features list

The following table lists the TODO items for Xerces-C++. It does not include fixing bugs that are opened in Bugzilla; unless such Bugzilla bug involves a major development effort and requires an architectural redesign, or is an enhancement suggestion.

Only those features that have volunteer identified will have a 'Target Date'. Those features that have 'Complete' status will be included in the 'Next Target Release'.

Features	Development Volunteer	Target Date	Status
More DOM Level 3 Core Support			
Schema 1.1			
PSVI			
SAX-like API			
Grammar Pool			
Persistent Grammar			
Memory Pool			

5 Installation

Windows NT/2000

Source distribution

The Xerces-C++ source is available in the source distribution.

Install the Xerces-C++ source distribution by using `unzip` on the xerces-c-src2_5_0.zip archive in the Windows environment. You can use WinZip, or any other UnZip utility.

```
    unzip xerces-c-src2_5_0.zip
```

This creates a 'xerces-c-src2_5_0' sub-directory containing the Xerces-C++ source distribution.

If you need to build the Xerces-C++ source after installation, please follow the Build Instructions.

Binary distribution

Install the Xerces-C++ binary distribution by using `unzip` on the xerces-c2_5_0-win32.zip archive in the Windows environment. You can use WinZip, or any other UnZip utility.

```
    unzip xerces-c2_5_0-win32.zip
```

This creates a 'xerces-c2_5_0-win32' sub-directory containing the Xerces-C++ binary distribution.

You need to add the 'xerces-c2_5_0-win32\bin' directory to your path:

To do this under Windows NT, go to the start menu, click the settings menu and select control panel. When the control panel opens, double click on System and select the 'Environment' tab. Locate the PATH variable under system variables and add <full_path_to_xerces-c2_5_0 >\bin to the PATH variable. To do this under Windows 2000 add this line to your AUTOEXEC.BAT file:

```
    SET PATH=<full_path_to_xerces-c2_5_0>\bin;%PATH%
```

or run the `SET PATH` command in your shell window.

Besides, if the parser is built with icu message loader (like IBM XML4C binaries), or message catalog loader, then you need to create a new environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

```
    SET XERCESC_NLS_HOME=<full_path_to_xerces-c2_5_0>\msg
```

The binary distribution has the built parser library and some samples executables. Please refer to the Samples for how to run the samples.

UNIX

Source distribution

The Xerces-C++ source is available in the source distribution.

Install the Xerces-C++ source distribution xerces-c-src2_5_0.tar.gz by extracting the files from the compressed archive.

```
gunzip xerces-c-src2_5_0.tar.gz
tar xerces-c-src2_5_0.tar
```

This creates a 'xerces-c-src2_5_0' sub-directory containing the Xerces-C++ source distribution.

> ***Note:*** *On Solaris, please use `gtar` instead of tar. See FAQ for more information.*

If you need to build the Xerces-C++ source after installation, please follow the Build Instructions.

Binary distribution

Install the binary distribution xerces-c2_5_0-xxx.tar.gz by extracting the files from the compressed archive; where 'xxx' is the corresponding UNIX platform. For example:

```
gunzip xerces-c2_5_0-linux.tar.gz
tar -xvf xerces-c2_5_0-linux.tar
```

This will create an 'xerces-c2_5_0-linux' sub-directory containing the Xerces-C++ binary distribution.

> ***Note:*** *On Solaris, please use `gtar` instead of tar. See FAQ for more information.*

You will need to add the xerces-c2_5_0-linux/bin directory to your PATH environment variable:

For Bourne Shell, K Shell or Bash, type:

```
export PATH="$PATH:$HOME/xerces-c2_5_0-linux/bin"
```

For C Shell, type:

```
setenv PATH "$PATH:$HOME/xerces-c2_5_0-linux/bin"
```

If you wish to make this setting permanent, you need to change your profile by changing your setup files which can be either .profile or .kshrc.

In addition, you will also need to set the library search path. (LIBPATH on AIX, LD_LIBRARY_PATH on Solaris and Linux, SHLIB_PATH on HP-UX, and DYLD_LIBRARY_PATH on Mac OS X).

For Bourne Shell, K Shell or Bash, type:

```
export LIBPATH=$XERCESCROOT/lib:$LIBPATH (on AIX)
export LD_LIBRARY_PATH=$XERCESCROOT/lib:$LD_LIBRARY_PATH (on Solaris, Linux)
export SHLIB_PATH=$XERCESCROOT/lib:$SHLIB_PATH (on HP-UX)
export DYLD_LIBRARY_PATH=$XERCESCROOT/lib:$DYLD_LIBRARY_PATH (on Mac OS X)
```

For C Shell, type:

```
setenv LIBPATH "$XERCESCROOT/lib:$LIBPATH" (on AIX)
setenv LD_LIBRARY_PATH "$XERCESCROOT/lib:$LD_LIBRARY_PATH" (on Solaris, Linux)
setenv SHLIB_PATH "$XERCESCROOT/lib:$SHLIB_PATH" (on HP-UX)
```

```
    setenv DYLD_LIBRARY_PATH "$XERCESCROOT/lib:$DYLD_LIBRARY_PATH" (Mac OS X)
```

Besides, if the parser is built with icu message loader (like IBM XML4C binaries), or message catalog loader, then you need to create a new environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

```
    export XERCESC_NLS_HOME=$XERCESCROOT/msg
    or
    setenv XERCESC_NLS_HOME=$XERCESCROOT/msg
```

The binary distribution has the built parser library and some samples executables. Please refer to the Samples for how to run the samples.

Cygwin

Source distribution

The Xerces-C++ source is available in the source distribution.

Install the Xerces-C++ source distribution xerces-c-src2_5_0.tar.gz by extracting the files from the compressed archive.

```
    tar -xvzf xerces-c-src2_5_0.tar.gz
```

This creates a 'xerces-c-src2_5_0' sub-directory containing the Xerces-C++ source distribution.

If you need to build the Xerces-C++ source after installation, please follow the Build Instructions.

Binary distribution

Install the binary distribution by running Cygwin [16] setup.exe. When you reach the "Packages" step of the Cygwin Setup wizard, expand the "Devel" category, then click in the "New" column next to "xerces-c-devel" until it reads "2.5.0-X".

This will install the necessary libraries and include files for the Xerces-C++ binary distribution.

If you wish to run programs linked to Xerces-C++ that were built in the Cygwin environment, you need to add your Cygwin "bin" directory to your Windows PATH environment variable. In typical Cygwin installations, the bin directory is in the Cygwin directory on the drive that windows is installed on. For instance, if windows is installed to C:\WINNT\System32, Your Cygwin bin directory may be "C:\cygwin\bin".

The binary distribution has the built parser library. Sample executables may be available in a future release on the Cygwin platform. In the meantime, they may be built from the source distribution by following the Build Instructions for "Building samples".

6 Build Instructions

Building on Windows and UNIX

Read the Building on Windows and UNIX document or jump directly to:

- Building Xerces-C++ on Windows using Microsoft Visual C++
- Building Xerces-C++ 64 bit binary on Windows XP using Intel C++ Compiler
- Building Xerces-C++ on Windows using Borland C++Builder
- Building Xerces-C++ on Windows using Borland C++ Compiler
- Building Xerces-C++ on Windows using Cygwin
- Building Xerces-C++ on UNIX platforms
- Building Xerces-C++ as a single-threaded library on Unix platforms

Building on Other Platforms

Read the Building on Other Platforms document or jump directly to:

- Building Xerces-C++ on iSeries (AS/400)
- Building Xerces-C++ on Macintosh

Other Build Instructions

Read the Other Build Instructions document or jump directly to:

- Building Xerces-C++ with ICU
- Building Xerces-C++ using RPM on Linux
- Building Xerces-C++ COM Wrapper on Windows
- Building User Documentation
- I wish to port Xerces to my favourite platform. Do you have any suggestions?
- What should I define XMLCh to be?
- Where can I look for more help?

7
Building on Windows and UNIX

Building Xerces-C++ on Windows using Microsoft Visual C++

Xerces-C++ source distribution comes with Microsoft Visual C++ projects and workspaces to help you build Xerces-C++. The following describes the steps you need to build Xerces-C++.

Building Xerces-C++ library

To build Xerces-C++ from the source distribution (using MSVC), you will need to open the workspace containing the project. If you are building your application, you may want to add the Xerces-C++ project inside your applications's workspace.

The workspace containing the Xerces-C++ project file and all other samples is in:

For MSVC Version 6:

```
    xerces-c-src2_5_0\Projects\Win32\VC6\xerces-all\xerces-all.dsw
```

For MSVC Version 7 (Visual C++.Net):

```
    xerces-c-src2_5_0\Projects\Win32\VC7\xerces-all\xerces-all.sln
```

Once you are inside MSVC, you need to build the project marked **XercesLib**.

If you want to include the Xerces-C++ project separately, you need to pick up:

```
    (For MSVC V6)
   xerces-c-src2_5_0\Projects\Win32\VC6\xerces-all\XercesLib\XercesLib.dsp
    (For MSVC V7)
   xerces-c-src2_5_0\Projects\Win32\VC7\xerces-all\XercesLib\XercesLib.vcproj
```

You must make sure that you are linking your application with the xerces-c_2.lib library and also make sure that the associated DLL is somewhere in your path.

> ***Note:*** *If you are working on the AlphaWorks version which uses ICU, you must have the ICU data DLL named* `icudata.dll` *available from your path setting. For finding out where you can get ICU from and build it, look at the How to Build ICU.*

Building samples

If you are using the source package, inside the same workspace (xerces-all.dsw), you'll find several other projects. These are for the samples. Select all the samples and right click on the selection. Then choose "Build (selection only)" to build all the samples in one shot.

If you are using the binary package, load the
xerces-c2_5_0-win32\samples\Projects\Win32\VC6\samples.dsw Microsoft Visual C++ workspace

inside your MSVC IDE. Then select all the samples and right click on the selection. Then choose "Build (selection only)" to build all the samples in one shot.

Building Xerces-C++ 64 bit binary on Windows XP using Intel C++ Compiler

Xerces-C++ source distribution comes with Microsoft Visual C++ NMake Files which work with Intel C++ Compiler. The following describes the steps you need to build Xerces-C++ 64 bit binary using Intel C++ Compiler.

Building Xerces-C++ library

Xerces-C++ source distribution provides a makefile `all.mak` which will build everything including samples, tests and the parser library.

```
cd xerces-c-src2_5_0\Projects\Win32\VC6\xerces-all\all
nmake -f all.mak "CFG=all - Win64 Release" CPP=ecl.exe
```

If you want to just build the Xerces-C++ parser library alone, then

```
cd xerces-c-src2_5_0\Projects\Win32\VC6\xerces-all\XercesLib
nmake -f XercesLib.mak "CFG=XercesLib - Win64 Release" CPP=ecl.exe
```

You must make sure that you are linking your application with the xerces-c_2.lib library and also make sure that the associated DLL is somewhere in your path.

Building Xerces-C++ on Windows using Borland C++Builder

Xerces-C++ sourec distribution comes with Borland C++Builder6 projects to help you build Xerces-C++. The following describes the steps you need to build Xerces-C++.

Building Xerces-C++ library

The library and demo projects are all contained in the Xerces-all project group:

- xerces-c-src2_5_0\Projects\Win32\BCB6\Xerces-all\Xerces-all.bpg

Each project in the group refers a directory below \Xerces-all. For example, the XercesLib project files are contained in the directory

- xerces-c-src2_5_0\Projects\Win32\BCB6\Xerces-all\XercesLib

To build any project, open the project manager. Double click on the project name. Then select "Project|Build" from the menu. For example, double click on XercesLib.dll in the manager. Then select "Project|Build XercesLib" from the menu. Once the library has been built, include XercesLib.lib with in application's project and place XercesLib.dll somewhere in your path.

Building Xerces-C++ on Windows using Borland C++ Compiler

Xerces-C++ sourec distribution comes with Borland C++ Compiler make files to help you build Xerces-C++. The following describes the steps you need to build Xerces-C++.

1. Change directory to `xerces-c-src2_5_0\Projects\Win32\BCC.551`
2. Run `MakeBuildDirs.bat`.
3. Then issue
   - `make -f Xerces-all.mak`

   to build the dll (without deprecated DOM API) and tests, or

- `make -f Xerces-all.mak -DWITHDEPRDOM=Y`
  to build the dll with deprecated DOM API (approx. 300k larger) and tests

Building Xerces-C++ on Windows using Cygwin

Do not jump into the build directly before reading this.

Xerces-C++ may be built in the Cygwin [16] environment for use by Cygwin applications. As with the UNIX platforms, Xerces-C++ on Cygwin uses GNU [17] tools. Therefore, with a couple of notable exceptions, Xerces-C++ on Cygwin is built using the same instructions as the UNIX platforms. The build environment variable XERCESCROOT must be set to the proper path containing the Xerces-C++ sources and **runConfigure** must be run with the "-pcygwin -cgcc -xg++" arguments.

Note that Cygwin is different from the UNIX platforms in the way that it finds libraries at run time. While UNIX platforms may use the environment variable LD_LIBRARY_PATH, Cygwin uses the PATH environment variable.

There is an issue with the gcc/g++ [18] compiler version 2.95, where C++ exceptions thrown from a dll will cause the application to crash, regardless of whether there is a "catch" statement. This bug doesn't occur in tests using gcc 3.1 or 3.2, so it is recommended that you use gcc 3.1 or higher.

Building Xerces-C++ on UNIX platforms

Xerces-C++ uses GNU [17] tools like Autoconf [19] and GNU Make [20] to build the system. You must first make sure you have these tools installed on your system before proceeding. If you do not have required tools, ask your system administrator to get them for you. These tools are free under the GNU Public License and may be obtained from the Free Software Foundation [17] .

Do not jump into the build directly before reading this.

Spending some time reading the following instructions will save you a lot of wasted time and support-related e-mail communication. The Xerces-C++ build instructions are a little different from normal product builds. Specifically, there are some wrapper-scripts that have been written to make life easier for you. You are free not to use these scripts and use Autoconf [19] and GNU Make [20] directly, but we want to make sure you know what you are by-passing and what risks you are taking. So read the following instructions carefully before attempting to build it yourself.

Besides having all necessary build tools, you also need to know what compilers we have tested Xerces-C++ on. The following table lists the relevant platforms and compilers.

Operating System	Compiler
32-bit binary	
Redhat Linux 7.2	Intel C++ Compiler v6, icc
AIX 5.1	xlC_r 5.0.2
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads
SuSE Linux 7.2 (S390)	g++ 2.95
64-bit binary	
Redhat Linux 7.2, IA64	Intel C++ Compiler v6, ecc
AIX 5.1	xlC_r 5.0.2
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads

If you are not using any of these compilers, you are taking a calculated risk by exploring new grounds. Your effort in making Xerces-C++ work on this new compiler is greatly appreciated and any problems you face can be addressed on the Xerces-C++ mailing list [21] .

Differences between the UNIX platforms: The description below is generic, but as every programmer is aware, there are minor differences within the various UNIX flavors the world has been bestowed with. The one difference that you need to watch out in the discussion below, pertains to the system environment variable for finding libraries. On **Linux** and **Solaris**, the environment variable name is called `LD_LIBRARY_PATH`, on **AIX** it is `LIBPATH`, on **Mac OS X** it is `DYLD_LIBRARY_PATH`, while on **HP-UX** it is `SHLIB_PATH`. The following discussion assumes you are working on Linux, but it is with subtle understanding that you know how to interpret it for the other UNIX flavors.

> ***Note:** If you wish to build Xerces-C++ with ICU, look at the Building Xerces-C++ with ICU. It tells you where you can get ICU and how to build Xerces-C++ with it.*

Setting build environment variables

Before doing the build, you must first set your environment variables to pick-up the compiler and also specify where you extracted Xerces-C++ on your machine. While the first one is probably set for you by the system administrator, just make sure you can invoke the compiler. You may do so by typing the compiler invocation command without any parameters (e.g. xlc_r, or g++, or cc) and check if you get a proper response back.

Next set your Xerces-C++ root path as follows:

```
    export XERCESCROOT=<full path to xerces-c-src2_5_0>
```

This should be the full path of the directory where you extracted Xerces-C++.

Building Xerces-C++ library

As mentioned earlier, to build Xerces-C++ from the source distribution, you must be ready with the GNU tools like autoconf [19] and gmake [20] before you attempt the build.

The autoconf tool is required on only one platform and produces a set of portable scripts (configure) that you can run on all other platforms without actually having the autoconf tool installed everywhere. In all probability the autoconf-generated script (called `configure`) is already in your `src/xercesc` directory. If not, type:

```
    cd $XERCESCROOT/src/xercesc
    autoconf
```

This generates a shell-script called `configure`. It is tempting to run this script directly as is normally the case, but wait a minute. Even if you are using the default compilers like gcc [18] and g++ [18] you need to export a few more environment variables before you can invoke configure.

Rather than make you to figure out what strange environment variables you need to use, we have provided you with a wrapper script that does the job for you. All you need to tell the script is what your compiler is, and what options you are going to use inside your build, and the script does everything for you. Here is what the script takes as input:

```

    runConfigure: Helper script to run "configure" for one of the supported
   platforms
    Usage: runConfigure "options"
           where options may be any of the following:
           -p <platform> (accepts 'aix', 'linux', 'freebsd',
                'netbsd', 'solaris', 'hp-10', 'hp-11', 'openserver', 'unixware',
                'os400', 'irix', 'ptx', 'tru64', 'macosx, 'cygwin', 'qnx')
                [required: no default]
           -c <C compiler name> (e.g. gcc, cc, xlc_r, icc or ecc)
                [default is make default; cc for gnu make]
           -x <C++ compiler name> (e.g. g++, CC, aCC, xlC_r, QCC
                icc or ecc) [default is make default; g++ for gnu make]
           -d (specifies that you want to build debug version)
                [default: no debug]
           -m <message loader> can be 'inmem', 'icu', 'MsgFile' or
                'iconv' [default: inmem]
           -n <net accessor> can be 'fileonly', 'libwww', 'socket' or
                'native' [default: socket]
           -t <transcoder> can be 'icu', 'Iconv400', 'Uniconv390',
                 'IconvFBSD', 'IconvGNU' or 'native'
                 [default: native]
           -r <thread option>  can be 'pthread' or 'dce'
                 (AIX, HP-11, and Solaris) or  'sproc' (IRIX) or 'none'
                 [default: pthread]
           -b <bitsToBuild> (accepts '64', '32') [default: 32]
           -l <extra linker options>
           -z <extra compiler options>
           -P <install-prefix>
           -C <any one extra configure options>
           -h (get help on the above commands)
```

> ***Note:*** *Xerces-C++ can be built as either a standalone library or as a library dependent on International Components for Unicode (ICU). For simplicity, the following discussion only explains standalone builds.*

Some additional explanation may be helpful for some of the options:

- **-m <message loader>, -t <transcoder>**

  If you specify `icu` as the value for either of these options, you must already have set the environment variable ICUROOT
- **-n <net accessor>**

  The default value `socket` handles HTTP URL's. The value `native` is only supported for macosx.

One of the common ways to build Xerces-C++ is as follows:

```
runConfigure -plinux -cgcc -xg++ -minmem -nsocket -tnative -rpthread
```

The response will be something like the following (extra line breaks have been added for readability). See especially the end, which tells you how **configure** was invoked.

```
Generating makefiles with the following options ...
Platform: linux
C Compiler: gcc
C++ Compiler: g++
Message Loader: inmem
Net Accessor: socket
Transcoder: native
Thread option: pthread
bitsToBuild option: 32
```

```
    Extra compile options:
    Extra link options:
    Extra configure options:
    Debug is OFF

    creating cache ./config.cache
    checking for gcc... gcc
    checking whether the C compiler
      (gcc   -w -O -DXML_USE_NATIVE_TRANSCODER -DXML_USE_INMEM_MESSAGELOADER
   -DXML_USE_PTHREADS
             -DXML_USE_NETACCESSOR_SOCKET       ) works... yes
    checking whether the C compiler
      (gcc   -w -O -DXML_USE_NATIVE_TRANSCODER -DXML_USE_INMEM_MESSAGELOADER
   -DXML_USE_PTHREADS
             -DXML_USE_NETACCESSOR_SOCKET       ) is a cross-compiler... no
    checking whether we are using GNU C... yes
    checking whether gcc accepts -g... yes
    checking for c++... g++
    checking whether the C++ compiler
      (g++   -w -O -DXML_USE_NATIVE_TRANSCODER -DXML_USE_INMEM_MESSAGELOADER
   -DXML_USE_PTHREADS
             -DXML_USE_NETACCESSOR_SOCKET      ) works... yes
    checking whether the C++ compiler
      (g++   -w -O -DXML_USE_NATIVE_TRANSCODER -DXML_USE_INMEM_MESSAGELOADER
   -DXML_USE_PTHREADS
             -DXML_USE_NETACCESSOR_SOCKET      ) is a cross-compiler... yes
    checking whether we are using GNU C++... yes
    checking whether g++ accepts -g... yes
    checking for a BSD compatible install... /usr/bin/install -c
    checking for autoconf... autoconf
    checking how to run the C preprocessor... gcc -E
    checking for ANSI C header files... yes
    checking for XMLByte... no
    checking host system type... i686-pc-linux-gnu
    updating cache ./config.cache
    creating ./config.status
    creating Makefile
    creating util/Makefile
    creating util/Transcoders/ICU/Makefile
    creating util/Transcoders/Iconv/Makefile
    creating util/Transcoders/Iconv390/Makefile
    creating util/Transcoders/Uniconv390/Makefile
    creating util/Transcoders/Iconv400/Makefile
    creating util/Transcoders/IconvFBSD/Makefile
    creating util/Transcoders/MacOSUnicodeConverter/Makefile
    creating util/Platforms/Makefile
    creating util/Platforms/Solaris/Makefile
    creating util/Platforms/AIX/Makefile
    creating util/Platforms/Linux/Makefile
    creating util/Platforms/FreeBSD/Makefile
    creating util/Platforms/HPUX/Makefile
```

```
 creating util/Platforms/OS390/Makefile
 creating util/Platforms/OS400/Makefile
 creating util/Platforms/IRIX/Makefile
 creating util/Platforms/PTX/Makefile
 creating util/Platforms/OpenServer/Makefile
 creating util/Platforms/UnixWare/Makefile
 creating util/Platforms/Tru64/Makefile
 creating util/Platforms/MacOS/Makefile
 creating util/Compilers/Makefile
 creating util/MsgLoaders/InMemory/Makefile
 creating util/MsgLoaders/ICU/Makefile
 creating util/MsgLoaders/MsgCatalog/Makefile
 creating util/MsgLoaders/MsgFile/Makefile
 creating util/NetAccessors/Socket/Makefile
 creating util/NetAccessors/libWWW/Makefile
 creating util/NetAccessors/MacOSURLAccessCF/Makefile
 creating util/regx/Makefile
 creating validators/Makefile
 creating validators/common/Makefile
 creating validators/datatype/Makefile
 creating validators/DTD/Makefile
 creating validators/schema/Makefile
 creating validators/schema/identity/Makefile
 creating framework/Makefile
 creating dom/Makefile
 creating dom/impl/Makefile
 creating dom/deprecated/Makefile
 creating parsers/Makefile
 creating internal/Makefile
 creating sax/Makefile
 creating sax2/Makefile
 creating ../../obj/Makefile

 Having build problems?
 Read instructions at http://xml.apache.org/xerces-c/build.html
 Still cannot resolve it?
 Find out if someone else had the same problem before.
 Go to http://marc.theaimsgroup.com/?l=xerces-c-dev

 In future, you may also directly type the following commands to create the
Makefiles.

 export TRANSCODER="NATIVE"
 export MESSAGELOADER="INMEM"
 export NETACCESSOR="Socket"
 export THREADS="pthread"
 export BITSTOBUILD="32"
 export CC="gcc"
 export CXX="g++"
 export CXXFLAGS=" -w -O -DXML_USE_NATIVE_TRANSCODER
-DXML_USE_INMEM_MESSAGELOADER
```

```
                    -DXML_USE_PTHREADS -DXML_USE_NETACCESSOR_SOCKET "
    export CFLAGS=" -w -O -DXML_USE_NATIVE_TRANSCODER -DXML_USE_INMEM_MESSAGELOADER
                  -DXML_USE_PTHREADS -DXML_USE_NETACCESSOR_SOCKET "
    export LDFLAGS=" "
    export LIBS=" -lpthread "
    configure

    If the result of the above commands look OK to you, go to the directory
    $HOME/xerces-c-src2_5_0/src/xercesc and type "gmake" to make the XERCES-C
   system.
```

So now you see what the wrapper script has actually been doing! It has invoked `configure` to create the Makefiles in the individual sub-directories, but in addition to that, it has set a few environment variables to correctly configure your compiler and compiler flags too.

Now that the Makefiles are all created, you are ready to do the actual build.

```
    gmake
```

Is that it? Yes, that's all you need to build Xerces-C++.

Building samples

The installation process for the samples is same on all UNIX platforms.

```
    cd xerces-c2_5_0-linux/samples
    ./runConfigure -p<platform> -c<C_compiler> -x<C++_compiler>
    gmake
```

This will create the object files in each sample directory and the executables in ' xerces-c2_5_0-linux/bin' directory.

Note that **runConfigure** is just a helper script and you are free to use **./configure** with the correct parameters to make it work on any platform-compiler combination of your choice. The script needs the following parameters:

```

    runConfigure: Helper script to run "configure" for one of the supported
   platforms
    Usage: runConfigure "options"
           where options may be any of the following:
           -p <platform> (accepts 'aix', 'beos', 'linux', 'freebsd', 'netbsd',
                     'solaris', 'hp-10', 'hp-11', 'openserver', 'unixware',
                     'os400', 'irix', 'ptx', 'tru64', 'macosx', 'cygwin')
                     [required; no default]
           -c <C compiler name> (e.g. gcc, cc, xlc_r, icc or ecc)
                     [default is make default; cc for gnu make]
           -x <C++ compiler name> (e.g. g++, CC, aCC, xlC_r, QCC,
                     icc or ecc)  [default is make default; g++ for gnu make]
           -d (specifies that you want to build debug version) [default: not debug]
           -r <thread option> can be 'pthread' or 'dce'
                 (AIX, HP-11, and Solaris) or 'sproc' (IRIX) or 'none'
                 [default: pthread]
           -b <bitsToBuild> (accepts '64', '32') [default: 32]
           -l <extra linker options>
           -z <extra compiler options>
```

```
          -h (get help on the above commands)
```

> ***Note: NOTE:****The code samples in this section assume that you are working on the Linux binary drop. If you are using some other UNIX flavor, please replace '-linux' with the appropriate platform name in the code samples.*

To delete all the generated object files and executables, type:

```
    gmake clean
```

Building Xerces-C++ as a single-threaded library on Unix platforms

To build a single-threaded library on Unix platforms you have to update one or more of the following files `Makefile.incl, Makefile.in, runConfigure`. The following steps guide you to create a single-threaded library for each platform:

For Aix -

- Replace `xlc_r` and `xlC_r` libraries with `xlc` and `xlC` respectively
- Replace `makeC++SharedLib_r` with `makeC++SharedLib`
- Remove the flag `-D_THREAD_SAFE`
- Remove inclusion of any threaded library directories from the `LIBPATH`
- Remove inclusion of `-lpthreads` and `-lpthread_compat`
- Add `-DAPP_NO_THREADS` to define the variable under AIX specific options in `Makefile.incl`

For Solaris -

- Add `-DAPP_NO_THREADS` to define the variable under SOLARIS specific options in `Makefile.incl`
- Remove compiler switch `-mt`
- Remove `-D_REENTRANT` flag from the 'compile' options
- Remove inclusion of `-lpthread`

For Linux -

- Add `-DAPP_NO_THREADS` to define the variable under LINUX specific options in `Makefile.incl`
- Remove `-D_REENTRANT` flag from the 'compile' options
- Remove inclusion of `-lpthread`

For HPUX -

- Add `-DAPP_NO_THREADS` to define the variable under HP specific options in `Makefile.incl`
- Remove inclusion of `-lpthread` and `-lcma`
- Remove threading defines like `-D_PTHREADS_DRAFT4 , -DXML_USE_DCE`

8 Building on Other Platforms

Building Xerces-C++ on iSeries (AS/400)

The following addresses the requirements and build of Xerces-C++ natively on the iSeries.

Building Xerces-C++ library

Requirements:

- OS/400 `QSHELL` interpreter installed (install base option 30, operating system)
- OS/400 - Portable App Solutions Environment (PASE) installed (install base option 33, operating system)
- QShell Utilities, PRPQ 5799-XEH
- iSeries Tools for Developers, PRPQ 5799-PTL (these are the gnu utilities)

Compiler:

- For v4r5m0: ILE C++ for AS/400, PRPQ 5799-GDW
- For v5: WebSphere Development ToolsSet, 5722-WDS (installed option 52, Compiler - ILE C++)

Recommendations:

- There is one option when building the XML4C parser on iSeries. For code page translation, you can use the iSeries native `Iconv400` support or ICU as the transcoder plug in. If you choose ICU, follow the instructions to build the ICU service program with the ICU download. Those instructions are not included here.
- We recommend the use of `Iconv400`. The binary posted on Alphaworks uses Iconv400.

Setup Instructions:

- Make sure that you have the requirements installed on your iSeries. We highly recommend that you read the write up that accompanies the iSeries Tools for Developers PRPQ. There are install instructions as well as information about how modules, programs and service programs can be created in Unix-like fashion using gnu utilities. Note that symbolic links are use in the file system to point to actual iSeries `*module`, `*pgm` and `*srvpgm` objects in libraries.
- Download the source zip file (NT version) directly to an iSeries IFS directory after creating a directory (eg. /XML4Cxxx) and then extract the source using a mapped drive. To do this, from Windows Explorer, select Tools - > Map Network Drive. Then select an available drive (e.g. F:) and specify an iSeries system you want to extract the zip file to (e.g. \\<your iSeries name>\root). Click on Finish. Then find the .zip file and right click on it and select Extract To ... Then select the files you want to extract to the iSeries system.
- Create iSeries target library. This library will be the target for the resulting modules and Xerces-C++ service program. You will specify this library on the OUTPUTDIR environment variable in step 4.
- Set up the following environment variables in your build process (use `ADDENVVAR` or `WRKENVVAR`

`CL` commands):

```
 XERCESCROOT - <the full path up to the Xerces-C++ src directory, but not
including 'src'>
 MAKE   - '/usr/bin/gmake'
 OUTPUTDIR  - <identifies target iSeries library for *module, *pgm and *srvpgm
objects>
 ICUROOT - (optional if using ICU)  <the path of your ICU includes>
```

- For v4r5m0 systems, add QCXXN, to your build process library list. This results in the resolution of CRTCPPMOD used by the icc compiler.

You may want to put the environment variables and library list setup instructions in a `CL` program so you will not forget these steps during your build.

Configure

To configure the make files for an iSeries build do the following under Qsh:

```
qsh:
cd <full path to Xerces-C++>/src/xercesc
runConfigure -p os400 -x icc -c icc -m inmem -t Iconv400
```

Troubleshooting:

```
error: configure: error: installation or configuration problem:
C compiler cannot create executables.
```

If during runConfigure you see the above error message, it can mean one of a few things. Either QCXXN is not on your library list **OR** the `runConfigure` cannot create the temporary modules (`CONFTest1`, etc) it uses to test out the compiler options or `PASE` is not installed. The second reason happens because the test modules already exist from a previous run of `runConfigure`. To correct the problem, do the following:

```
CL:
DLTMOD <OUTPUTDIR library>/CONFT* and
DLTPGM <OUTPUTDIR library>/CONFT*
```

Build

```
qsh:
cd <full path to Xerces-C++>/src/xercesc
gmake
```

The above gmake should result in a service program being created in your specified library and a symbolic link to that service program placed in <path to Xerces-C++/lib >. It is highly possible that the service program will not create however due to number of modules and path names, see trouble shooting for the workaround.

After the service program has successfully been created and a link established, you can either bind your XML application programs directly to the parser's service program via the `BNDSRVPGM` option on the `CRTPGM` or `CRTSRVPGM` command or you can specify a binding directory on your `icc` command. To specify an archive file to bind to, use the `-L, -l` binding options on icc. An archive file on iSeries is a binding directory. To create an archive file, use `qar` command. (see the iSeries Tools for Developers write up).

After building the Xerces-C service program, create a binding directory by doing the following (note, this binding directory is used when building the samples. Also, note that the .a file below can have a different name based on the parser version (using apache xerces versioning)):

```
    qsh:
    cd <full path to Xerces-C++>/lib
    qar -cuv libxerces-c25.0.so *.o
    will results in
    command = CRTBNDDIR BNDDIR(yourlib/libxercesc)
   TEXT('/yourlib/Xerces-C++/lib/libxerces-c25.0.so')
    command = ADDBNDDIRE BNDDIR(yourlib/libxercesc) OBJ((yourlib/LIBXERCESC *SRVPGM)
   )
```

Troubleshooting gmake problem:

Due to the number of modules (the .o symbolic links) that make up the service program and the path to get to those modules, the qshell ld request to create the service program will likely fail because the request is too large, you may get a message like the following at the end of the gmake request:

```
    FAILURE: spawnp()  with errno = 3491
    GMAKE: vfork: Argument list too long.
```

If this is the case, you can manually create the service program by doing the following:

```
    CL:
    CRTSRVPGM  (<OUTPUTDIR-library>/libxercesc)  MODULE(<OUTPUTDIR-library>/*ALL)
   EXPORT(*ALL) TEXT('XML4C parser version xxx')
    OPTION(*DUPPROC *DUPVAR)
```

Note that if you manually create the service program you want to make sure that you do not include any CONFT* modules or samples modules in the OUTPUTDIR library. After the service program is manually created you can add a symbolic link to the service program into the appropriate /lib directory by qsh:

```
    qsh:
    cd <full path to Xerces-C++>/lib
    ln -s /qsys.lib/<outputdir>.lib/libxercesc.srvpgm   libxerces-c25.0.so
    qar -cuv libxerces-c25.0.so *.o
```

If you are on a v4 system using the ILE C++ PRPQ compiler (which is referred to as the 'old' compiler) you will get compiler errors requiring a few manual changes to the source:

- src/xercesc/dom/impl/DOMDocumentImpl.cpp
- src/xercesc/dom/impl/DOMDocumentImpl.hpp
- src/xercesc/dom/deprecated/DocumentImpl.cpp
- src/xercesc/dom/deprecated/DocumentImpl.hpp
- src/xercesc/validators/common/ContentSpecNode.hpp

Update the following routines in src/xercesc/dom/deprecated/DocumentImpl.cpp as follows:

```
     void DocumentImpl::setUserData(NodeImpl* n, void* data)
     {
            if (!userData && data)
      #ifdef __OS400__
                    userData = new RefHashTableOf<char>(29, false, new HashPtr());
      #else
                    userData = new RefHashTableOf<void>(29, false, new HashPtr());
      #endif
            if (!data && userData)
                    userData->removeKey((void*)n);
            else
      #ifdef __OS400__
                    userData->put((void*)n,(char*)data);
      #else
                    userData->put((void*)n,data);
      #endif
     }

     void* DocumentImpl::getUserData(NodeImpl* n)
     {
            if (userData)
      #ifdef __OS400__
                    return (void*)userData->get((void*)n);
      #else
                    return userData->get((void*)n);
      #endif
            else
                    return null;
     }
```

To update src/xercesc/dom/deprecated/DoumentImpl.hpp as follows:

```

     #ifdef __OS400__
            RefHashTableOf<char>         *userData;
     #else

            RefHashTableOf<void>         *userData;
     #endif

```

Update the following routines in src/xercesc/dom/impl/DOMDocumentImpl.cpp as follows:

```

     void DOMDocumentImpl::setUserData(DOMNode* n, void* data)
     {
            if (!fUserData && data)
     #ifdef __OS400__
                    fUserData = new (this) RefHashTableOf<char>(29, false, new
   (this) HashPtr());
     #else
                    fUserData = new (this) RefHashTableOf<void>(29, false, new
   (this) HashPtr());
     #endif
```

```
        if (!data && fUserData)
                fUserData->removeKey((void*)n);
        else
#ifdef __OS400__
                fUserData->put((void*)n,(char*)data);
#else
                fUserData->put((void*)n,data);
#endif
}

void* DOMDocumentImpl::getUserData(const DOMNode* n) const
{
        if (fUserData)
#ifdef __OS400__
                return (void*) fUserData->get((void*)n);
#else
                return fUserData->get((void*)n);
#endif

        else
                return 0;
}
```

To update src/xercesc/dom/impl/DOMDocumentImpl.hpp:

```
#ifdef __OS400__
   RefHashTableOf<char>        *fUserData;
#else
   RefHashTableOf<void>        *fUserData;
#endif
```

Update validators/common/ContentSpecNode.hpp removing the following:

```
#ifndef __OS400__
inline
#endif
ContentSpecNode::~ContentSpecNode()
```

To build for transcoder ICU:

1. Make sure you have an `ICUROOT` path set up so that you can find the ICU header files (usually `/usr/local`)
2. Make sure you have created a binding directory (symbolic link) in the file system so that you can bind the Xerces-C++ service program to the ICU service program and specify that on the `EXTRA_LINK_OPTIONS` in `src/xercesc/Makefile.incl` (usually the default is a link in `/usr/local/lib`).

Building Samples on iSeries

Note that the samples will create programs bind to the BND directory object created by qar referenced above.

```
    qsh
    cd <full path to Xerces-C++>/samples
    runConfigure -p os400 -x icc -c icc
    gmake
```

Building Xerces-C++ on Macintosh

The Xerces-C++ Mac port has the key following attributes:

1. Built atop CoreServices APIs and a limited number of Carbon APIs; supports builds for both Mac OS Classic, Carbon, and Mac OS X systems.
2. Has a Mac OS native transcoder that utilizes the built-in Mac OS Unicode converter [MacOSUnicodeConverter].
3. Has two Mac OS native netaccessor classes. The first is based on Carbon and classic supported URLAccess and may be used in the broadest variety of configurations [MacOSURLAccess]. The second [MacOSURLAccessCF] is based on CFURLAccess, which requires either Carbon or Mac OS X CoreServices.framework. This second NetAccessor is useful in Mac OS X configurations where reliance on the full Carbon.framework would prohibit execution of the Xerces code in a remote context that has no access to the GUI.
4. Supports builds from Metroworks CodeWarrior, Apple Xcode, and Mac OS X shell. Projects for Apple Project Builder are still included, but may not be up to date (you may need to revise the projects to accomodate recent file additions, deletions, or other changes in Xerces-C++).

Using Xerces-C++ with CodeWarrior

Xerces-C++ and CodeWarrior:

Xerces-C++ may be built with CodeWarrior under Mac OS Classic or Mac OS X. Since the Xerces-C++ code contains some files with very long names, and earlier versions of Mac OS, as well as earlier versions of CodeWarrior, did not support file names longer than 32 characters, CodeWarrior 8.0 is required. If you are building Xerces-C++ on a Mac OS 9 system, be extremely carefull in how to unpack and/or transfer the Xerces-C++ files to that system, to ensure that their file names are not trancated in the process.

Installing Xerces-C++ for use with CodeWarrior:

Note: versions of CodeWarrior prior to 8.0 did not support HFS+ long file names, and thus required special steps to alter the file names prior to use. This restriction has been removed for CodeWarrior 8.0, and the projects now directly reference the unaltered source tree. The project files in this release require CodeWarrior v8.0 or higher.

It is extemely important to ensure that you retrieve and unpack the sources with a tool that does not truncate file names. The command line gnutar utility on Mac OS X will do the right thing; older versions of StuffIt truncate names to 31 characters as they unpack tar archives, though versions >= 7.0.1 seem to work, at least on Mac OS X. The command line tool tar will truncate path names that get too long; gnutar should be used instead. Failure to heed these warnings will result in broken projects.

Building Xerces-C++ with CodeWarrior:

- Run CodeWarrior (requires CodeWarrior 8.0 with support for long file names).
- Import the project Projects/MacOS/CodeWarrior/XercesLib/XercesLib.mcp.xml, saving it back out to the same directory as XercesLib.mcp.
- This project contains five build targets that build all combinations of classic, carbon, debug, and release versions, with an all target that builds all of these. Build any or all of these.

Building Xerces-C++ Samples with CodeWarrior:

A CodeWarrior project is included that builds the DOMPrint sample. This may be used as an example from which to build additional sample projects. Please read the following important notes:

- Once again, it is required that you import the .xml version of the project file, and save it back out.
- The Xerces-C++ sample programs are written to assume a command line interface. To avoid making Macintosh-specific changes to these command line programs, we have opted to instead require that you make a small extension to your CodeWarrior runtime that supports such command line programs. Please read and follow the usage notes in XercesSampleSupport/XercesSampleStartupFragment.c.

Building Xerces-C++ with Xcode

Projects are included to build the Xerces-C++ library and DOMPrint sample under Apple's Xcode for Mac OS X. The following notes apply:

- Be sure to heed warnings under "special instructions" below regarding which tools must be used to unpack archives: gnutar is your friend.
- The Xcode project builds XercesLib as the framework Xerces.framework. This framework, however, does not currently include a correct set of public headers. Any referencing code must have an include path directive that points into the Xerces-C++ src directory.
- The DOMPrint project illustrates one such usage of the Xerces.framework.

Projects for Apple's Project Builder environment, which is no longer supported under Mac OS X 10.3, are provided but may be out of date. You may need to add or delete files from the project in order to support changes in Xerces-C++. Please feel free to submit patches against these projects If you care about the Project Builder projects, and want to keep them up to date.

Building Xerces-C++ from the Mac OS X command line

Support for Mac OS X command line builds is now included in the standard "unix" Xerces-C++ build infrastructure.

- In general, the Mac OS X command line build follows the generic unix build instructions. You need to set your XERCESCROOT environment variable, `./runConfigure`, and `make`. Be sure to heed warnings under "special instructions" below regarding which tools must be used to unpack archives: gnutar is your friend.

```
    setenv XERCESCROOT "<xerces-c-directory>"
    cd src/xercesc
    ./runConfigure -p macosx -n native
    make
```

- Similar instructions apply to build the samples and tests, though the `-n` flag is not used in these cases:

```
    cd samples
    ./runConfigure -p macosx
    make
```

Special usage information for Xerces-C++ on the Macintosh

Unpacking the tar archive

It is extemely important to ensure that you retrieve and unpack the sources with a tool that does not truncate file names. The command line gnutar utility on Mac OS X will do the right thing; older versions of StuffIt truncate names to 31 characters as they unpack tar archives, though versions >= 7.0.1 seem to work, at least on Mac OS X. The command line tool tar will truncate path names that get too long; gnutar should be used instead. Failure to heed these warnings will result in broken projects.

File Path Specification

Apart from the build instructions, above, the most important note about use of Xerces-C++ on the Macintosh is that Xerces-C++ expects all filename paths to be specified in unix syntax. If running natively under a Mac OS X system, this path will be the standard posix path as expected by the shell. The easiest means of creating and interpreting these paths will be through the routines `XMLCreateFullPathFromFSRef` and `XMLParsePathToFSRef` as declared in the file `MacOSPlatformUtils.hpp`. `FSSpec` variants of these routines are also supplied.

Mac OS Version Compatibility

Xerces-C++ requires that several key components of the Mac OS be relatively up to date. It should be readily compatible with any system above Mac OS 9.0. Compatibility with earlier systems may perhaps be achieved if you can install appropriate components.

Required components are:

- Unicode Converter and Text Encoding Converter. These provide the base transcoding service used to support Xerces-C++ transcoding requirements.

Optional components are:

- URLAccess. Provides NetAccessor support to Xerces-C++ for use in fetching network referenced entities. If URLAccess is not installed, any such references will fail; the absence of URLAccess, however, will not in itself prevent Xerces-C++ from running. If Xerces-C++ is configured to use MacOSURLAccessCF, then URLAccess (and thus Carbon) is not required, but CoreServices.framework is required for Mac OS X.
- Multiprocessing library. Provides mutual exclusion support. Once again, the routines will back down gracefully if Multiprocessing support is not available.
- HFS+ APIs. If HFS+ APIs are available, all file access is performed using the HFS+ fork APIs to support long file access, and to support long unicode compliant file names. In the absence of HFS+ APIs, classic HFS APIs are used instead.

9 Other Build Instructions

Building Xerces-C++ with ICU

Xerces-C++ may be built in stand-alone mode using native encoding support and also using ICU where you get support over 180 different encodings and/or locale specific message support. ICU stands for International Components for Unicode and is an open source distribution from IBM. You can get ICU libraries [22] from IBM's developerWorks site [23] or go to the ICU download page [24] directly.

> ***Note: Important:*** *Please remember that* ***ICU and Xerces-C++ must be built with the same compiler****, preferably with the same version. You cannot for example, build ICU with a threaded version of the xlC compiler and build Xerces-C++ with a non-threaded one.*

Building on Windows

There are two options to build Xerces-C++ with ICU on Windows. One is to use the MSDEV GUI environment, and the other is to invoke the compiler from the command line.

Using, the GUI environment, requires one to edit the project files. Here, we will describe only the second option. It involves using the perl script '`packageBinaries.pl`'.

Prerequisites:

- Perl 5.004 or higher
- Cygwin tools or MKS Toolkit
- zip.exe

Extract Xerces-C++ source files from the .zip archive using WinZip, say in the root directory (an arbitrary drive x:). It should create a directory like '`x:\xerces-c-src2_5_0`'.

Extract the ICU files, using WinZip, in root directory of the disk where you have installed Xerces-C++, sources. After extraction, there should be a new directory '`x:\icu`' which contains all the ICU source files.

Start a command prompt to get a new shell window. Make sure you have perl, cygwin tools (`uname`, `rm`, `cp`, ...), and `zip.exe` somewhere in the path. Next setup the environment for MSVC using `VCVARS32.BAT`' or a similar file. Then at the prompt enter:

```
    set XERCESCROOT=x:\xerces-c-src2_5_0
    set ICUROOT=x:\icu
    cd x:\xerces-c-src2_5_0\scripts
```

To build with ICU, either specify using ICU transcoding service,

```
  perl packageBinaries.pl -s x:\xerces-c-src2_5_0 -o x:\temp\xerces-c2_5_0-win32
 -t icu
```

or specify using ICU message loader service

```
  perl packageBinaries.pl -s x:\xerces-c-src2_5_0 -o x:\temp\xerces-c2_5_0-win32
 -m icu
```

(Match the source directory to your system; the target directory can be anything you want.)

If everything is setup right and works right, then you should see a binary drop created in the target directory specified above. This script will build both ICU and Xerces-C++, and copy the files (relevant to the binary drop) to the target directory.

If the parser is built with icu message loader (as mentioned above), or message catalog loader, you need an environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

For a description of options available, you can enter:

```
  perl packageBinaries.pl
```

Building on UNIX

Extract Xerces-C++ source files into, say, the home directory ($HOME). It should create a directory like '`$HOME/xerces-c-src2_5_0`'.

Extract the ICU files into the same directory where you have installed Xerces-C++ sources. After extraction, there should be a new directory '`$HOME/icu`' which contains all the ICU source files.

Build the ICU according to the ICU Build instruction in ICU Readme [25] . Then have its dll, `libicuuc*` and `libicudt*` available from your library search path.

Then build the Xerces-C++ with ICU. This is similar to building a standalone Xerces-C++ library as instructed in "Building Xerces-C++ on UNIX platforms"; except that you have to specify the transcoder option `'-t icu'` and/or the message loader option `'-m icu'`. For example:

```
  runConfigure -plinux -cgcc -xg++ -minmem -nsocket -ticu -rpthread
```

Or instead of building the ICU and Xerces-C++ manually in two steps, you can use the bundled perl script '`packageBinaries.pl`' which will build both of them in one step. For example:

```
  export XERCESCROOT=$HOME/xerces-c-src2_5_0
  export ICUROOT=$HOME/icu
  cd $HOME/xerces-c-src2_5_0/scripts
```

To build with ICU, either specify using ICU transcoding service,

```
  perl packageBinaries.pl -s $HOME/xerces-c-src2_5_0 -o
 $HOME/temp/xerces-c2_5_0-aix -t icu
```

or specify using ICU message loader service

```
 perl packageBinaries.pl -s $HOME/xerces-c-src2_5_0 -o
$HOME/temp/xerces-c2_5_0-aix -m icu
```

If the parser is built with icu message loader (as mentioned above), or message catalog loader, you need an environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

Building Xerces-C++ using RPM on Linux

Xerces-C++ may be built from the distributed source archive directly on Linux using RPM. For example:

```
rpm -ta xerces-c-src2_5_0.tar.gz (rpm 4.0 and older)
or
rpmbuild -ta xerces-c-src2_5_0.tar.gz (rpm 4.1 and later; ships with RedHat 8)
```

The Xerces-C++ RPM specificattion can be found in `xerces-c-src2_5_0/xerces-c.spec`.

Please refer to the RPM-HOWTO [26] , for more RPM related information.

Building Xerces-C++ COM Wrapper on Windows

To build the COM module for use with XML on Windows platforms, you must first set up your machine appropriately with necessary tools and software modules and then try to compile it. The end result is an additional library that you can use along with the standard Xerces-C++ for writing VB templates or for use with IE 5.0 using JavaScript.

Setting up your machine for COM

To build the COM project you will need to install the MS PlatformSDK. Some of the header files we use don't come with Visual C++ 6.0. You may download it from Microsoft's Website at http://www.microsoft.com/msdownload/platformsdk/setuplauncher.htm or directly FTP it from ftp://ftp.microsoft.com/developr/PlatformSDK/April2000/Msi/WinNT/x86/InstMsi.exe.

The installation is huge, but you don't need most of it. So you may do a **custom install** by just selecting "Build Environment" and choosing the required components. First select the top level Platform SDK. Then click the down arrow and make all of the components unavailable. Next open the "Build Environment" branch and select only the following items:

- Win32 API
- Component Services
- Web Services - Internet Explorer

Important: When the installation is complete you need to update VC6's include path to include `..\platformsdk\include\atl30`. You do this by choosing "Tools - > Options - > Directories". This path should be placed *second* after the normal PlatformSDK include. You change the order of the paths by clicking the up and down arrows.

> ***Note:*** *The order in which the directories appear on your path is important. Your first include path should be* `..\platformsdk\include`*. The second one should be* `..\platformsdk\include\atl30`*.*

Building COM module for Xerces-C++

Once you have set up your machine, build Xerces-C++ COM module by choosing the project named 'xml4com' inside the workspace. Then select your build mode to be **xml4com - Win32 Release MinDependency**. Finally build the project. This will produce a DLL named `xerces-com.dll` which needs to be present in your path (on local machine) before you can use it.

Testing the COM module

There are some sample test programs in the `test/COMTest` directory which show examples of navigating and searching an XML tree using DOM. You need to browse the HTML files in this directory using IE 5.0. Make sure that your build has worked properly, specially the registration of the ActiveX controls that happens in the final step.

You may also want to check out the NIST DOM test suite at http://xw2k.sdct.itl.nist.gov/BRADY/DOM/. You will have to modify the documents in the NIST suite to load the Xerces COM object instead of the MSIE COM object.

Building User Documentation

The user documentation (this very page that you are reading on the browser right now), was generated using an XML application called StyleBook. This application makes use of Xerces-J and Xalan to create the HTML file from the XML source files. The XML source files for the documentation are part of the Xerces-C++ module. These files reside in the `doc` directory.

Pre-requisites for building the user documentation are:

- JDK 1.2.2 (or later).
- Xerces-J 1.0.1.**bundled**
- Xalan-J 0.19.2.**bundled**
- Stylebook 1.0-b2. **bundled**
- The Apache Style files (dtd's and .xsl files).**bundled**

Invoke a command window and setup PATH to include the JDK 1.2.2 bin directory

Next, cd to the Xerces-C++ source drop root directory, and enter

- Under Windows:
```
createDocs
```
- Under Unix's:
```
sh createDocs.bat
```

This should generate the .html files in the 'doc/html' directory.

I wish to port Xerces to my favourite platform. Do you have any suggestions?

All platform dependent code in Xerces has been isolated to a couple of files, which should ease the porting effort. Please refer to Porting Guidelines for further details.

What should I define XMLCh to be?

XMLCh should be defined to be a type suitable for holding a utf-16 encoded (16 bit) value, usually an `unsigned short`.

All XML data is handled within Xerces-C++ as strings of XMLCh characters. Regardless of the size of the type chosen, the data stored in variables of type XMLCh will always be utf-16 encoded values.

Unlike XMLCh, the encoding of wchar_t is platform dependent. Sometimes it is utf-16 (AIX, Windows), sometimes ucs-4 (Solaris, Linux), sometimes it is not based on Unicode at all (HP/UX, AS/400, system 390).

Some earlier releases of Xerces-C++ defined XMLCh to be the same type as wchar_t on most platforms, with the goal of making it possible to pass XMLCh strings to library or system functions that were expecting wchar_t parameters. This approach has been abandoned because of

- Portability problems with any code that assumes that the types of XMLCh and wchar_t are compatible
- Excessive memory usage, especially in the DOM, on platforms with 32 bit wchar_t.
- utf-16 encoded XMLCh is not always compatible with ucs-4 encoded wchar_t on Solaris and Linux. The problem occurs with Unicode characters with values greater than 64k; in ucs-4 the value is stored as a single 32 bit quantity. With utf-16, the value will be stored as a "surrogate pair" of two 16 bit values. Even with XMLCh equated to wchar_t, xerces will still create the utf-16 encoded surrogate pairs, which are illegal in ucs-4 encoded wchar_t strings.

Where can I look for more help?

If you have read this page, followed the instructions, and still cannot resolve your problem(s), there is more help. You can find out if others have solved this same problem before you, by checking the Apache XML mailing list archives at http://marc.theaimsgroup.com/?l=xerces-c-dev [27] and the Bugzilla [28] Apache bug database.

10 FAQs

Distributing Xerces-C++

What platforms / compilers are being used to build the binary distribution kits?

Xerces binaries has been built on the following platforms with these compilers

Operating System	**Compiler**
32-bit binary	
Windows NT 4.0 SP5	MSVC 6.0 SP3
Redhat Linux 7.2	Intel C++ Compiler v6, icc
AIX 5.1	xlC_r 5.0.2
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads
SuSE Linux 7.2 (S390)	g++ 2.95
64-bit binary	
Windows XP, IA64	Intel C++ Compiler v7, ecl
Redhat Linux 7.2, IA64	Intel C++ Compiler v6, ecc
AIX 5.1	xlC_r 5.0.2
Solaris 2.7	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads

What are the differences between Xerces-C and XML4C?

Xerces-C has intrinsic support for ASCII, UTF-8, UTF-16 (Big/Small Endian), UCS4 (Big/Small Endian), EBCDIC code pages IBM037, IBM1047 and IBM1140 encodings, ISO-8859-1 (aka Latin1) and Windows-1252. This means that it can parse input XML files in these above mentioned encodings.

However, if you wish to parse XML files in any other encodings, say in Shift-JIS, Big5 etc., then you cannot use Xerces-C. XML4C addresses this need. It combines Xerces-C and International Components for Unicode (ICU) [22] and provides support for over 100 different encodings. XML4C also uses ICU Resource Bundle to load the messages.

ICU is also an open source project but is licensed under the X License [29] . XML4C is published by IBM and can be downloaded from their Alphaworks [30] site. The license to use XML4C is simply to comply with the Apache license (because of Xerces-C) and X License (because of ICU).

XML4C binaries are published for the same set of platforms / compilers as Xerces-C++, see FAQ: What platforms / compilers are being used to build the binary distribution kits? and the documentation in Alphaworks [30] .

Which DLL's do I need to distribute with my application?

As mentioned above, there are two configurations in which Xerces-C binaries are shipped. One is from the Apache site [31] , while the other is from IBM published at IBM's Alphaworks Site [30] .

If you are using the binaries from the Apache download site [32] site, then you only need to distribute **one** file:

xerces-c_2_5_0.dll for Windows NT/2000, or

libxerces-c25.0.so for AIX, or

libxerces-c.so.25.0 for Solaris/Linux, or

libxerces-c.sl.25.0 for HP-UX.

However, if you are using the XML4C binaries then in **addition** to the library file mentioned above, you also need to ship:

1. **ICU shared library file**:
   icuuc*.dll for Windows NT/2000, or
   libicuuc*.a for AIX, or
   libicuuc*.so for Solaris/Linux, or
   libicuuc*.sl for HP-UX.
2. **ICU converter data shared library file:**
   icudt*.dll for Windows NT/2000, or
   libicudt*.a for AIX, or
   libicudt*.so for Solaris/Linux, or
   libicudt*.sl for HP-UX.
3. **The Xerces-C++ Message file:**
   XercesMessages*.dll for Windows NT/2000, or
   libXercesMessages*.a for AIX, or
   libXercesMessages*.so for Solaris/Linux, or
   libXercesMessages*.sl for HP-UX.

How do I package the sources to create a binary drop?

You have to first compile the sources inside your IDE to create the required DLLs and EXEs. Then you need to copy over the binaries to another directory for the binary drop. A perl script has been provided to give you a jump start. You need to install perl on your machine for the script to work. If you have changed your source tree, you have to modify the script to suit your current directory structure. To invoke the script, go to the \<Xerces>\scripts directory, and type:

```
    perl packageBinaries.pl
```

You will get a message that somewhat looks like this (changes always happen, we are evolving you see!):

```
    Usage is: packageBinaries <options>
    options are:  -s <source_directory>
                  -o <target_directory>
                  -c <C compiler name> (e.g. gcc or xlc_r)
                  -x <C++ compiler name> (e.g. g++ or xlC_r)
                  -m <message loader> can be 'inmem', 'icu' or 'iconv'
                  -n <net accessor> can be 'fileonly' or 'libwww'
                  -t <transcoder> can be 'icu' or 'native'
                  -r <thread option> can be 'pthread' or 'dce' (only used on HP-11)
                  -h to get help on these commands
    Example: perl packageBinaries.pl -s$HOME/xerces-c-src2_5_0
                                     -o$HOME/xerces-c2_5_0
                                     -cgcc -xg++ -minmem
                                     -nfileonly -tnative
```

Make sure that your compiler can be invoked from the command line and follow the instructions to produce a binary drop.

I do not see binaries for my platform. When will they be available?

The reason why you see binaries only for some specific platforms is that we have had the maximum requests for them. Moreover, we have limited resources and hence cannot publish binaries for every platform. If you wish to contribute your time and effort in building binaries for a specific platform/environment then please send a mail to the Xerces-C++ mailing list [21] . We can definitely use any extra help in this open source project

When will a port to my platform be available?

We would like to see Xerces ported to as many platforms as there are. Again, due to limited resources we cannot do all the ports. We will help you make this port happen. Here are some Porting Guidelines.

We strongly encourage you to submit the changes that are required to make it work on another platform. We will incorporate these changes in the source code base and make them available in the future releases.

All porting changes may be sent to the Xerces-C++ mailing list [21] .

How can I port Xerces to my favourite platform?

Here are some Porting Guidelines.

What application do you use to create the documentation?

We have used an internal XML based application to create the documentation. The documentation files are all written in XML and the application, internally codenamed StyleBook, makes use of XSL to transform it into an HTML document that you are seeing right now. It is currently available on the Apache [33] open source website as Cocoon [34] .

The API documentation is generated using Doxygen [35] and GraphViz [36] .

See FAQ: Regenerating (API) documention?

Can I use Xerces in my product?

Yes! Read the license agreement first and if you still have further questions, then please address them to the Xerces-C++ mailing list [21] .

How do I uninstall Xerces-C++?

Xerces-C++ only installs itself in a single directory and does not set any registry entries. Thus, to uninstall, you only need to remove the directory where you installed it, and all Xerces-C++ related files will be removed.

I am getting a tar checksum error on Solaris. What's the problem?

The problem is caused by a limitation in the original tar spec, which prevented it from archiving files with long pathnames. Unfortunately, various current versions of tar use different extensions for eliminating this restriction which are incompatible with each other (or they do not remove the restriction at all). Rather than altering the pathnames for the Xerces-C++ package, which would make them compatible with the original tar spec but make it more difficult to know what was where, it was decided to use GNU tar (gtar), which handles arbitrarily long pathnames and is freely available on every platform on which Xerces-C++ is supported. If you don't already have GNU tar installed on your system, you can obtain it from the Free Software Foundation http://www.gnu.org/software/tar/tar.html [37] . For additional background information on this problem, see the online manual GNU tar and POSIX tar [38] for the utility.

Building / Running FAQs

Why do I get compilation error saying undeclared identifier or class undefined?

Xerces-C++ 2.5.0 now supports C++ Namespace.

If C++ Namespace is ENABLED, users' applications must namespace qualified all the Xerces-C++ classes/data/variables with `"xercesc::"` or add the `"using namespace xercesc;"` clause. Users also need to ensure all forward declarations are properly qualified or scope. For example

```
#include <stdio.h>
#include <stdlib.h>
#include <xercesc/sax/HandlerBase.hpp>

// indicate using Xerces-C++ namespace in general
using namespace xercesc;

// need to properly scope any forward declarations
namespace xercesc {
class AttributeList;
}

// or namespace qualifier the forward declarations
class xercesc::ErrorHandler;

class MySAXHandlers : public HandlerBase
{
public:
    // -----------------------------------------------------------------------
    //  Handlers for the SAX DocumentHandler interface
    // -----------------------------------------------------------------------
    void startElement(const XMLCh* const name, AttributeList& attributes);
    void characters(const XMLCh* const chars, const unsigned int length);
:
:
};
```

See the Programming Guide Using C++ Namespace for more details.

Why do I get compilation error saying DOMDocument was declared twice using Microsoft Visual C++.Net?

Your application somehow has picked up the Microsoft SDK header `Msxml.h` which has its own typedef of `DOMDocument`. This confuses with the Xerces-C++ 2.5.0 `xercesc::DOMDocument` and thus lead to the compilation errors.

Qualifier the use of DOMDocument in your application explicitly e.g.

`XERCES_CPP_NAMESPACE_QUALIFIER DOMDocument * fDoc;`

will eliminate these compilation problems.

Why can't the compiler find the include files?

A common cause for this problem is setting build environment variable XERCESROOT instead of setting variable XERCESCROOT. Note the C before ROOT.

Why do I get Internal Compiler Error when compiling Xerces-C++ for a 64bit target with gcc?

This is a compiler problem. Try turning off optimization to bypass the problem.

Why do I get compilation error when compiling Xerces-C++ on FreeBSD with native transcoder?

Please make sure you configure with "-t IconvFBSD" to use FreeBSD specific native transcoder.

Or you can use ICU transcoder (configure with -t icu) instead of the native transcoder.

Building Xerces-C++ with compiler GCC 2.7.x or 2.8.x gives problem, what's wrong?

Users using GCC 2.7.x or 2.8.x may have unsuccessful compile/link/run experience with Xerces-C++. There were issues related to templates and multi threaded exception handling with this old version GCC compiler.

Please upgrade to at least GCC 2.95.2.

Why does my application give unresolved linking errors?

Please check the following:

1. If you're using the binary build of Xerces-C++, make sure that the OS and compiler are the same version as the ones used to build the binary (please refer to FAQ: What platforms / compilers are being used to build the binary distribution kits?). Different OS and compiler versions might cause unresolved linking problems or compilation errors. If the versions are different, rebuild the Xerces-C++ library on your system before building your application. If you're using ICU (which is packaged with XML4C) you need to rebuild the compatible version of ICU first.
2. Check that the library path is set properly and that the correct versions of `gmake` and `autoconf` are on your system.
3. If C++ Namespace support is ENABLED (all the binary distributions of Xerces-C++ 2.5.0 are built with C++ Namespace enabled), users' applications must namespace qualified all the Xerces-C++ classes/data/variables with `"xercesc::"` or add the `"using namespace xercesc"` clause. See the Programming Guide Using C++ Namespace for more details.

Why do I get link error saying icudata library not found when building with ICU?

There is a bug in the Makefile of ICU 1.7, 1.8 and 1.8.1. The link created during ICU installation in $ICUROOT is, for example,

icudata.so@ - > icudt17l.so

instead of

libicudata.so@ - > libicudt17l.so

Therefore the -licudata doesn't work. To bypass the problem, please manually create the following link:

libicudata.so@ - > libicudt17l.so

This problem has been fixed in ICU 2.0.

I cannot run the sample applications. What is wrong?

In order to run an application built using Xerces you must set up your path and library search path properly. In the stand-alone version from Apache, you must have the Xerces-C++ runtime library available from your path settings.

On Windows this library is called `xerces-c_2_5_0.dll` which must be available from your `PATH` settings. (Note that there are separate debug and release dlls for Windows. The release dll is named `xerces-c_2_5_0.dll`, and the debug dll is named `xerces-c_2_5_0d.dll`).

On UNIX platforms the library is called libxerces-c.so.25.0 (or libxerces-c25.0.so or libxerces-c.sl.25.0) which must be available from your `LD_LIBRARY_PATH` (or `LIBPATH` or `SHLIB_PATH`) environment variable.

Thus, if you installed your binaries under `$HOME/fastxmlparser`, you need to point your library path to that directory.

```
    export LIBPATH=$LIBPATH:$HOME/fastxmlparser/lib # (AIX)
    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/fastxmlparser/lib # (Solaris,
  Linux)
    export SHLIB_PATH=$SHLIB_PATH:$HOME/fastxmlparser/lib # (HP-UX)
```

If you are using the XML4C parser from IBM, you will need to put in two additional DLLs. In the Windows build these are `icuuc*.dll` and `icudt*.dll` which must be available from your PATH settings. On UNIX, these libraries are called `libicuuc*.so` and `libicudt*.so` (or `.sl` for HP-UX or `.a` for AIX) which must be available from your library search path.

If the parser is built with icu message loader (as mentioned above), or message catalog loader, you need an environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

Why does my application crash on AIX when I run it under a multi-threaded environment?

AIX maintains two kinds of libraries on the system, thread-safe and non-thread safe. Multi-threaded libraries on AIX follow a different naming convention, Usually the multi-threaded library names are followed with "_r". For example, libc.a is single threaded whereas libc_r.a is multi-threaded.

To make your multi-threaded application run on AIX, you **must** ensure that you do not have a "system library path" in your `LIBPATH` environment variable when you run the application. The appropriate libraries (threaded or non-threaded) are automatically picked up at runtime. An application usually crashes when you build your application for multi-threaded operation but don't point to the thread-safe version of the system libraries. For example, LIBPATH can be simply set as:

```
    LIBPATH=$HOME/<Xerces>/lib
```

Where <Xerces> points to the directory where the Xerces application resides.

If, for any reason unrelated to Xerces, you need to keep a "system library path" in your LIBPATH environment variable, you must make sure that you have placed the thread-safe path before you specify the normal system path. For example, you must place */lib/threads* before */lib* in your LIBPATH variable. That is to say your LIBPATH may look like this:

```
    export LIBPATH=$HOME/<Xerces>/lib:/usr/lib/threads:/usr/lib
```

Where /usr/lib is where your system libraries are.

Why does my multi-threaded application crash on Solaris 2.6?

The problem appears because the throw call on Solaris 2.6 is not multi-thread safe. Sun Microsystems provides a patch to solve this problem. To get the latest patch for solving this problem, go to SunSolve.sun.com [39] and get the appropriate patch for your operating system. For Intel machines running Solaris, you need to get Patch ID 104678. For SPARC machines you need to get Patch ID #105591.

I just built my own application using the Xerces-C++ parser. Why does it crash?

In order to work with the Xerces-C++ parser, you have to first initialize the XML subsystem. The most common mistake is to forget this initialization. Before you make any calls to Xerces-C++ APIs, you must call XMLPlatformUtils::Initialize():

```
    try {
       XMLPlatformUtils::Initialize();
    }
    catch (const XMLException& toCatch) {
       // Do your failure processing here
    }
```

This initializes the Xerces system and sets its internal variables. Note that you must the include `xercesc/util/PlatformUtils.hpp` file for this to work.

Why does deleting a transcoded string result in assertion on windows?

Both your application program and the Xerces-C++ DLL must use the same *DLL* version of the runtime library. If either statically links to the runtime library, the problem will still occur.

For example, for a Win32/VC6 build, the runtime library build setting MUST be "Multithreaded DLL" for release builds and "Debug Multithreaded DLL" for debug builds.

Or for example for a Win32/BCB6 build, application need to switch to Multithreaded runtime to avoid such memory access violation.

To bypass such problem, instead of calling operator delete[] directly, you can use the provided function XMLString::release to delete any string that was allocated by the parser. This will ensure the string is allocated and deleted by the same DLL and such assertion problem should be resolved.

The libs/dll's I downloaded keep me from using the debugger in VC6.0. I am using the 'D', debug versions of them. "no symbolic information found" is what it says. Do I have to compile everything from source to make it work?

Unless you have the .pdb files, all you are getting with the debug library is that it uses the debug heap manager, so that you can compile your stuff in debug mode and not be dangerous. If you want full symbolic info for the Xerces-C++ library, you'll need the .pdb files, and to get those, you'll need to rebuild the Xerces-C++ library.

"First-chance exception in DOMPrint.exe (KERNEL32.DLL): 0xE06D7363: Microsoft C++ Exception." I am always getting this message when I am using the parser. My programs are terminating abnormally. Even the samples are giving this exception. I am using Visual C++ 6.0 with latest service pack installed.

Xerces-C++ uses C++ exceptions internally, as part of its normal operation. By default, the MSVC debugger will stop on each of these with the "First-chance exception ..." message.

To stop this from happening do this:

- start debugging (so the debug menu appears)
- from the debug menu select "Exceptions"
- from the box that opens select "Microsoft C++ Exception" and set it to "Stop if not handled" instead of "stop always".

You'll still land in the debugger if your program is terminating abnormally, but it will be at your problem, not from the internal Xerces-C++ exceptions.

"Fatal Error: Cannot open include file: XXX: No such file or directory"?

Due to the recent directory change, you may need to either update your project file, makefile, or your source/header file, for details, please refer to Directory Change.

"Cannot load message domain, Xerces Panic Error"?

If the parser is built with icu message loader (like IBM XML4C binaries), you need to make sure that the message library, (for exact name see FAQ: Which DLL's do I need to distribute with my application?) is located in a directory which is on the library search path. Or the message resource file, XercesMessages_en_US.res, is in the directory given at the call to XMLPlatformUtils::Initialize(), or is located in the directory pointed to by the environment variable XERCESC_NLS_HOME, or at $XERCESCROOT/msg.

If the parser is built with message catalog loader, you need to make sure that the message catalog file, XercesMessages_en_US.cat, is in the directory given at the call to XMLPlatformUtils::Initialize(), or is located in the directory pointed to by the environment variable XERCESC_NLS_HOME, or at $XERCESCROOT/msg.

"Why my document is valid on some platform while invalid on others"?

The parser relies on the system call, strtod(), to parse a string representation of a double/float data. In the case of no invalid characters found, the strtod() returns a double/float value if it is representable on that platform, or raises ERANGE to indicate either underflow or underflow occurs. And the parser assigns zero to the said data if underflow is found.

The threshold, where the strtod() decides if an underflow occurs, varies on platforms. On windows, it is roughly the order of e-308, on Linux, e-325, and on AIX, HP and Solaris, e-324.

So in an instance document, a data of value 1.0e-310 from a type with minExclusive 0, is considered invalid on windows (since it is converted to 0 and therefore violates the minExclusive constraint), but valid on other unix platforms (since it remains the original value).

The discussion above applies to data in xsd file as well.

How do I regenerate the (API) documentation?

To use the internal XML based application that creates the documentation, you must have a Java Virtual machine installed on your system. The application itself, written in Java, is part of Xerces.

To regenerate the documentation, go to directory `$XERCESCROOT` and start `createDocs.sh` (for Unix) or `createdocs.bat` (for Windows). The result can be found in directory `$XERCESCROOT/doc/html`.

To regenerate the API documentation, you need to have at least Doxygen [35] installed on your system.

If you want the API documentation to contain dependency graphs, you also need to have GraphViz [36] installed on your system. (Note that some people object to the GraphViz license [40] , so there are Linux distributions that do not include it.)

If you do not have GraphViz, or do not want to use it, you have to edit file `$XERCESCROOT/doc/Doxyfile` and change `HAVE_DOT = YES` into `HAVE_DOT = NO`.

To actually regenerate the API documentation, go to directory `$XERCESCROOT/doc/` and start Doxygen. The result can be found in directory `$XERCESCROOT/doc/html/apiDocs`.

Programming/Parsing FAQs

Does Xerces-C++ support Schema?

Yes. The Xerces-C++ 2.5.0 contains an implementation of the W3C XML Schema Language, a recommendation of the Worldwide Web Consortium available in three parts: XML Schema: Primer [41] and XML Schema: Structures [42] and XML Schema: Datatypes [43] . We consider this implementation complete. See the Schema page for limitations.

Why Xerces-C++ does not support this particular Schema feature?

The Xerces-C++ 2.5.0 contains an implementation of the W3C XML Schema Language, a recommendation of the Worldwide Web Consortium available in three parts: XML Schema: Primer [41] and XML Schema: Structures [42] and XML Schema: Datatypes [43] . We consider this implementation complete. See the Schema page for limitations.

If you find any Schema feature which is specified in the W3C XML Schema Language Recommendation does not work with Xerces-C++ 2.5.0, we encourage the submission of bugs as described in Bug Reporting page.

Why does my application crash when instantiating the parser?

In order to work with the Xerces-C++ parser, you have to first initialize the XML subsystem. The most common mistake is to forget this initialization. Before you make any calls to Xerces-C++ APIs, you must call XMLPlatformUtils::Initialize():

```
try {
   XMLPlatformUtils::Initialize();
}
catch (const XMLException& toCatch) {
   // Do your failure processing here
}
```

This initializes the Xerces system and sets its internal variables. Note that you must the include `xercesc/util/PlatformUtils.hpp` file for this to work.

Is it OK to call the XMLPlatformUtils::Initialize/Terminate pair of routines multiple times in one program?

Yes. Since Xerces-C++ 1.5.2, the code has been enhanced so that calling XMLPlatformUtils::Initialize/Terminate pair of routines multiple times in one process is now allowed.

But the application needs to guarantee that only one thread has entered either the method XMLPlatformUtils::Initialize() or the method XMLPlatformUtils::Terminate() at any one time.

If you are calling XMLPlatformUtils::Initialize() a number of times, and then follow with XMLPlatformUtils::Terminate() the same number of times, only the first XMLPlatformUtils::Initialize() will do the initialization, and only the last XMLPlatformUtils::Terminate() will clean up the memory. The other calls are ignored.

To ensure all the memory held by the parser are freed, the number of XMLPlatformUtils::Terminate() calls should match the number of XMLPlatformUtils::Initialize() calls.

Consider the following code snippets (for illustration simplicity the following sample code is not coded in try/catch clause):

```
// The XMLPlatformUtils::Initialize/Terminate calls are paired.
{
    // Initialize the parser
    XMLPlatformUtils::Initialize();

    SAXParser* parser = new SAXParser;
    parser->parse(xmlFile);
    delete parser;

    // Free all memory that was being held by the parser
    XMLPlatformUtils::Terminate();

    // Initialize the parser
```

```
        XMLPlatformUtils::Initialize();

        parser = new SAXParser;
        parser->parse(xmlFile);
        delete parser;

        // Free all memory that was being held by the parser
        XMLPlatformUtils::Terminate();
    }
```

```
    // calls XMLPlatformUtils::Initialize() three times
    // then calls XMLPlatformUtils::Terminate() numerous times
    {
        // Initialize the parser
        XMLPlatformUtils::Initialize();

        // The next two calls are no-op
        XMLPlatformUtils::Initialize();
        XMLPlatformUtils::Initialize();

        SAXParser* parser = new SAXParser;
        parser->parse(xmlFile);
        delete parser;

        // The first two XMLPlatformUtils::Terminate() calls are no-op
        XMLPlatformUtils::Terminate();
        XMLPlatformUtils::Terminate();

        // This third XMLPlatformUtils::Terminate() will free all memory that was
   being held by the parser
        XMLPlatformUtils::Terminate();

        // This extra fourth XMLPlatformUtils::Terminate() call is no-op.
        // However calling XMLPlatformUtils::Terminate() without a matching
   XMLPlatformUtils::Initialize()
        // is dangerous and should be avoided.
        XMLPlatformUtils::Terminate();
    }
```

Why does my application crash or hang if XMLPlatformUtils::Initialize()/Terminate() pair is called more than once?

Please make sure you are using the Xerces-C++ 1.5.2 or up.

Earlier version of Xerces-C++ does not allow XMLPlatformUtils::Initialize()/Terminate() pair to be called more than once or has a problem.

Why does my application crash after calling XMLPlatformUtils::Terminate()?

Please make sure the XMLPlatformUtils::Terminate() is the last Xerces-C++ function to be called in your program. NO explicit nor implicit Xerces-C++ destructor (those local data that are destructed when going out of scope) should be called after XMLPlatformUtils::Terminate().

For example consider the following code snippets which is incorrect (for illustration simplicity the following sample code is not coded in try/catch clause):

```
1: {
2:    XMLPlatformUtils::Initialize();
3:    DOMString c("hello");
4:    XMLPlatformUtils::Terminate();
5: }
```

The DOMString object "c" is destructed when going out of scope at line 5 before the closing brace. As a result, DOMString destructor is called at line 5 after XMLPlatformUtils::Terminate() which is wrong. Correct code should be:

```
1: {
2:    XMLPlatformUtils::Initialize();
2a:    {
3:           DOMString c("hello");
3a:    }
4:    XMLPlatformUtils::Terminate();
5: }
```

The extra pair of braces (line 2a and 3a) ensures that all implicit destructors are called before terminating Xerces-C++.

In addition the application also needs to guarantee that only one thread has entered either the method XMLPlatformUtils::Initialize() or the method XMLPlatformUtils::Terminate() at any one time.

I'm suddenly getting segfaults with Xerces-C 2.3.0; why might this be?

The introduction of pluggable memory management into Xerces-C, one of the main features of 2.3.0, means that application writers have to be more conscious about destructors being invoked implicitly after a call to XMLPlatformUtils::Terminate(). For example, the following code is guaranteed to produce a segmentation fault under Xerces-C 2.3.0, while it happened to work under previous versions (in fact, this was how our SAXPrint sample was formerly written; try-catch blocks removed for brevity):

```
void myParsingFunction()
{
    XMLPlatformUtils::Initialize();
    SAXParser parser;
    //parser.various method calls
    XMLPlatformUtils::Terminate();
} // seg fault here!
```

The reason this will produce a segmentation fault is that any dynamic memory the SAXParser (or any other of Xerces's parsers) needs to allocate is now allocated by default by a static object owned by XMLPlatformUtils. When the XMLPlatformUtils::Terminate() call is made, this object is destroyed--and, consequently, so are all the objects that it directly created. This includes all the objects dynamically allocated by the SAXParser. When the parser object goes out of scope, its destructor is invoked, and this attempts to destroy all the objects that it created--which have of course just been destroyed by the static MemoryManager in XMLPlatformUtils.

To avoid this, one must either explicitly scope the parser object inside calls to XMLPlatformUtils::Initialize() and XMLPlatformUtils::Terminate(), or dynamically allocate the parser object and destroy it explicitly before the call to XMLPlatformUtils::Terminate() is made.

Another way of producing segmentation faults--that again, unfortunately, was employed by some of our samples--is to have calls to XMLPlatformUtils::Terminate() in a catch block that catches any of Xerces's exceptions. Since the destructor of the exception will implicitly be invoked upon exit from the catch block, and since some of the exceptions' destructors call on Xerces's default memory manager to destroy dynamically-allocated objects, their destruction will provoke a segmentation fault even if a return statement is placed in the catch block since the default memory manager will no longer exist. This practice is now avoided in all our samples.

Is Xerces-C++ thread-safe?

This is not a question that has a simple yes/no answer. Here are the rules for using Xerces-C++ in a multi-threaded environment:

Within an address space, an instance of the parser may be used without restriction from a single thread, or an instance of the parser can be accessed from multiple threads, provided the application guarantees that only one thread has entered a method of the parser at any one time.

When two or more parser instances exist in a process, the instances can be used concurrently, without external synchronization. That is, in an application containing two parsers and two threads, one parser can be running within the first thread concurrently with the second parser running within the second thread.

The same rules apply to Xerces-C++ DOM documents. Multiple document instances may be concurrently accessed from different threads, but any given document instance can only be accessed by one thread at a time.

DOMStrings allow multiple concurrent readers. All DOMString const methods are thread safe, and can be concurrently entered by multiple threads. Non-const DOMString methods, such as `appendData()`, are not thread safe and the application must guarantee that no other methods (including const methods) are executed concurrently with them.

The application also needs to guarantee that only one thread has entered either the method XMLPlatformUtils::Initialize() or the method XMLPlatformUtils::Terminate() at any one time.

I am seeing memory leaks in Xerces-C++. Are they real?

The Xerces-C++ library allocates and caches some commonly reused items. The storage for these may be reported as memory leaks by some heap analysis tools; to avoid the problem, call the function `XMLPlatformUtils::Terminate()` before your application exits. This will free all memory that was being held by the library.

For most applications, the use of `Terminate()` is optional. The system will recover all memory when the application process shuts down. The exception to this is the use of Xerces-C++ from DLLs that will be repeatedly loaded and unloaded from within the same process. To avoid memory leaks with this kind of use, `Terminate()` must be called before unloading the Xerces-C++ library

To ensure all the memory held by the parser are freed, the number of XMLPlatformUtils::Terminate() calls should match the number of XMLPlatformUtils::Initialize() calls.

If you are using XML4C where ICU is used, you may call ICU function u_cleanup() to clean up ICU static data. Please see ICU documentation [22] for details.

I find memory leaks in Xerces-C++. How do I eliminate it?

The "leaks" that are reported through a leak-detector or heap-analysis tools aren't really leaks in most application, in that the memory usage does not grow over time as the XML parser is used and re-used.

What you are seeing as leaks are actually lazily evaluated data allocated into static variables. This data gets released when the application ends. You can make a call to `XMLPlatformUtil::terminate()` to release all the lazily allocated variables before you exit your program.

To ensure all the memory held by the parser are freed, the number of XMLPlatformUtils::Terminate() calls should match the number of XMLPlatformUtils::Initialize() calls.

If you are using XML4C where ICU is used, you may call ICU function u_cleanup() to clean up ICU static data. Please see ICU documentation [22] for details.

Is there a function that I have totally missed that creates an XML file from a DTD, (obviously with the values missing, a skeleton, as it were)?

No. This is not supported.

Can I use Xerces-C++ to perform "write validation" (which is having an appropriate Grammar and being able to add elements to the DOM whilst validating against the grammar)?

No. This is not supported.

The best you can do for now is to create the DOM document, write it back as XML and re-parse it.

Is there a facility in Xerces-C++ to validate the data contained in a DOM tree? That is, without saving and re-parsing the source document?

No. The best option for now is to generate XML source from the DOM and feed that back into the parser.

How to write out a DOM tree into a string or an XML file?

Please make sure you are using Xerces-C++ 2.5.0 or up.

You can use the DOMWriter::writeToString, or DOMWriter::writeNode to serialize a DOM tree. Please refer to the sample DOMPrint or the API documentation for more details of DOMWriter.

Why does DOMNode::cloneNode() not clone the pointer assigned to a DOMNode via DOMNode::setUserData()?

Xerces-C++ supports the DOMNode::userData specified in the DOM level 3 Node interface [44] . As is made clear in the description of the behaviour of `cloneNode()`, userData that has been set on the Node is not cloned. Thus, if the userData is to be copied to the new Node, this copy must be effected manually. Note further that the operation of `importNode()` is specified similarly.

How are entity reference nodes handled in DOM?

If you are using the native DOM classes, the function `setCreateEntityReferenceNodes` controls how entities appear in the DOM tree. When setCreateEntityReferenceNodes is set to true (the default), an occurrence of an entity reference in the XML document will be represented by a subtree with an EntityReference node at the root whose children represent the entity expansion. Entity expansion will be a DOM tree representing the structure of the entity expansion, not a text node containing the entity expansion as text.

If setCreateEntityReferenceNodes is false, an entity reference in the XML document is represented by only the nodes that represent the entity expansion. The DOM tree will not contain any entityReference nodes.

What kinds of URLs are currently supported in Xerces-C++?

The `XMLURL` class provides for limited URL support. It understands the `file://, http://`, and `ftp://` URL types, and is capable or parsing them into their constituent components, and normalizing them. It also supports the commonly required action of conglomerating a base and relative URL into a single URL. In other words, it performs the limited set of functions required by an XML parser.

Another thing that URLs commonly do are to create an input stream that provides access to the entity referenced. The parser, as shipped, only supports this functionality on URLs in the form `file:///` and `file://localhost/`, i.e. only when the URL refers to a local file.

You may enable support for HTTP and FTP URLs by implementing and installing a NetAccessor object. When a NetAccessor object is installed, the URL class will use it to create input streams for the remote entities referred to by such URLs.

How can I add support for URLs with HTTP/FTP protocols?

Support for the http: protocol is now included by default on all platforms.

To address the need to make remote connections to resources specified using additional protocols, ftp for example, Xerces-C++ provides the `NetAccessor` interface. The header file is `src/xercesc/util/XMLNetAccessor.hpp`. This interface allows you to plug in your own implementation of URL networking code into the Xerces-C++ parser.

Can I use Xerces-C++ to parse HTML?

Yes, but only if the HTML follows the rules given in the XML specification [2] . Most HTML, however, does not follow the XML rules, and will generate XML well-formedness errors.

I keep getting an error: "invalid UTF-8 character". What's wrong?

Most commonly, the XML `encoding =` declaration is either incorrect or missing. Without a declaration, XML defaults to the use utf-8 character encoding, which is not compatible with the default text file encoding on most systems.

The XML declaration should look something like this:

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

Make sure to specify the encoding that is actually used by file. The encoding for "plain" text files depends both on the operating system and the locale (country and language) in use.

Another common source of problems is that some characters are not allowed in XML documents, according to the XML spec. Typical disallowed characters are control characters, even if you escape them using the Character Reference form. See the XML spec [45] , sections 2.2 and 4.1 for details. If the parser is generating an `Invalid character (Unicode: 0x???)` error, it is very likely that there's a character in there that you can't see. You can generally use a UNIX command like "od -hc" to find it.

What encodings are supported by Xerces-C / XML4C?

Xerces-C has intrinsic support for ASCII, UTF-8, UTF-16 (Big/Small Endian), UCS4 (Big/Small Endian), EBCDIC code pages IBM037, IBM1047 and IBM1140 encodings, ISO-8859-1 (aka Latin1) and Windows-1252. This means that it can parse input XML files in these above mentioned encodings.

XML4C -- the version of Xerces-C available from IBM -- combines Xerces-C and International Components for Unicode (ICU) [22] and extends the encoding support to over 100 different encodings that are allowed by ICU. In particular, all the encodings registered with the Internet Assigned Numbers Authority (IANA) [46] are supported in XML4C.

Some implementations or ports of Xerces-C provide support for additional encodings. The exact set will depend on the supplier of the parser and on the character set transcoding services in use.

What character encoding should I use when creating XML documents?

The best choice in most cases is either utf-8 or utf-16. Advantages of these encodings include:

- The best portability. These encodings are more widely supported by XML processors than any others, meaning that your documents will have the best possible chance of being read correctly, no matter where they end up.

- Full international character support. Both utf-8 and utf-16 cover the full Unicode character set, which includes all of the characters from all major national, international and industry character sets.
- Efficient. utf-8 has the smaller storage requirements for documents that are primarily composed of characters from the Latin alphabet. utf-16 is more efficient for encoding Asian languages. But both encodings cover all languages without loss.

The only drawback of utf-8 or utf-16 is that they are not the native text file format for most systems, meaning that common text file editors and viewers can not be directly used.

A second choice of encoding would be any of the others listed in the table above. This works best when the xml encoding is the same as the default system encoding on the machine where the XML document is being prepared, because the document will then display correctly as a plain text file. For UNIX systems in countries speaking Western European languages, the encoding will usually be iso-8859-1.

The versions of Xerces distributed by IBM, both C and Java (known respectively as XML4C and XML4J), include all of the encodings listed in the above table, on all platforms.

A word of caution for Windows users: The default character set on Windows systems is windows-1252, not iso-8859-1. While Xerces-C++ does recognize this Windows encoding, it is a poor choice for portable XML data because it is not widely recognized by other XML processing tools. If you are using a Windows-based editing tool to generate XML, check which character set it generates, and make sure that the resulting XML specifies the correct name in the `encoding="..."` declaration.

Is EBCDIC supported?

Yes, Xerces-C++ supports EBCDIC with the ibm1140, ibm037 and ibm1047 encodings. When creating EBCDIC encoded XML data, the preferred encoding is ibm1140. The ibm037 encoding, and its alternate name, ebcdic-cp-us, is almost the same as ibm1140, but it lacks the Euro symbol.

These three encodings, ibm1140, ibm037 and ibm1047, are available on both Xerces-C and IBM XML4C, on all platforms.

On IBM System 390, XML4C also supports three alternative forms, ibm037-s390, ibm1140-s390, and ibm1047-s390. These are similar to the base ibm037, ibm1140, and ibm1047 encodings, but with alternate mappings of the EBCDIC new-line character, which allows them to appear as normal text files on System 390. These encodings are not supported on other platforms, and should not be used for portable data.

XML4C on System 390 and AS/400 also provides additional EBCDIC encodings, including those for the character sets of different countries. The exact set supported will be platform dependent, and these encodings are not recommended for portable XML data.

Why does deleting a transcoded string result in assertion on windows?

Both your application program and the Xerces-C++ DLL must use the same *DLL* version of the runtime library. If either statically links to the runtime library, the problem will still occur.

For example, for a Win32/VC6 build, the runtime library build setting MUST be "Multithreaded DLL" for release builds and "Debug Multithreaded DLL" for debug builds.

Or for example for a Win32/BCB6 build, application need to switch to Multithreaded runtime to avoid such memory access violation.

To bypass such problem, instead of calling operator delete[] directly, you can use the provided function XMLString::release to delete any string that was allocated by the parser. This will ensure the string is allocated and deleted by the same DLL and such assertion problem should be resolved.

How do I transcode to/from something besides the local code page?

XMLString::transcode() will transcode from XMLCh to the local code page, and other APIs which take a char* assume that the source text is in the local code page. If this is not true, you must transcode the text yourself. You can do this using local transcoding support on your OS, such as Iconv on Unix or IBM's ICU package. However, if your transcoding needs are simple, you can achieve some better portability by using the Xerces-C++ parser's transcoder wrappers. You get a transcoder like this:

- Call XMLPlatformUtils::fgTransServer- >MakeNewTranscoderFor() and provide the name of the encoding you wish to create a transcoder for. This will return a transcoder to you, which you own and must delete when you are through with it. NOTE: You must provide a maximum block size that you will pass to the transcoder at one time, and you must pass blocks of characters of this count or smaller when you do your transcoding. The reason for this is that this is really an internal API and is used by the parser itself to do transcoding. The parser always does transcoding in known block sizes, and this allows transcoders to be much more efficient for internal use since it knows the max size it will ever have to deal with and can set itself up for that internally. In general, you should stick to block sizes in the 4 to 64K range.
- The returned transcoder is something derived from XMLTranscoder, so they are all returned to you via that interface.
- This object is really just a wrapper around the underlying transcoding system actually in use by your version of Xerces, and does whatever is necessary to handle differences between the XMLCh representation and the representation used by that underlying transcoding system.
- The transcoder object has two primary APIs, transcodeFrom() and transcodeTo(). These transcode between the XMLCh format and the encoding you indicated.
- These APIs will transcode as much of the source data as will fit into the outgoing buffer you provide. They will tell you how much of the source they ate and how much of the target they filled. You can use this information to continue the process until all source is consumed.
- char* data is always dealt with in terms of bytes, and XMLCh data is always dealt with in terms of characters. Don't mix up which you are dealing with or you will not get the correct results, since many encodings don't have a one to one relationship of characters to bytes.
- When transcoding from XMLCh to the target encoding, the transcodeTo() method provides an 'unrepresentable flag' parameter, which tells the transcoder how to deal with an XMLCh code point that cannot be converted legally to the target encoding, which can easily happen since XMLCh is Unicode and can represent thousands of code points. The options are to use a default replacement character (which the underlying transcoding service will choose, and which is guaranteed to be legal for the target encoding), or to throw an exception.

Here is an example:

```
 // create an XMLTranscoder that is able to transcode between Unicode and Big5
 // ASSUMPTION: assumes your underlying transcoding utility supports this
encoding Big5
 XMLTranscoder* t =
makeNewTranscoderFor("Big5", failReason, 16*1024);

 // source string is in Unicode, wanna to transcode to Big5
 t->transcodeTo(source_unicode, length, result_Big5, length, charsEaten,
XMLTranscoder::UnRep_Throw );

 // source string in Big5, wanna to transcode to Unicode
 t->transcodeFrom(source_Big5, length, result_unicode, length, bytesEaten,
   (unsigned char*)charSz);
```

Why does setProperty not work?

The function `SAX2XMLReader::setProperty(const XMLCh* const name, void* value)` and `DOMBuilder::setProperty(const XMLCh* const name, void* value)` takes a void pointer for the property value. Application is required to initialize this void pointer to a correct type. See SAX2 Programming Guide and DOM Programming Guide to learn exactly what type of property value that each property expects for processing. Passing a void pointer that was initialized with a wrong type will lead to unexpected result.

Why does getProperty not work?

The function `void* SAX2XMLReader::getProperty(const XMLCh* const name)` and `void* DOMBuilder::getProperty(const XMLCh* const name)` returns a void pointer for the property value. See SAX2 Programming Guide and exactly what type of object each property returns.

The parser owns the returned pointer. The memory allocated for the returned pointer will be destroyed when the parser is deleted. To ensure accessibility of the returned information after the parser is deleted, callers need to copy and store the returned information somewhere else; otherwise you may get unexpected result. Since the returned pointer is a generic void pointer, see SAX2 Programming Guide and DOM Programming Guide to learn exactly what type of property value each property returns for replication.

Why does the parser still try to locate the DTD even validation is turned off and how to ignore external DTD reference?

When DTD is referenced, the parser will try to read it, because DTDs can provide a lot more information than just validation. It defines entities and notations, external unparsed entities, default attributes, character entities, etc... So it will always try to read it if present, even if validation is turned off.

To ignore the DTD, with Xerces-C++ 2.5.0 or up, you can call `setLoadExternalDTD(false)` (or `setFeature(XMLUni::fgXercesLoadExternalDTD, false)` to disable the loading of external DTD. The parser will then ignore any external DTD completely if the validationScheme is set to Val_Never.

Note: This flag is ignored if the validationScheme is set to Val_Always or Val_Auto.

To ignore the DTD in earlier version of Xerces-C++, the only way to get around this is to install an EntityResolver (see the Redirect sample for an example of how this is done), and reset the DTD file to "".

Why do I get segmentation fault when running on Redhat Linux?

There were some problems with Redhat Linux 7.x with C++ exception handling across shared libraries. More details can be found here [47] . Please try to upgrade your Redhat Linux gcc to the latest patch level and see if it helps.

Why does the XML data generated by the DOMWriter does not match my original XML input?

If you parse an xml document using XercesDOMParser or DOMBuilder and pass such DOMNode to DOMWriter for serialization, you may not get something that is exactly the same as the original XML data. The parser may have done normalization, end of line conversion, or has expanded the entity reference as per the XML 1.0 spec, 4.4 XML Processor Treatment of Entities and References. From DOMWriter perspective, it does not know what the original string was, all it sees is a processed DOMNode generated by the parser. But since the DOMWriter is supposed to generate something that is parsable if sent back to the parser, it will not print the DOMNode node value as is. The DOMWriter may do some "touch up" to the output data for it to be parsable.

See How does DOMWriter handle built-in entity Reference in node value? to understand further how DOMWriter touches up the entity reference.

Why does my application crash when deleting the parser after releasing a document?

In most cases, the parser handles deleting documents when the parser gets deleted. However, if an application needs to release a document, it shall adopt the document before releasing it, so that the parser knows that the ownership of this particular document is transfered to the application and will not try to delete it once the parser gets deleted.

```
XercesDOMParser *parser = new XercesDOMParser;
...
try
{
    parser->parse(gXmlFile);
}
catch ()
{
...
}
DOMNode *doc = parser->getDocument();
...
parser->adoptDocument();
doc->release();
...
delete parser;
```

The alternative to release document is to call parser's resetDocumentPool(), which releases all the documents parsed.

Why do we have two versions of XMLString::transcode (one with memory manager and one without)?

With the introdcution of the configurable memory manager, we didn't want to break users by changing the signature of the method. Also, we did not want to provide a default memory manager as it would introduce a side effect with users experiencing some strange core dumps. The latter will occur when the scope of the string allocated is beyond that of XMLPlatformUtils::Terminate (i.e. a string is allocated using the default memory manager which is deleted when XMLPlatformUtils::Terminate is called, but the allocated string is deleted later). We plan to deprecate the transcode method with no parameters in later releases.

Other Xerces-C++ Questions

Are the Xerces parsers Year-2000-compliant?

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

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

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

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

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

How do I determine the version of Xerces-C++ I am using?

The version string for Xerces-C++ is in one of the header files. Look inside the file `src/xercesc/util/XercesVersion.hpp` or, in the binary distribution, look in `include/xercesc/utils/XercesVersion.hpp`.

If you don't have the header files, you have to find the version information from the shared library name. On Windows right click on the DLL name xerces-c_2_5_0.dll in the bin directory and look up properties. The version information may be found on the Version tab.

On AIX, just look for the library name libxerces-c25.0.so (or libxerces-c.so.25.0 on Solaris/Linux and libxerces-c.sl.25.0 on HP-UX). The version number is indicated in the name of the library.

I can't use C++. Do you have a Java version?

Yes. The Xerces family of products also has a Java version. More information is available at: http://xml.apache.org/xerces2-j/index.html [48]

Where can I find additional information on XML?

The Web. http://www.oasis-open.org/cover/xml.html [49] is an excellent place to start, with links to overviews, FAQs, specifications, industry news, applications and other software, related standards, etc.

Is there any kind of support available for Xerces-C++?

Xerces-C++ comes with **no** formal support.

Every volunteer project obtains its strength from the people involved in it. Mailing lists provide a simple and effective communication mechanism. You are welcome to join any of these mailing lists (or all of them if you wish). You can choose to lurk, or to actively participate. It's up to you. Before you join these lists, you should look over the resources in the Reference Library section

Instructions for subscribing are at http://xml.apache.org/mail.html. Archives of the lists are available from http://marc.theaimsgroup.com/?l=xerces-c-dev [27]

I found a defect -- how do I report it?

See Bug Reporting.

I have a patch to the Xerces-C++ source code. How do I submit it?

Mail it to the Xerces-C++ mailing list [21] . There are no set rules about how or what must be included -- if you've fixed a problem or enhanced the code in some way, we really would like to get your changes, and will take them in any reasonable form.

Generally a diff of the changed files against the current sources from CVS is good, along with some kind of description of what the change is. (Working with the current sources is important!)

Where can I get predefined character entity definitions??

Download http://www.w3.org/TR/xhtml1/xhtml1.zip. [50]

Does Xerces-C++ support XPath?

No. The Xerces-C++ 2.5.0 only has partial XPath implementation for the purposes of handling Schema identity constraints. For full XPath support, you can refer Apache Xalan C++ [51] or other Open Source Projects like Pathan [52] .

11 Xerces-C++ Samples

Introduction

Xerces-C++ comes packaged with sample applications that demonstrate salient features of the parser using simple applications written on top of the SAX and DOM APIs provided by the parser. Sample XML data files are provided in the samples/data directory.

Building the Samples

For general information related to building--including platform-specific information--please refer to the Build Page. This information covers a standard installation where one has downloaded the entire Xerces-C binary distribution to a place on the filesystem to which one has write access. Below, we cover what to do if Xerces-C has been preinstalled on a system into a directory to which one does not have write access, but compiling the samples is desired.

In this situation, just do the following:

1. Copy the entire contents of the `samples` directory into a directory named samples that you have write-access to;
2. Set the `XERCESCOUT`environment variable to point to the directory that is the parent of the newly-created `samples` directory;
3. Proceed as normal to compile the samples (not forgetting to set `XERCESCROOT` to point to the place on the file system where Xerces-C's include files etc. are to be found);

Once this is done, the compiled samples will be placed in a directory named `bin`; this directory will be located in the same directory as the `samples` directory created in step 1 above.

Running the Samples

The sample applications are dependent on the Xerces-C++ shared library (and could also depend on the ICU library if you built Xerces-C++ with ICU). Therefore, on Windows platforms you must make sure that your `PATH` environment variable is set properly to pick up these shared libraries at runtime.

On UNIX platforms you must ensure that *LIBPATH* environment variable is set properly to pick up the shared libraries at runtime. (UNIX gurus will understand here that *LIBPATH* actually translates to **LD_LIBRARY_PATH** on Solaris and Linux, **SHLIB_PATH** on HP-UX, **DYLD_LIBRARY_PATH** on Mac OS X, and stays as **LIBPATH** on AIX).

To set you LIBPATH (on AIX for example), you would type:

```
    export LIBPATH=xerces-c2_5_0/lib:$LIBPATH
```

On both Windows and UNIX platforms, if the parser is built with icu message loader (like IBM XML4C binaries), or message catalog loader, then you need to set another environment variable, XERCESC_NLS_HOME to point to the directory, $XERCESCROOT/msg, where the message files reside.

```
set XERCESC_NLS_HOME=$XERCESCROOT\msg
or
export XERCESC_NLS_HOME=$XERCESCROOT/msg
or
setenv XERCESC_NLS_HOME=$XERCESCROOT/msg
```

Once you have set up the environment variables, you can run the samples by opening a command window (or your shell prompt for UNIX environments).

Xerces-C++ Samples

- SAXCount
  SAXCount counts the elements, attributes, spaces and characters in an XML file.
- SAXPrint
  SAXPrint parses an XML file and prints it out.
- DOMCount
  DOMCount counts the elements in a XML file.
- DOMPrint
  DOMPrint parses an XML file and prints it out.
- MemParse
  MemParse parses XML in a memory buffer, outputing the number of elements and attributes.
- Redirect
  Redirect redirects the input stream for external entities.
- PParse
  PParse demonstrates progressive parsing.
- StdInParse
  StdInParse demonstrates streaming XML data from standard input.
- EnumVal
  EnumVal shows how to enumerate the markup decls in a DTD Grammar.
- SEnumVal
  SEnumVal shows how to enumerate the markup decls in a Schema Grammar.
- CreateDOMDocument
  CreateDOMDocument creates a DOM tree in memory from scratch.
- SAX2Count
  SAX2Count counts the elements, attributes, spaces and characters in an XML file.
- SAX2Print
  SAX2Print parses an XML file and prints it out.
- PSVIWriter
  PSVIWriter exposes the underlying PSVI of the parsed XML file.
- SCMPrint
  SCMPrint parses an XSD file and prints information about the Schema Component Model.

12
Sample: SAXCount

SAXCount

SAXCount is the simplest application that counts the elements and characters of a given XML file using the (event based) SAX API.

Running SAXCount

The SAXCount sample parses an XML file and prints out a count of the number of elements in the file. To run SAXCount, enter the following

```
    SAXCount <XML File>
```

The following parameters may be set from the command line

```
    Usage:
        SAXCount [options] <XML file | List file>

    This program invokes the SAX Parser, and then prints the
    number of elements, attributes, spaces and characters found
    in each XML file, using SAX API.

    Options:
        -l          Indicate the input file is a List File that has a list of xml
   files.
                    Default to off (Input file is an XML file).
        -v=xxx      Validation scheme [always | never | auto*].
        -n          Enable namespace processing. Defaults to off.
        -s          Enable schema processing. Defaults to off.
        -f          Enable full schema constraint checking. Defaults to off.
        -locale=ll_CC specify the locale, default: en_US
        -?          Show this help.

      * = Default if not provided explicitly.
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from SAXCount

```
    cd xerces-c2_5_0-linux/samples/data
    SAXCount -v=always personal.xml
    personal.xml: 60 ms (37 elems, 12 attrs, 134 spaces, 134 chars)
```

Running SAXCount with the validating parser gives a different result because ignorable white-space is counted separately from regular characters.

```
    SAXCount -v=never personal.xml
    personal.xml: 10 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

Note that the sum of spaces and characters in both versions is the same.

Note: *The time reported by the program may be different depending on your machine processor.*

13 Sample: SAXPrint

SAXPrint

SAXPrint uses the SAX APIs to parse an XML file and print it back. Do note that the output of this sample is not exactly the same as the input (in terms of whitespaces, first line), but the output has the same information content as the input.

Running SAXPrint

The SAXPrint sample parses an XML file and prints out the contents again in XML (some loss occurs). To run SAXPrint, enter the following

```
SAXPrint <XML file>
```

The following parameters may be set from the command line

```
Usage:
    SAXPrint [options] <XML file>

This program invokes the SAX Parser, and then prints the
data returned by the various SAX handlers for the specified
XML file.

Options:
    -u=xxx      Handle unrepresentable chars [fail | rep | ref*].
    -v=xxx      Validation scheme [always | never | auto*].
    -n          Enable namespace processing.
    -s          Enable schema processing.
    -f          Enable full schema constraint checking.
    -x=XXX      Use a particular encoding for output (LATIN1*).
    -?          Show this help.

  * = Default if not provided explicitly.

The parser has intrinsic support for the following encodings:
    UTF-8, USASCII, ISO8859-1, UTF-16[BL]E, UCS-4[BL]E,
    WINDOWS-1252, IBM1140, IBM037, IBM1047.
```

-u=fail will fail when unrepresentable characters are encountered

-u=rep will replace with the substitution character for that codepage

-u=ref will report the character as a reference

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from SAXPrint

```
    cd xerces-c2_5_0-linux/samples/data
    SAXPrint -v=always personal.xml

    <?xml version="1.0" encoding="LATIN1"?>
    <personnel>

      <person id="Big.Boss">
        <name><family>Boss</family> <given>Big</given></name>
        <email>chief@foo.com</email>
        <link subordinates="one.worker two.worker three.worker
                               four.worker five.worker"></link>
      </person>

      <person id="one.worker">
        <name><family>Worker</family> <given>One</given></name>
        <email>one@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="two.worker">
        <name><family>Worker</family> <given>Two</given></name>
        <email>two@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="three.worker">
        <name><family>Worker</family> <given>Three</given></name>
        <email>three@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="four.worker">
        <name><family>Worker</family> <given>Four</given></name>
        <email>four@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="five.worker">
        <name><family>Worker</family> <given>Five</given></name>
        <email>five@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

    </personnel>
```

***Note:** SAXPrint does not reproduce the original XML file. SAXPrint and DOMPrint produce different results because of the way the two APIs store data and capture events.*

14 Sample: DOMCount

DOMCount

DOMCount uses the provided DOM API to parse an XML file, constructs the DOM tree and walks through the tree counting the elements (using just one API call).

Running DOMCount

The DOMCount sample parses an XML file and prints out a count of the number of elements in the file. To run DOMCount, enter the following

```
    DOMCount <XML file>
```

The following parameters may be set from the command line

```
    Usage:
        DOMCount [options] <XML file | List file>

    This program invokes the DOM parser, builds the DOM tree,
    and then prints the number of elements found in each XML file.

    Options:
        -l          Indicate the input file is a List File that has a list of xml
   files.
                    Default to off (Input file is an XML file).
        -v=xxx      Validation scheme [always | never | auto*].
        -n          Enable namespace processing. Defaults to off.
        -s          Enable schema processing. Defaults to off.
        -f          Enable full schema constraint checking. Defaults to off.
        -locale=ll_CC specify the locale, default: en_US
        -p          Print out names of elements and attributes encountered.
        -?          Show this help.

      * = Default if not provided explicitly.
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from DOMCount

```
    cd xerces-c2_5_0-linux/samples/data
    DOMCount -v=always personal.xml
    personal.xml: 20 ms (37 elems)
```

Note: *The time reported by the system may be different, depending on your processor type.*

15
Sample: DOMPrint

DOMPrint

DOMPrint parses an XML file, constructs the DOM tree, and invokes DOMWriter::writeNode() to serialize the resultant DOM tree back to XML stream.

Running DOMPrint

The DOMPrint sample parses an XML file, using either a validating or non-validating DOM parser configuration, builds a DOM tree, and then invokes DOMWriter::writeNode() to serialize the resultant DOM tree. To run DOMPrint, enter the following:

```
    DOMPrint <XML file>
```

The following parameters may be set from the command line

```

    Usage:
     DOMPrint [options] <XML file>

    This program invokes the DOM parser, and builds the DOM tree
    It then asks the DOMWriter to serialize the DOM tree.

    Options:
        -e          create entity reference nodes. Default is no expansion.
        -v=xxx      Validation scheme [always | never | auto*].
        -n          Enable namespace processing. Default is off.
        -s          Enable schema processing. Default is off.
        -f          Enable full schema constraint checking. Defaults is off.
        -wenc=XXX   Use a particular encoding for output. Default is
                    the same encoding as the input XML file. UTF-8 if
                    input XML file has not XML declaration.
        -wfile=xxx  Write to a file instead of stdout.
        -wscs=xxx   Enable/Disable split-cdata-sections.      Default on.
        -wddc=xxx   Enable/Disable discard-default-content.   Default on.
        -wflt=xxx   Enable/Disable filtering.                 Default off.
        -wfpp=xxx   Enable/Disable format-pretty-print.       Default off.
        -wbom=xxx   Enable/Disable write Byte-Order-Mark      Default off.
        -?          Show this help
      * = Default if not provided explicitly.

    The parser has intrinsic support for the following encodings:\n"
```

```
        UTF-8, USASCII, ISO8859-1, UTF-16[BL]E, UCS-4[BL]E,\n"
        WINDOWS-1252, IBM1140, IBM037, IBM1047.\n"
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from DOMPrint

```
    cd xerces-c2_5_0-linux/samples/data
    DOMPrint -v=always personal.xml

    <?xml version="1.0" encoding="iso-8859-1"?>

    <!DOCTYPE personnel SYSTEM "personal.dtd">
    <!-- @version: -->
    <personnel>

    <person id="Big.Boss">
      <name><family>Boss</family> <given>Big</given></name>
      <email>chief@foo.com</email>
      <link subordinates="one.worker two.worker three.worker
                             four.worker five.worker"></link>
    </person>

    <person id="one.worker">
      <name><family>Worker</family> <given>One</given></name>
      <email>one@foo.com</email>
      <link manager="Big.Boss"></link>
    </person>

    <person id="two.worker">
      <name><family>Worker</family> <given>Two</given></name>
      <email>two@foo.com</email>
      <link manager="Big.Boss"></link>
    </person>

    <person id="three.worker">
      <name><family>Worker</family> <given>Three</given></name>
      <email>three@foo.com</email>
      <link manager="Big.Boss"></link>
    </person>

    <person id="four.worker">
      <name><family>Worker</family> <given>Four</given></name>
      <email>four@foo.com</email>
      <link manager="Big.Boss"></link>
    </person>

    <person id="five.worker">
```

```
      <name><family>Worker</family> <given>Five</given></name>
      <email>five@foo.com</email>
      <link manager="Big.Boss"></link>
    </person>

    </personnel>
```

Note that DOMPrint does not reproduce the original XML file. DOMPrint and SAXPrint produce different results because of the way the two APIs store data and capture events.

Application needs to provide its own implementation of DOMErrorHandler (in this sample, the DOMPrintErrorHandler), if it would like to receive notification from the serializer in the case any error occurs during the serialization.

Application needs to provide its own implementation of DOMWriterFilter (in this sample, the DOMPrintFilter), if it would like to filter out certain part of the DOM representation, but must be aware that thus may render the resultant XML stream invalid.

Application may choose any combination of characters as the end of line sequence to be used in the resultant XML stream, but must be aware that thus may render the resultant XML stream ill formed.

Application may choose a particular encoding name in which the output XML stream would be, but must be aware that if characters, unrepresentable in the encoding specified, appearing in markups, may force the serializer to terminate serialization prematurely, and thus no complete serialization would be done.

Application shall query the serializer first, before set any feature/mode(true, false), or be ready to catch exception if this feature/mode is not supported by the serializer.

Application needs to clean up the filter, error handler and format target objects created for the serialization.

16 Sample: MemParse

MemParse

MemParse uses the Validating SAX Parser to parse a memory buffer containing XML statements, and reports the number of elements and attributes found.

Running MemParse

This program uses the SAX Parser to parse a memory buffer containing XML statements, and reports the number of elements and attributes found.

The following parameters may be set from the command line

```
Usage:
    MemParse [options]

This program uses the SAX Parser to parse a memory buffer
containing XML statements, and reports the number of
elements and attributes found.

Options:
    -v=xxx      Validation scheme [always | never | auto*].
    -n          Enable namespace processing. Defaults to off.
    -s          Enable schema processing. Defaults to off.
    -f          Enable full schema constraint checking. Defaults to off.
    -?          Show this help.

  * = Default if not provided explicitly.
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from MemParse

```
MemParse -v=always
```

The output is the following:

```
Finished parsing the memory buffer containing the following XML statements:
```

```
<?xml version='1.0' encoding='ascii'?>
<!DOCTYPE company [
<!ELEMENT company     (product,category,developedAt)>
<!ELEMENT product     (#PCDATA)>
<!ELEMENT category    (#PCDATA)>
<!ATTLIST category idea CDATA #IMPLIED>
<!ELEMENT developedAt (#PCDATA)>
]>

<company>
  <product>XML4C</product>
  <category idea='great'>XML Parsing Tools</category>
  <developedAt>
    IBM Center for Java Technology, Silicon Valley, Cupertino, CA
  </developedAt>
</company>

Parsing took 10 ms (4 elements, 1 attributes, 16 spaces, 95 characters).
```

Running MemParse with the validating parser gives a different result because ignorable white-space is counted separately from regular characters.

```
MemParse -v=never
```

The output is the following:

```
Finished parsing the memory buffer containing the following XML statements:

<?xml version='1.0' encoding='ascii'?>
<!DOCTYPE company [
<!ELEMENT company     (product,category,developedAt)>
<!ELEMENT product     (#PCDATA)>
<!ELEMENT category    (#PCDATA)>
<!ATTLIST category idea CDATA #IMPLIED>
<!ELEMENT developedAt (#PCDATA)>
]>

<company>
  <product>XML4C</product>
  <category idea='great'>XML Parsing Tools</category>
  <developedAt>
    IBM Center for Java Technology, Silicon Valley, Cupertino, CA
  </developedAt>
</company>

Parsing took 10 ms (4 elements, 1 attributes, 0 spaces, 111 characters).
```

Note that the sum of spaces and characters in both versions is the same.

Note: *The time reported by the system may be different, depending on your processor type.*

17 Sample: Redirect

Redirect

Redirect uses the SAX EntityResolver handler to redirect the input stream for external entities. It installs an entity resolver, traps the call to the external DTD file and redirects it to another specific file which contains the actual DTD.

Running Redirect

This program illustrates how a XML application can use the SAX EntityResolver handler to redirect the input stream for external entities. It installs an entity resolver, traps the call to the external DTD file and redirects it to another specific file which contains the actual DTD.

The program then counts and reports the number of elements and attributes in the given XML file.

```
Redirect <XML file>
```

Redirect is invoked as follows:

```
cd xerces-c2_5_0-linux/samples/data
Redirect personal.xml
```

The output is the following:

```
cd xerces-c2_5_0-linux/samples/data
Redirect personal.xml
personal.xml: 30 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

External files required to run this sample are 'personal.xml', 'personal.dtd' and 'redirect.dtd', which are all present in the 'samples/data' directory. Make sure that you run redirect in the samples/data directory.

The 'resolveEntity' callback in this sample looks for an external entity with system id as 'personal.dtd'. When it is asked to resolve this particular external entity, it creates and returns a new InputSource for the file 'redirect.dtd'.

A real-world XML application can similarly do application specific processing when encountering external entities. For example, an application might want to redirect all references to entities outside of its domain to local cached copies.

Note: *The time reported by the program may be different depending on your machine processor.*

18 Sample: PParse

PParse

PParse demonstrates progressive parsing.

In this example, the programmer doesn't have to depend upon throwing an exception to terminate the parsing operation. Calling parseFirst() will cause the DTD to be parsed (both internal and external subsets) and any pre-content, i.e. everything up to but not including the root element. Subsequent calls to parseNext() will cause one more piece of markup to be parsed, and spit out from the core scanning code to the parser. You can quit the parse any time by just not calling parseNext() anymore and breaking out of the loop. When you call parseNext() and the end of the root element is the next piece of markup, the parser will continue on to the end of the file and return false, to let you know that the parse is done.

Running PParse

PParse parses an XML file and prints out a count of the number of elements in the file

```
    Usage:
        PParse [options] <XML file>

    This program demonstrates the progressive parse capabilities of
    the parser system. It allows you to do a scanFirst() call followed by
    a loop which calls scanNext(). You can drop out when you've found what
    ever it is you want. In our little test, our event handler looks for
    16 new elements then sets a flag to indicate its found what it wants.
    At that point, our progressive parse loop exits.

    Options:
          -v=xxx        - Validation scheme [always | never | auto*].
          -n            - Enable namespace processing [default is off].
          -s            - Enable schema processing [default is off].
          -f            - Enable full schema constraint checking [default is off].
          -?            - Show this help.

      * = Default if not provided explicitly.
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from PParse

```
    cd xerces-c2_5_0-linux/samples/data
    PParse -v=always personal.xml
    personal.xml: 60 ms (37 elems, 12 attrs, 134 spaces, 134 chars)
```

Running PParse with the validating parser gives a different result because ignorable white-space is counted separately from regular characters.

```
    PParse -v=never personal.xml
    personal.xml: 10 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

Note that the sum of spaces and characters in both versions is the same.

***Note:** The time reported by the program may be different depending on your machine processor.*

19
Sample: StdInParse

StdInParse

StdInParse demonstrates streaming XML data from standard input.

Running StdInParse

The StdInParse sample parses an XML file from standard input and prints out a count of the number of elements in the file. To run StdInParse, enter the following:

```
    StdInParse < <XML file>
```

The following parameters may be set from the command line

```

    Usage:
        StdInParse [options] < <XML file>

    This program demonstrates streaming XML data from standard
    input.  It then uses the SAX Parser, and prints the
    number of elements, attributes, spaces and characters found
    in the input, using SAX API.

    Options:
        -v=xxx      Validation scheme [always | never | auto*].
        -n          Enable namespace processing. Defaults to off.
        -s          Enable schema processing. Defaults to off.
        -f          Enable full schema constraint checking. Defaults to off.
        -?          Show this help.

      * = Default if not provided explicitly.

```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Make sure that you run StdInParse in the samples/data directory.

Here is a sample output from StdInParse:

```
    cd xerces-c2_5_0-linux/samples/data
    StdInParse -v=always < personal.xml
    stdin: 10 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

Running StdInParse with the validating parser gives a different result because ignorable white-space is counted separately from regular characters.

```
    StdInParse -v=never < personal.xml
    stdin: 10 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

Note that the sum of spaces and characters in both versions is the same.

> ***Note:*** *The time reported by the program may be different depending on your machine processor.*

20
Sample: EnumVal

EnumVal

EnumVal shows how to enumerate the markup decls in a DTD Grammar.

Running EnumVal

This program parses the specified XML file, then shows how to enumerate the contents of the DTD Grammar.

```
Usage:
    EnumVal <XML file>

This program parses the specified XML file, then shows how to
enumerate the contents of the DTD Grammar. Essentially,
shows how one can access the DTD information stored in internal
data structures.
```

Here is a sample output from EnumVal

```
cd xerces-c2_5_0-linux/samples/data
EnumVal personal.xml

ELEMENTS:
----------------------------
  Name: personnel
  Content Model: (person)+

  Name: person
  Content Model: (name,email*,url*,link?)
  Attributes:
    Name:id, Type: ID

  Name: name
  Content Model: (#PCDATA|family|given)*

  Name: email
  Content Model: (#PCDATA)*

  Name: url
```

```
Content Model: EMPTY
Attributes:
  Name:href, Type: CDATA

Name: link
Content Model: EMPTY
Attributes:
  Name:subordinates, Type: IDREF(S)
  Name:manager, Type: IDREF(S)

Name: family
Content Model: (#PCDATA)*

Name: given
Content Model: (#PCDATA)*
```

21
Sample: CreateDOMDocument

CreateDOMDocument

CreateDOMDocument, illustrates how you can create a DOM tree in memory from scratch. It then reports the elements in the tree that was just created.

Running CreateDOMDocument

The CreateDOMDocument sample illustrates how you can create a DOM tree in memory from scratch. To run CreateDOMDocument, enter the following

```
    CreateDOMDocument
```

Here is a sample output from CreateDOMDocument

```
    CreateDOMDocument
    The tree just created contains: 4 elements.
```

22
Sample: SAX2Count

SAX2Count

SAX2Count is the simplest application that counts the elements and characters of a given XML file using the (event based) SAX2 API.

Running SAX2Count

The SAX2Count sample parses an XML file and prints out a count of the number of elements in the file. To run SAX2Count, enter the following

```
    SAX2Count <XML File>
```

The following parameters may be set from the command line

```
    Usage:
        SAX2Count [options] <XML file | List file>

    This program invokes the SAX2XMLReader, and then prints the
    number of elements, attributes, spaces and characters found
    in each XML file, using SAX2 API.

    Options:
        -l          Indicate the input file is a List File that has a list of xml
   files.
                    Default to off (Input file is an XML file).
        -v=xxx      Validation scheme [always | never | auto*].
        -f          Enable full schema constraint checking processing. Defaults to
   off.
        -n          Disable namespace processing. Defaults to on.
                    NOTE: THIS IS OPPOSITE FROM OTHER SAMPLES.
        -s          Disable schema processing. Defaults to on.
                    NOTE: THIS IS OPPOSITE FROM OTHER SAMPLES.
        -locale=ll_CC specify the locale, default: en_US
        -?          Show this help.

      * = Default if not provided explicitly.
```

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from SAX2Count

```
    cd xerces-c2_5_0-linux/samples/data
    SAX2Count -v=always personal.xml
    personal.xml: 60 ms (37 elems, 12 attrs, 134 spaces, 134 chars)
```

Running SAX2Count with the validating parser gives a different result because ignorable white-space is counted separately from regular characters.

```
    SAX2Count -v=never personal.xml
    personal.xml: 10 ms (37 elems, 12 attrs, 0 spaces, 268 chars)
```

Note that the sum of spaces and characters in both versions is the same.

> ***Note:*** *The time reported by the program may be different depending on your machine processor.*

23 Sample: SAX2Print

SAX2Print

SAX2Print uses the SAX2 APIs to parse an XML file and print it back. Do note that the output of this sample is not exactly the same as the input (in terms of whitespaces, first line), but the output has the same information content as the input.

Running SAX2Print

The SAX2Print sample parses an XML file and prints out the contents again in XML (some loss occurs). To run SAX2Print, enter the following

```
SAX2Print <XML file>
```

The following parameters may be set from the command line

```

Usage:
    SAX2Print [options] <XML file>

This program invokes the SAX2XMLReader, and then prints the
data returned by the various SAX2 handlers for the specified
XML file.

Options:
    -u=xxx      Handle unrepresentable chars [fail | rep | ref*].
    -v=xxx      Validation scheme [always | never | auto*].
    -e          Expand Namespace Alias with URI's.
    -x=XXX      Use a particular encoding for output (LATIN1*).
    -f          Enable full schema constraint checking processing. Defaults to
off.
    -p          Enable namespace-prefixes feature. Defaults to off.\n"
    -n          Disable namespace processing. Defaults to on.\n"
                NOTE: THIS IS OPPOSITE FROM OTHER SAMPLES.\n"
    -s          Disable schema processing. Defaults to on.
                NOTE: THIS IS OPPOSITE FROM OTHER SAMPLES.
    -?          Show this help.

  * = Default if not provided explicitly.

The parser has intrinsic support for the following encodings:
    UTF-8, USASCII, ISO8859-1, UTF-16[BL]E, UCS-4[BL]E,
        WINDOWS-1252, IBM1140, IBM037, IBM1047.
```

-u=fail will fail when unrepresentable characters are encountered

-u=rep will replace with the substitution character for that codepage

-u=ref will report the character as a reference

-v=always will force validation

-v=never will not use any validation

-v=auto will validate if a DOCTYPE declaration or a schema declaration is present in the XML document

Here is a sample output from SAX2Print

```
    cd xerces-c2_5_0-linux/samples/data
    SAX2Print -v=always personal.xml

    <?xml version="1.0" encoding="LATIN1"?>
    <personnel>

      <person id="Big.Boss">
        <name><family>Boss</family> <given>Big</given></name>
        <email>chief@foo.com</email>
        <link subordinates="one.worker two.worker three.worker
                               four.worker five.worker"></link>
      </person>

      <person id="one.worker">
        <name><family>Worker</family> <given>One</given></name>
        <email>one@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="two.worker">
        <name><family>Worker</family> <given>Two</given></name>
        <email>two@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="three.worker">
        <name><family>Worker</family> <given>Three</given></name>
        <email>three@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="four.worker">
        <name><family>Worker</family> <given>Four</given></name>
        <email>four@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

      <person id="five.worker">
        <name><family>Worker</family> <given>Five</given></name>
        <email>five@foo.com</email>
        <link manager="Big.Boss"></link>
      </person>

    </personnel>
```

***Note:** SAX2Print does not reproduce the original XML file. SAX2Print and DOMPrint produce different results because of the way the two APIs store data and capture events.*

24
Sample: SEnumVal

SEnumVal

SEnumVal shows how to enumerate the markup decls in a Schema Grammar.

Running SEnumVal

This program parses the specified XML file, then shows how to enumerate the contents of the Schema Grammar.

```
Usage:
    SEnumVal <XML file>

This program parses a file, then shows how to enumerate the
contents of the Schema Grammar. Essentially, shows how one can
access the Schema information stored in internal data structures.
```

Here is a sample output from SEnumVal

```
cd xerces-c2_5_0-linux/samples/data
SEnumVal personal-schema.xml

Name:                   personnel
Model Type:             Children
Create Reason:  Declared
ContentType:    OneOrMore
Content Model:  (person)+
ComplexType:
        TypeName:       ,C0
        ContentType:    OneOrMore
--------------------------------------------
Name:                   person
Model Type:             Children
Create Reason:  Declared
ContentType:    Sequence
Content Model:  (name,email*,url*,link?)
ComplexType:
        TypeName:       ,C1
        ContentType:    Sequence
Attributes:
```

```
            Name:                   salary
            Type:                   CDATA
            Default Type:   #IMPLIED
            Base Datatype:          Decimal
    Facets:
            fractionDigits=0

            Name:                   id
            Type:                   ID
            Default Type:   #REQUIRED
            Base Datatype:          ID

            Name:                   contr
            Type:                   CDATA
            Default Type:   #DEFAULT
            Value:                  false
            Base Datatype:          string
    Enumeration:
                true
                false

            Name:                   note
            Type:                   CDATA
            Default Type:   #IMPLIED
            Base Datatype:          string

    --------------------------------------------
    Name:                   name
    Model Type:             Children
    Create Reason:  Declared
    ContentType:    All
    Content Model: All(family,given)
    ComplexType:
            TypeName:       ,C3
            ContentType:    All
    --------------------------------------------
    Name:                   family
    Model Type:             Simple
    Create Reason:  Declared
    Base Datatype:          string
    --------------------------------------------
    Name:                   given
    Model Type:             Simple
    Create Reason:  Declared
    Base Datatype:          string
    --------------------------------------------
    Name:                   email
    Model Type:             Simple
    Create Reason:  Declared
    Base Datatype:          string
    --------------------------------------------
    Name:                   url
    Model Type:             Empty
    Create Reason:  Declared
    Content Model:  EMPTY
    ComplexType:
            TypeName:       ,C4
    Attributes:
            Name:                   href
            Type:                   CDATA
            Default Type:   #DEFAULT
            Value:                  http://
            Base Datatype:          string

    --------------------------------------------
    Name:                   link
    Model Type:             Empty
    Create Reason:  Declared
    Content Model:  EMPTY
    ComplexType:
            TypeName:       ,C5
    Attributes:
            Name:                   subordinates
            Type:                   IDREFS
            Default Type:   #IMPLIED
            Base Datatype:          List

            Name:                   manager
            Type:                   IDREF
            Default Type:   #IMPLIED
            Base Datatype:          IDREF

    --------------------------------------------
```

25
Sample: PSVIWriter

PSVIWriter

PSVIWriter shows how to access the PSVI and Schema Component Model information for the parsed document.

Running PSVIWriter

This program parses the specified XML file, then exposes the PSVI and Schema Component Model information.

```
Usage:
    PSVIWriter [options] <XML file | List file>

This program invokes the SAX2XMLReaderImpl, and then exposes the
underlying PSVI of each parsed XML file, using SAX2 API.

Options:
    -f          Enable full schema constraint checking processing. Defaults to
off.
    -o=xxx      Output PSVI to file xxx (default is stdout)
    -e=xxx      Output errors to file xxx (default is stdout)
    -u=xxx      Handle unrepresentable chars [fail | rep | ref*].
    -x=XXX      Use a particular encoding for output (UTF8*).
    -l          Indicate the input file is a List File that has a list of xml
files.
                Default to off (Input file is an XML file).
    -?          Show this help.

  * = Default if not provided explicitly.
```

Here is some sample output from PSVWriter (as the output is verbose it has been truncated)

```
cd xerces-c2_5_0-linux/samples/data
PSVIWriter personal.xml

<document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns:psv="http://apache.org/xml/2001/PSVInfosetExtension"
             xmlns="http://www.w3.org/2001/05/XMLInfoset">
        <characterEncodingScheme>UTF8</characterEncodingScheme>
```

```
            <standalone xsi:nil="true"/>
            <version>1.0</version>
            <children>
                    <comment>
                            <content> @version:  </content>
                    </comment>

    ...
```

26
Sample: SCMPrint

SCMPrint

SCMPrint shows how to access the Schema Content Model information.

Running SCMPrint

This program parses the specified XSD file, then shows how to access the Schema Content Model information.

```
Usage:
    SCMPrint [options] <XSD file | List file>

This program loads XML Schema file(s), to show how one can
access the Schema Content Model information.

Options:
    -f     Enable full schema constraint checking processing. Defaults to off.
    -l     Indicate the input file is a List File that has a list of XSD files.
           Default to off (Input file is a XSD file).
    -?     Show this help.
```

Here is some sample output from SCMPrint (as the output is verbose it has been truncated)

```
cd xerces-c2_5_0-linux/samples/data
SCMPrint personal.xsd

********** Printing out information from Schema **********

Processing Namespace:
============================================

Name:                 personnel
Component Type: Element
Content Model
        Type: Complex
        Name: C0

--------------------------------------------
Name:                 person
```

```
Component Type: Element
Content Model
        Type: Complex
        Name: C1

--------------------------------------------
Name:                 name
Component Type: Element
Content Model
        Type: Complex
        Name: C2

--------------------------------------------
Name:                 family
Component Type: Element
Content Model
        Type: Complex
        Name: C3

--------------------------------------------
Name:                 given
Component Type: Element
Content Model
        Type: Complex
        Name: C4

--------------------------------------------
Name:                 email
Component Type: Element
Content Model
        Type: Simple
        Name: string

--------------------------------------------
Name:                 url
Component Type: Element
Content Model
        Type: Complex
        Name: C5

--------------------------------------------
Name:                 link
Component Type: Element
Content Model
        Type: Complex
        Name: C6

--------------------------------------------
Processing Namespace:   http://www.w3.org/2001/XMLSchema
============================================
no elements


Name:                 http://www.w3.org/2001/XMLSchema, anyType
Component Type: Type Definition
Category:         Complex
Base:                 anyType
Content Model: (* (wildcard))

...
```

27 API Documentation

API Docs for Xerces-C++

Xerces-C++ is packaged with the API documentation for SAX and DOM, the two most common programming interfaces for XML. The most common framework classes have also been documented.

Xerces-C++ SAX is an implementation of the SAX 1.0/2.0 [7] specification.

Xerces-C++ DOM is an implementation of the

- DOM Level 1 Specification [4] , a W3C Recommendation of October 1, 1998
- DOM Level 2 Core Specification [5] , a W3C Recommendation of November 13, 2000
- DOM Level 2 Traversal and Range Specification [6] , a W3C Recommendation of November 13, 2000
- Contains a partial implementation of the DOM Level 3.0 Core Specification [53] , and DOM Level 3.0 Abstract Schemas and Load and Save Specification [54] (W3C Working Draft of 09 April 2002). This implementation is experimental. See DOM Level 3 Support for details.

For a complete understanding of how the Xerces-C++ APIs work, we recommend you to read these documents.

See the **Xerces-C++ API documentation** for more details.

> ***Note:*** *The API documentation is automatically generated using Doxygen [35] and GraphViz [36] .*

28
Programming Guide

DOM Programming Guide

The DOM API is based on the Apache Recommended DOM C++ binding.

Read the DOM Programming Guide document or jump directly to:

- Design Objectives
- DOM Level 3 Support in Xerces-C++
- Using DOM API
    - Accessing API from application code
    - Class Names
    - Objects Management
    - Memory Management
    - String Type
- XercesDOMParser
    - Constructing a XercesDOMParser
    - Supported Features
    - Supported Properties
- DOMBuilder
    - Constructing a DOMBuilder
    - How to interchange DOMInputSource and SAX InputSource?
    - Supported Features
    - Supported Properties
- DOMWriter
    - Constructing a DOMWriter
    - How does DOMWriter handle built-in entity Reference in node value?
    - Supported Features
- Deprecated - Java-like DOM

SAX Programming Guide

Read the SAX Programming Guide document or jump directly to:

- Using the SAX API
- SAXParser
    - Constructing a SAXParser
    - Supported Features

SAX2 Programming Guide

Read the SAX2 Programming Guide document or jump directly to:

- Using the SAX2 API
- SAX2XMLReader
  - Constructing an XML Reader
  - Supported Features
  - Supported Properties

Other Features

Read the Xerces-C++ Programming Guide document or jump directly to:

- Version Macros
- Schema Support
- Progressive Parsing
- Preparsing Grammar and Grammar Caching
- Loadable Message Text
- Pluggable Transcoders
- Porting Guidelines
- Using C++ Namespace
- Specify Locale for Message Loader
- Specify Location for Message Loader
- Use Specific Scanner
- Pluggable Panic Handler
- Pluggable Memory Manager
- Managing Security Vulnerabilities

29
SAX1 Programming Guide

Using the SAX API

The SAX API for XML parsers was originally developed for Java. Please be aware that there is no standard SAX API for C++, and that use of the Xerces-C++ SAX API does not guarantee client code compatibility with other C++ XML parsers.

The SAX API presents a callback based API to the parser. An application that uses SAX provides an instance of a handler class to the parser. When the parser detects XML constructs, it calls the methods of the handler class, passing them information about the construct that was detected. The most commonly used handler classes are DocumentHandler which is called when XML constructs are recognized, and ErrorHandler which is called when an error occurs. The header files for the various SAX handler classes are in '<xerces-c2_5_0 >/include/xercesc/sax'

As a convenience, Xerces-C++ provides the class HandlerBase, which is a single class which is publicly derived from all the Handler classes. HandlerBase's default implementation of the handler callback methods is to do nothing. A convenient way to get started with Xerces-C++ is to derive your own handler class from HandlerBase and override just those methods in HandlerBase which you are interested in customizing. This simple example shows how to create a handler which will print element names, and print fatal error messages. The source code for the sample applications show additional examples of how to write handler classes.

This is the header file MySAXHandler.hpp:

```
#include <xercesc/sax/HandlerBase.hpp>

class MySAXHandler : public HandlerBase {
public:
    void startElement(const XMLCh* const, AttributeList&);
    void fatalError(const SAXParseException&);
};
```

This is the implementation file MySAXHandler.cpp:

```
#include "MySAXHandler.hpp"
#include <iostream.h>

MySAXHandler::MySAXHandler()
{
}

MySAXHandler::startElement(const XMLCh* const name,
                           AttributeList& attributes)
{
    char* message = XMLString::transcode(name);
    cout << "I saw element: "<< message << endl;
    XMLString::release(&message);
}

MySAXHandler::fatalError(const SAXParseException& exception)
{
    char* message = XMLString::transcode(exception.getMessage());
    cout << "Fatal Error: " << message
         << " at line: " << exception.getLineNumber()
         << endl;
}
```

The XMLCh and AttributeList types are supplied by Xerces-C++ and are documented in the include files. Examples of their usage appear in the source code to the sample applications.

SAXParser

Constructing a SAXParser

In order to use Xerces-C++ to parse XML files, you will need to create an instance of the SAXParser class. The example below shows the code you need in order to create an instance of SAXParser. The DocumentHandler and ErrorHandler instances required by the SAX API are provided using the HandlerBase class supplied with Xerces-C++.

```

        #include <xercesc/parsers/SAXParser.hpp>
        #include <xercesc/sax/HandlerBase.hpp>
        #include <xercesc/util/XMLString.hpp>

        int main (int argc, char* args[]) {

            try {
                XMLPlatformUtils::Initialize();
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Error during initialization! :\n"
                     << message << "\n";
                XMLString::release(&message);
                return 1;
            }

            char* xmlFile = "x1.xml";
            SAXParser* parser = new SAXParser();
            parser->setDoValidation(true);    // optional.
            parser->setDoNamespaces(true);    // optional

            DocumentHandler* docHandler = new HandlerBase();
            ErrorHandler* errHandler = (ErrorHandler*) docHandler;
            parser->setDocumentHandler(docHandler);
            parser->setErrorHandler(errHandler);

            try {
                parser->parse(xmlFile);
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (const SAXParseException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (...) {
                cout << "Unexpected Exception \n" ;
                return -1;
            }

            delete parser;
            delete docHandler;
            return 0;
        }
```

SAXParser Supported Features

The behavior of the SAXParser is dependant on the values of the following features. All of the features below are set using the "setter" methods (e.g. `setDoNamespaces`), and are queried using the corresponding "getter" methods (e.g. `getDoNamespaces`). The following only gives you a quick summary of supported features. Please refer to API Documentation for complete detail.

None of these features can be modified in the middle of a parse, or an exception will be thrown.

void setDoNamespaces(const bool)	
true:	Perform Namespace processing.
false:	Do not perform Namespace processing.
default:	false
note:	If the validation scheme is set to Val_Always or Val_Auto, then the document must contain a grammar that supports the use of namespaces.
see:	setValidationScheme

void setDoValidation(const bool) (deprecated) please use setValidationScheme .	
true:	Report all validation errors.

false:	Do not report validation errors.
default:	see the default of setValidationScheme .
see:	setValidationScheme

void setValidationScheme(const ValSchemes)	
Val_Auto:	The parser will report validation errors only if a grammar is specified.
Val_Always:	The parser will always report validation errors.
Val_Never:	Do not report validation errors.
default:	Val_Auto
note:	If set to Val_Always, the document must specify a grammar. If this feature is set to Val_Never and document specifies a grammar, that grammar might be parsed but no validation of the document contents will be performed.
see:	setLoadExternalDTD

void setDoSchema(const bool)	
true:	Enable the parser's schema support.
false:	Disable the parser's schema support.
default:	false
note	If set to true, namespace processing must also be turned on.
see:	setDoNamespaces

void setValidationSchemaFullChecking(const bool)	
true:	Enable full schema constraint checking, including checking which may be time-consuming or memory intensive. Currently, particle unique attribution constraint checking and particle derivation restriction checking are controlled by this option.
false:	Disable full schema constraint checking.
default:	false
note:	This feature checks the Schema grammar itself for additional errors that are time-consuming or memory intensive. It does **not** affect the level of checking performed on document instances that use Schema grammars.
see:	setDoSchema

void setLoadExternalDTD(const bool)	
true:	Load the External DTD.
false:	Ignore the external DTD completely.
default:	true
note	This feature is ignored and DTD is always loaded if the validation scheme is set to Val_Always or Val_Auto.
see:	setValidationScheme

void setExitOnFirstFatalError(const bool)	
true:	Stops parse on first fatal error.
false:	Attempt to continue parsing after a fatal error.
default:	true
note:	The behavior of the parser when this feature is set to false is **undetermined**! Therefore use this feature with extreme caution because the parser may get stuck in an infinite loop or worse.

void setValidationConstraintFatal(const bool)	
true:	The parser will treat validation error as fatal and will exit depends on the state of setExitOnFirstFatalError .
false:	The parser will report the error and continue processing.
default:	false
note:	Setting this true does not mean the validation error will be printed with the word "Fatal Error". It is still printed as "Error", but the parser will exit if setExitOnFirstFatalError is set to true.
see:	setExitOnFirstFatalError

void useCachedGrammarInParse(const bool)	
true:	Use cached grammar if it exists in the pool.
false:	Parse the schema grammar.
default:	false
note:	The getter function for this method is called isUsingCachedGrammarInParse.
note:	If the grammar caching option is enabled, this option is set to true automatically. Any setting to this option by the users is a no-op.
see:	cacheGrammarFromParse

void cacheGrammarFromParse(const bool)	

true:	Cache the grammar in the pool for re-use in subsequent parses.
false:	Do not cache the grammar in the pool
default:	false
note:	The getter function for this method is called isCachingGrammarFromParse
note:	If set to true, the useCachedGrammarInParse is also set to true automatically.
see:	useCachedGrammarInParse

void setStandardUriConformant(const bool)	
true:	Force standard uri conformance.
false:	Do not force standard uri conformance.
default:	false
note:	If set to true, malformed uri will be rejected and fatal error will be issued.

void setCalculateSrcOfs(const bool)	
true:	Enable src offset calculation.
false:	Disable src offset calculation.
default:	false
note:	If set to true, the user can inquire about the current src offset within the input source. Setting it to false (default) improves the performance.

void setExternalSchemaLocation(const XMLCh* const)	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. Similar situation happens to <import> element in schema documents. This property allows the user to specify a list of schemas to use. If the targetNamespace of a schema specified using this method matches the targetNamespace of a schema occurring in the instance document in schemaLocation attribute, or if the targetNamespace matches the namespace attribute of <import> element, the schema specified by the user using this property will be used (i.e., the schemaLocation attribute in the instance document or on the <import> element will be effectively ignored).

Value	The syntax is the same as for schemaLocation attributes in instance documents: e.g, "http://www.example.com file_name.xsd". The user can specify more than one XML Schema in the list.
Value Type	XMLCh*

void setExternalNoNamespaceSchemaLocation(const XMLCh* const)	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. This property allows the user to specify the no target namespace XML Schema Location externally. If specified, the instance document's noNamespaceSchemaLocation attribute will be effectively ignored.
Value	The syntax is the same as for the noNamespaceSchemaLocation attribute that may occur in an instance document: e.g."file_name.xsd".
Value Type	XMLCh*

void useScanner(const XMLCh* const)	
Description	This property allows the user to specify the name of the XMLScanner to use for scanning XML documents. If not specified, the default scanner "IGXMLScanner" is used.
Value	The recognized scanner names are: 1."WFXMLScanner" - scanner that performs well-formedness checking only. 2. "DGXMLScanner" - scanner that handles XML documents with DTD grammar information. 3. "SGXMLScanner" - scanner that handles XML documents with XML schema grammar information. 4. "IGXMLScanner" - scanner that handles XML documents with DTD or/and XML schema grammar information. Users can use the predefined constants defined in XMLUni directly (fgWFXMLScanner, fgDGXMLScanner, fgSGXMLScanner, or fgIGXMLScanner) or a string that matches the value of one of those constants.

Value Type	XMLCh*
note:	See Use Specific Scanner for more programming details.

setSecurityManager(Security Manager * const)	
Description	Certain valid XML and XML Schema constructs can force a processor to consume more system resources than an application may wish. In fact, certain features could be exploited by malicious document writers to produce a denial-of-service attack. This property allows applications to impose limits on the amount of resources the processor will consume while processing these constructs.
Value	An instance of the SecurityManager class (see `xercesc/util/SecurityManager`). This class's documentation describes the particular limits that may be set. Note that, when instantiated, default values for limits that should be appropriate in most settings are provided. The default implementation is not thread-safe; if thread-safety is required, the application should extend this class, overriding methods appropriately. The parser will not adopt the SecurityManager instance; the application is responsible for deleting it when it is finished with it. If no SecurityManager instance has been provided to the parser (the default) then processing strictly conforming to the relevant specifications will be performed.
Value Type	SecurityManager*

30 SAX2 Programming Guide

Using the SAX2 API

The SAX2 API for XML parsers was originally developed for Java. Please be aware that there is no standard SAX2 API for C++, and that use of the Xerces-C++ SAX2 API does not guarantee client code compatibility with other C++ XML parsers.

The SAX2 API presents a callback based API to the parser. An application that uses SAX2 provides an instance of a handler class to the parser. When the parser detects XML constructs, it calls the methods of the handler class, passing them information about the construct that was detected. The most commonly used handler classes are ContentHandler which is called when XML constructs are recognized, and ErrorHandler which is called when an error occurs. The header files for the various SAX2 handler classes are in '<xerces-c2_5_0 >/include/xercesc/sax2'

As a convenience, Xerces-C++ provides the class DefaultHandler, which is a single class which is publicly derived from all the Handler classes. DefaultHandler's default implementation of the handler callback methods is to do nothing. A convenient way to get started with Xerces-C++ is to derive your own handler class from DefaultHandler and override just those methods in HandlerBase which you are interested in customizing. This simple example shows how to create a handler which will print element names, and print fatal error messages. The source code for the sample applications show additional examples of how to write handler classes.

This is the header file MySAX2Handler.hpp:

```
#include <xercesc/sax2/DefaultHandler.hpp>

class MySAX2Handler : public DefaultHandler {
public:
    void startElement(
        const   XMLCh* const    uri,
        const   XMLCh* const    localname,
        const   XMLCh* const    qname,
        const   Attributes&     attrs
    );
    void fatalError(const SAXParseException&);
};
```

This is the implementation file MySAX2Handler.cpp:

```
#include "MySAX2Handler.hpp"
#include <iostream.h>
```

```
    MySAX2Handler::MySAX2Handler()
    {
    }

    MySAX2Handler::startElement(const   XMLCh* const    uri,
                                const   XMLCh* const    localname,
                                const   XMLCh* const    qname,
                                const   Attributes&     attrs)
    {
        char* message = XMLString::transcode(name);
        cout << "I saw element: "<< message << endl;
        XMLString::release(&message);
    }

    MySAX2Handler::fatalError(const SAXParseException& exception)
    {
        char* message = XMLString::transcode(exception.getMessage());
        cout << "Fatal Error: " << message
             << " at line: " << exception.getLineNumber()
             << endl;
    }
```

The XMLCh and Attributes types are supplied by Xerces-C++ and are documented in the include files. Examples of their usage appear in the source code to the sample applications.

SAX2XMLReader

Constructing an XML Reader

In order to use Xerces-C++ to parse XML files, you will need to create an instance of the SAX2XMLReader class. The example below shows the code you need in order to create an instance of SAX2XMLReader. The ContentHandler and ErrorHandler instances required by the SAX API are provided using the DefaultHandler class supplied with Xerces-C++.

```
        #include <xercesc/sax2/SAX2XMLReader.hpp>
        #include <xercesc/sax2/XMLReaderFactory.hpp>
        #include <xercesc/sax2/DefaultHandler.hpp>
        #include <xercesc/util/XMLString.hpp>

        int main (int argc, char* args[]) {

            try {
                XMLPlatformUtils::Initialize();
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Error during initialization! :\n"
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return 1;
            }
            char* xmlFile = "x1.xml";
            SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();
            parser->setFeature(XMLUni::fgSAX2CoreValidation, true)   // optional
            parser->setFeature(XMLUni::fgSAX2CoreNameSpaces, true)   // optional

            DefaultHandler* defaultHandler = new DefaultHandler();
            parser->setContentHandler(defaultHandler);
            parser->setErrorHandler(defaultHandler);

            try {
                parser->parse(xmlFile);
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (const SAXParseException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (...) {
                cout << "Unexpected Exception \n" ;
                return -1;
            }

            delete parser;
            delete defaultHandler;
            return 0;
        }
```

Supported Features in SAX2XMLReader

The behavior of the SAX2XMLReader is dependant on the values of the following features. All of the features below can be set using the function `SAX2XMLReader::setFeature(cons XMLCh* const, const bool)`. And can be queried using the function `bool SAX2XMLReader::getFeature(const XMLCh* const)`.

None of these features can be modified in the middle of a parse, or an exception will be thrown.

SAX2 Features

http://xml.org/sax/features/namespaces	
true:	Perform Namespace processing.
false:	Do not perform Namespace processing.
default:	true
XMLUni Predefined Constant:	fgSAX2CoreNameSpaces

note:	If the validation feature is set to true, then the document must contain a grammar that supports the use of namespaces.
see:	http://xml.org/sax/features/namespace-prefixes
see:	http://xml.org/sax/features/validation

http://xml.org/sax/features/namespace-prefixes	
true:	Report the original prefixed names and attributes used for Namespace declarations.
false:	Do not report attributes used for Namespace declarations, and optionally do not report original prefixed names.
default:	false
XMLUni Predefined Constant:	fgSAX2CoreNameSpacePrefixes

http://xml.org/sax/features/validation	
true:	Report all validation errors.
false:	Do not report validation errors.
default:	true
XMLUni Predefined Constant:	fgSAX2CoreValidation
note:	If this feature is set to true, the document must specify a grammar. If this feature is set to false and document specifies a grammar, that grammar might be parsed but no validation of the document contents will be performed.
see:	http://apache.org/xml/features/validation/dynamic
see:	http://apache.org/xml/features/nonvalidating/load-external-

Xerces Features

http://apache.org/xml/features/validation/dynamic	
true:	The parser will validate the document only if a grammar is specified. (http://xml.org/sax/features/validation must be true).
false:	Validation is determined by the state of the http://xml.org/sax/features/validation feature.
default:	false
XMLUni Predefined Constant:	fgXercesDynamic
see:	http://xml.org/sax/features/validation

http://apache.org/xml/features/validation/schema	
true:	Enable the parser's schema support.
false:	Disable the parser's schema support.
default:	true
XMLUni Predefined Constant:	fgXercesSchema
note	If set to true, namespace processing must also be turned on.

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

http://apache.org/xml/features/validation/schema-full-checking	
true:	Enable full schema constraint checking, including checking which may be time-consuming or memory intensive. Currently, particle unique attribution constraint checking and particle derivation restriction checking are controlled by this option.
false:	Disable full schema constraint checking.
default:	false
XMLUni Predefined Constant:	fgXercesSchemaFullChecking
note:	This feature checks the schema grammar itself for additional errors that are time-consuming or memory intensive. It does **not** affect the level of checking performed on document instances that use schema grammars.
see:	http://apache.org/xml/features/validation/schema

http://apache.org/xml/features/nonvalidating/load-external-dtd	
true:	Load the external DTD.
false:	Ignore the external DTD completely.
default:	true
XMLUni Predefined Constant:	fgXercesLoadExternalDTD
note	This feature is ignored and DTD is always loaded when validation is on.
see:	http://xml.org/sax/features/validation

http://apache.org/xml/features/continue-after-fatal-error	
true:	Attempt to continue parsing after a fatal error.
false:	Stops parse on first fatal error.
default:	false
XMLUni Predefined Constant:	fgXercesContinueAfterFatalError
note:	The behavior of the parser when this feature is set to true is **undetermined**! Therefore use this feature with extreme caution because the parser may get stuck in an infinite loop or worse.

http://apache.org/xml/features/validation-error-as-fatal	
true:	The parser will treat validation error as fatal and will exit depends on the state of http://apache.org/xml/features/continue-after-fatal-error .
false:	The parser will report the error and continue processing.
default:	false

XMLUni Predefined Constant:	fgXercesValidationErrorAsFatal
note:	Setting this true does not mean the validation error will be printed with the word "Fatal Error". It is still printed as "Error", but the parser will exit if http://apache.org/xml/features/continue-after-fatal-error is set to false.
see:	http://apache.org/xml/features/continue-after-fatal-error

http://apache.org/xml/features/validation/use-cachedGrammarInParse	
true:	Use cached grammar if it exists in the pool.
false:	Parse the schema grammar.
default:	false
XMLUni Predefined Constant:	fgXercesUseCachedGrammarInParse
note:	If http://apache.org/xml/features/validation/cache-grammarF is enabled, this feature is set to true automatically. Any setting to this feature by the users is a no-op.
see:	http://apache.org/xml/features/validation/cache-grammarF

http://apache.org/xml/features/validation/cache-grammarFromParse	
true:	Cache the grammar in the pool for re-use in subsequent parses.
false:	Do not cache the grammar in the pool
default:	false
XMLUni Predefined Constant:	fgXercesCacheGrammarFromParse
note:	If set to true, the http://apache.org/xml/features/validation/use-cachedGran is also set to true automatically.
see:	http://apache.org/xml/features/validation/use-cachedGran

http://apache.org/xml/features/standard-uri-conformant	
true:	Force standard uri conformance.
false:	Do not force standard uri conformance.
default:	false
XMLUni Predefined Constant:	fgXercesStandardUriConformant
note:	If set to true, malformed uri will be rejected and fatal error will be issued.

http://apache.org/xml/features/calculate-src-ofs	
true:	Enable src offset calculation.
false:	Disable src offset calculation.
default:	false
XMLUni Predefined Constant:	fgXercesCalculateSrcOfs

note:	If set to true, the user can inquire about the current src offset within the input source. Setting it to false (default) improves the performance.

Supported Properties in SAX2XMLReader

The behavior of the SAX2XMLReader is dependant on the values of the following properties. All of the properties below can be set using the function `SAX2XMLReader::setProperty(const XMLCh* const, void*)`. It takes a void pointer as the property value. Application is required to initialize this void pointer to a correct type. Please check the column "Value Type" below to learn exactly what type of property value each property expects for processing. Passing a void pointer that was initialized with a wrong type will lead to unexpected result. If the same property is set more than once, the last one takes effect.

Property values can be queried using the function `void* SAX2XMLReader::getProperty(const XMLCh* const)`. The parser owns the returned pointer, and the memory allocated for the returned pointer will be destroyed when the parser is deleted. To ensure accessibility of the returned information after the parser is deleted, callers need to copy and store the returned information somewhere else. Since the returned pointer is a generic void pointer, check the column "Value Type" below to learn exactly what type of object each property returns for replication.

None of these properties can be modified in the middle of a parse, or an exception will be thrown.

Xerces Properties

http://apache.org/xml/properties/schema/external-schemaLocation	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. Similar situation happens to <import> element in schema documents. This property allows the user to specify a list of schemas to use. If the targetNamespace of a schema specified using this method matches the targetNamespace of a schema occurring in the instance document in schemaLocation attribute, or if the targetNamespace matches the namespace attribute of <import> element, the schema specified by the user using this property will be used (i.e., the schemaLocation attribute in the instance document or on the <import> element will be effectively ignored).
Value	The syntax is the same as for schemaLocation attributes in instance documents: e.g, "http://www.example.com file_name.xsd". The user can specify more than one XML Schema in the list.

Value Type	XMLCh*
XMLUni Predefined Constant:	fgXercesSchemaExternalSchemaLocation

http://apache.org/xml/properties/schema/external-noNamespaceSchemaLocation	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. This property allows the user to specify the no target namespace XML Schema Location externally. If specified, the instance document's noNamespaceSchemaLocation attribute will be effectively ignored.
Value	The syntax is the same as for the noNamespaceSchemaLocation attribute that may occur in an instance document: e.g."file_name.xsd".
Value Type	XMLCh*
XMLUni Predefined Constant:	fgXercesSchemaExternalNoNameSpaceSchemaLocation

http://apache.org/xml/properties/scannerName	
Description	This property allows the user to specify the name of the XMLScanner to use for scanning XML documents. If not specified, the default scanner "IGXMLScanner" is used.
Value	The recognized scanner names are: 1."WFXMLScanner" - scanner that performs well-formedness checking only. 2. "DGXMLScanner" - scanner that handles XML documents with DTD grammar information. 3. "SGXMLScanner" - scanner that handles XML documents with XML schema grammar information. 4. "IGXMLScanner" - scanner that handles XML documents with DTD or/and XML schema grammar information. Users can use the predefined constants defined in XMLUni directly (fgWFXMLScanner, fgDGXMLScanner, fgSGXMLScanner, or fgIGXMLScanner) or a string that matches the value of one of those constants.
Value Type	XMLCh*
XMLUni Predefined Constant:	fgXercesScannerName
note:	See Use Specific Scanner for more programming details.

http://apache.org/xml/properties/security-manager	
Description	Certain valid XML and XML Schema constructs can force a processor to consume more system resources than an application may wish. In fact, certain features could be exploited by malicious document writers to produce a denial-of-service attack. This property allows applications to impose limits on the amount of resources the processor will consume while processing these constructs.
Value	An instance of the SecurityManager class (see `xercesc/util/SecurityManager`). This class's documentation describes the particular limits that may be set. Note that, when instantiated, default values for limits that should be appropriate in most settings are provided. The default implementation is not thread-safe; if thread-safety is required, the application should extend this class, overriding methods appropriately. The parser will not adopt the SecurityManager instance; the application is responsible for deleting it when it is finished with it. If no SecurityManager instance has been provided to the parser (the default) then processing strictly conforming to the relevant specifications will be performed.
Value Type	SecurityManager*
XMLUni Predefined Constant:	fgXercesSecurityManager

31 DOM Programming Guide

Design Objectives

The C++ DOM implementation is based on the Apache Recommended DOM C++ binding.

The design objective aims at meeting the following requirements:

- Reduced memory footprint.
- Fast - especially for use in server style and multi-threaded applications.
- Good scalability on multiprocessor systems.
- More C++ like and less Java like.

DOM Level 3 Support in Xerces-C++

The Xerces-C++ 2.5.0 contains a partial implementation of the W3C Document Object Model Level 3. This implementation is experimental. See the document DOM Level 3 Support for details.

Using DOM API

Accessing API from application code

```
    #include <xercesc/dom/DOM.hpp>
```

The header file <dom/DOM.hpp> includes all the individual headers for the DOM API classes.

Class Names

The DOM class names are prefixed with "DOM" (if not already), e.g. "DOMNode". The intent is to prevent conflicts between DOM class names and other names that may already be in use by an application or other libraries that a DOM based application must link with.

```
       DOMDocument*   myDocument;
       DOMNode*       aNode;
       DOMText*       someText;
```

Objects Management

Applications would use normal C++ pointers to directly access the implementation objects for Nodes in C++ DOM.

Consider the following code snippets

```
        DOMNode*       aNode;
        DOMNode* docRootNode;

        aNode = someDocument->createElement(anElementName);
        docRootNode = someDocument->getDocumentElement();
        docRootNode->appendChild(aNode);
```

Memory Management

The C++ DOM implementation provides a release() method for releasing any "orphaned" resources that were created through createXXXX factory method. Memory for any returned object are owned by implementation. Please see Apache Recommended DOM C++ binding for details.

Objects created by DOMImplementation::createXXXX

Users **must** call the release() function when finished using any objects that were created by the DOMImplementation::createXXXX (e.g. DOMBuilder, DOMWriter, DOMDocument, DOMDocumentType).

Acesss to a released object will lead to unexpected behaviour.

> ***Note:*** *When a DOMDocument is released, all its associated children AND any objects it owned (e.g. DOMRange, DOMTreeWalker, DOMNodeIterator or any orphaned nodes) will also be released.*
>
> ***Note:*** *When a DOMDocument is cloned, the cloned document has nothing related to the original master document and need to be released explicitly.*
>
> ***Note:*** *When a DOMDocumentType has been inserted into a DOMDocument and thus has a owner, it will then be released automatically when its owner document is released. DOMException::INVALID_ACCESS_ERR will be raised if releasing such owned node.*

Objects created by DOMDocument::createXXXX

Users **can** call the release() function to indicate the release of any orphaned nodes. When an orphaned Node is released, its associated children will also be released. Acesss to a released Node will lead to unexpected behaviour. These orphaned Nodes will eventually be released, if not already done so, when its owner document is released

> ***Note:*** *DOMException::INVALID_ACCESS_ERR will be raised if releasing a Node that has a parent (has a owner).*

Objects created by DOMDocumentRange::createRange or DOMDocumentTraversal::createXXXX

Users **can** call release() function when finished using the DOMRange, DOMNodeIterator, DOMTreeWalker. Acesss to a released object will lead to unexpected behaviour. These objects will eventually be released, if not already done so, when its owner document is released

Here is an example

```

         //
         //  Create a small document tree
         //

         {
          XMLCh* tempStr[100];

          XMLString::transcode("Range", tempStr, 99);
          DOMImplementation* impl =
DOMImplementationRegistry::getDOMImplementation(tempStr, 0);

          XMLString::transcode("root", tempStr, 99);
          DOMDocument*   doc = impl->createDocument(0, tempStr, 0);
          DOMElement*   root = doc->getDocumentElement();

          XMLString::transcode("FirstElement", tempStr, 99);
          DOMElement*   e1 = doc->createElement(tempStr);
          root->appendChild(e1);

          XMLString::transcode("SecondElement", tempStr, 99);
          DOMElement*   e2 = doc->createElement(tempStr);
          root->appendChild(e2);

          XMLString::transcode("aTextNode", tempStr, 99);
          DOMText*       textNode = doc->createTextNode(tempStr);
          e1->appendChild(textNode);

          // optionally, call release() to release the resource associated with
the range after done
          DOMRange* range = doc->createRange();
          range->release();

          // removedElement is an orphaned node, optionally call release() to
release associated resource
          DOMElement* removedElement = root->removeChild(e2);
          removedElement->release();

          // no need to release this returned object which is owned by
implementation
          XMLString::transcode("*", tempStr, 99);
          DOMNodeList*    nodeList = doc->getElementsByTagName(tempStr);

          // done with the document, must call release() to release the entire
document resources
          doc->release();
      };
```

String Type

The C++ DOM uses the plain, null-terminated (XMLCh *) utf-16 strings as the String type. The (XMLCh*) utf-16 type string has low overhead.

```
    //C++ DOM
    const XMLCh* nodeValue = aNode->getNodeValue();
```

All the string data would remain in memory until the document object is released. But such string data may be RECYCLED by the implementation if necessary. Users should make appropriate copy of any returned string for safe reference.

For example after a DOMNode has been released, the memory allocated for its node value will be recycled by the implementation.

```
    XMLCh xfoo[] = {chLatin_f, chLatin_o, chLatin_o, chNull};

    // pAttr has node value = "foo"
    // fNodeValue has "foo"
    pAttr->setNodeValue(xfoo);
    const XMLCh* fNodeValue = pAttr->getNodeValue();

    // fNodeValue has "foo"
    // make a copy of the string for future reference
    XMLCh* oldNodeValue = XMLString::replicate(fNodeValue);

    // release the node pAttr
    pAttr->release()

    // other operations
    :
    :

    // implementation may have recycled the memory of the pAttr already
    // so it's not safe to expect fNodeValue still have "foo"
    if (XMLString::compareString(xfoo, fNodeValue))
        printf("fNodeValue has some other content\n");

    // should use your own safe copy
    if (!XMLString::compareString(xfoo, oldNodeValue))
        printf("Use your own copy of the oldNodeValue if want to reference the
string later\n");

    // delete your own replicated string when done
    XMLString::release(&oldNodeValue);
```

Or if DOMNode::setNodeValue() is called to set a new node value, the implementation will simply overwrite the node value memory area. So any previous pointers will now have the new value automatically. Users should make appropriate copy of any previous returned string for safe reference. For example

```
    XMLCh xfoo[] = {chLatin_f, chLatin_o, chLatin_o, chNull};
    XMLCh xfee[] = {chLatin_f, chLatin_e, chLatin_e, chNull};

    // pAttr has node value = "foo"
    pAttr->setNodeValue(xfoo);
    const XMLCh* fNodeValue = pAttr->getNodeValue();
```

```
      // fNodeValue has "foo"
      // make a copy of the string for future reference
      XMLCh* oldNodeValue = XMLString::replicate(fNodeValue);

      // now set pAttr with a new node value "fee"
      pAttr->setNodeValue(xfee);

      // should not rely on fNodeValue for the old node value, it may not compare
      if (XMLString::compareString(xfoo, fNodeValue))
          printf("Should not rely on fNodeValue for the old node value\n");

      // should use your own safe copy
      if (!XMLString::compareString(xfoo, oldNodeValue))
          printf("Use your own copy of the oldNodeValue if want to reference the
  string later\n");

      // delete your own replicated string when done
      XMLString::release(&oldNodeValue);
```

This is to prevent memory growth when DOMNode::setNodeValue() is being called hundreds of times. This design allows users to actively select which returned string should stay in memory by manually copying the string to application's own heap.

XercesDOMParser

Constructing a XercesDOMParser

In order to use Xerces-C++ to parse XML files using DOM, you can create an instance of the XercesDOMParser class. The example below shows the code you need in order to create an instance of the XercesDOMParser.

```
       #include <xercesc/parsers/XercesDOMParser.hpp>
       #include <xercesc/dom/DOM.hpp>
       #include <xercesc/sax/HandlerBase.hpp>
       #include <xercesc/util/XMLString.hpp>
       #include <xercesc/util/PlatformUtils.hpp>

       int main (int argc, char* args[]) {

           try {
               XMLPlatformUtils::Initialize();
           }
           catch (const XMLException& toCatch) {
               char* message = XMLString::transcode(toCatch.getMessage());
               cout << "Error during initialization! :\n"
                    << message << "\n";
               XMLString::release(&message);
               return 1;
           }
```

```
            XercesDOMParser* parser = new XercesDOMParser();
            parser->setValidationScheme(XercesDOMParser::Val_Always);    //
   optional.
            parser->setDoNamespaces(true);    // optional

            ErrorHandler* errHandler = (ErrorHandler*) new HandlerBase();
            parser->setErrorHandler(errHandler);

            char* xmlFile = "x1.xml";

            try {
                parser->parse(xmlFile);
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (const DOMException& toCatch) {
                char* message = XMLString::transcode(toCatch.msg);
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (...) {
                cout << "Unexpected Exception \n" ;
                return -1;
            }

            delete parser;
            delete errHandler;
            return 0;
        }
```

XercesDOMParser Supported Features

The behavior of the XercesDOMParser is dependant on the values of the following features. All of the features below are set using the "setter" methods (e.g. `setDoNamespaces`), and are queried using the corresponding "getter" methods (e.g. `getDoNamespaces`). The following only gives you a quick summary of supported features. Please refer to API Documentation for complete detail.

void setCreateEntityReferenceNodes(const bool)	
true:	Create EntityReference nodes in the DOM tree. The EntityReference nodes and their child nodes will be read-only.

false:	Do not create EntityReference nodes in the DOM tree. No EntityReference nodes will be created, only the nodes corresponding to their fully expanded substitution text will be created.
default:	true
note:	This feature only affects the appearance of EntityReference nodes in the DOM tree. The document will always contain the entity reference child nodes.

void setExpandEntityReferences(const bool) (deprecated) please use setCreateEntityReferenceNodes	
true:	Do not create EntityReference nodes in the DOM tree. No EntityReference nodes will be created, only the nodes corresponding to their fully expanded sustitution text will be created.
false:	Create EntityReference nodes in the DOM tree. The EntityReference nodes and their child nodes will be read-only.
default:	false
see:	setCreateEntityReferenceNodes

void setIncludeIgnorableWhitespace(const bool)	
true:	Include text nodes that can be considered "ignorable whitespace" in the DOM tree.
false:	Do not include ignorable whitespace in the DOM tree.
default:	true
note:	The only way that the parser can determine if text is ignorable is by reading the associated grammar and having a content model for the document. When ignorable whitespace text nodes are included in the DOM tree, they will be flagged as ignorable; and the method DOMText::isIgnorableWhitespace() will return true for those text nodes.

void setDoNamespaces(const bool)	
true:	Perform Namespace processing.
false:	Do not perform Namespace processing.
default:	false
note:	If the validation scheme is set to Val_Always or Val_Auto, then the document must contain a grammar that supports the use of namespaces.

see:	setValidationScheme

void setDoValidation(const bool) (deprecated) please use setValidationScheme	
true:	Report all validation errors.
false:	Do not report validation errors.
default:	see the default of setValidationScheme
see:	setValidationScheme

void setValidationScheme(const ValSchemes)	
Val_Auto:	The parser will report validation errors only if a grammar is specified.
Val_Always:	The parser will always report validation errors.
Val_Never:	Do not report validation errors.
default:	Val_Auto
note:	If set to Val_Always, the document must specify a grammar. If this feature is set to Val_Never and document specifies a grammar, that grammar might be parsed but no validation of the document contents will be performed.
see:	setLoadExternalDTD

void setDoSchema(const bool)	
true:	Enable the parser's schema support.
false:	Disable the parser's schema support.
default:	false
note	If set to true, namespace processing must also be turned on.
see:	setDoNamespaces

void setValidationSchemaFullChecking(const bool)	
true:	Enable full schema constraint checking, including checking which may be time-consuming or memory intensive. Currently, particle unique attribution constraint checking and particle derivation restriction checking are controlled by this option.
false:	Disable full schema constraint checking.
default:	false

note:	This feature checks the Schema grammar itself for additional errors that are time-consuming or memory intensive. It does **not** affect the level of checking performed on document instances that use Schema grammars.
see:	setDoSchema

void setLoadExternalDTD(const bool)	
true:	Load the External DTD .
false:	Ignore the external DTD completely.
default:	true
note	This feature is ignored and DTD is always loaded if the validation scheme is set to Val_Always or Val_Auto.
see:	setValidationScheme

void setExitOnFirstFatalError(const bool)	
true:	Stops parse on first fatal error.
false:	Attempt to continue parsing after a fatal error.
default:	true
note:	The behavior of the parser when this feature is set to false is **undetermined**! Therefore use this feature with extreme caution because the parser may get stuck in an infinite loop or worse.

void setValidationConstraintFatal(const bool)	
true:	The parser will treat validation error as fatal and will exit depends on the state of setExitOnFirstFatalError
false:	The parser will report the error and continue processing.
default:	false
note:	Setting this true does not mean the validation error will be printed with the word "Fatal Error". It is still printed as "Error", but the parser will exit if setExitOnFirstFatalError is set to true.
see:	setExitOnFirstFatalError

void useCachedGrammarInParse(const bool)	
true:	Use cached grammar if it exists in the pool.
false:	Parse the schema grammar.
default:	false
note:	The getter function for this method is called isUsingCachedGrammarInParse.

note:	If the grammar caching option is enabled, this option is set to true automatically. Any setting to this option by the users is a no-op.
see:	cacheGrammarFromParse

void cacheGrammarFromParse(const bool)	
true:	Cache the grammar in the pool for re-use in subsequent parses.
false:	Do not cache the grammar in the pool
default:	false
note:	The getter function for this method is called isCachingGrammarFromParse
note:	If set to true, the useCachedGrammarInParse is also set to true automatically.
see:	useCachedGrammarInParse

void setStandardUriConformant(const bool)	
true:	Force standard uri conformance.
false:	Do not force standard uri conformance.
default:	false
note:	If set to true, malformed uri will be rejected and fatal error will be issued.

void setCalculateSrcOfs(const bool)	
true:	Enable src offset calculation.
false:	Disable src offset calculation.
default:	false
note:	If set to true, the user can inquire about the current src offset within the input source. Setting it to false (default) improves the performance.

XercesDOMParser Supported Properties

The behavior of the XercesDOMParser is dependant on the values of the following properties. All of the properties below are set using the "setter" methods (e.g. `setExternalSchemaLocation`), and are queried using the corresponding "getter" methods (e.g. `getExternalSchemaLocation`). The following only gives you a quick summary of supported features. Please refer to API Documentation for complete details.

void setExternalSchemaLocation(const XMLCh*)	

Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. Similar situation happens to <import> element in schema documents. This property allows the user to specify a list of schemas to use. If the targetNamespace of a schema specified using this method matches the targetNamespace of a schema occurring in the instance document in schemaLocation attribute, or if the targetNamespace matches the namespace attribute of <import> element, the schema specified by the user using this property will be used (i.e., the schemaLocation attribute in the instance document or on the <import> element will be effectively ignored).
Value	The syntax is the same as for schemaLocation attributes in instance documents: e.g, "http://www.example.com file_name.xsd". The user can specify more than one XML Schema in the list.
Value Type	XMLCh*

void setExternalNoNamespaceSchemaLocation(const XMLCh* const)	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. This property allows the user to specify the no target namespace XML Schema Location externally. If specified, the instance document's noNamespaceSchemaLocation attribute will be effectively ignored.
Value	The syntax is the same as for the noNamespaceSchemaLocation attribute that may occur in an instance document: e.g."file_name.xsd".
Value Type	XMLCh*

void useScanner(const XMLCh* const)	

Description	This property allows the user to specify the name of the XMLScanner to use for scanning XML documents. If not specified, the default scanner "IGXMLScanner" is used.
Value	The recognized scanner names are: 1."WFXMLScanner" - scanner that performs well-formedness checking only. 2. "DGXMLScanner" - scanner that handles XML documents with DTD grammar information. 3. "SGXMLScanner" - scanner that handles XML documents with XML schema grammar information. 4. "IGXMLScanner" - scanner that handles XML documents with DTD or/and XML schema grammar information. Users can use the predefined constants defined in XMLUni directly (fgWFXMLScanner, fgDGXMLScanner, fgSGXMLScanner, or fgIGXMLScanner) or a string that matches the value of one of those constants.
Value Type	XMLCh*
note:	See Use Specific Scanner for more programming details.

void useImplementation(const XMLCh* const)	
Description	This property allows the user to specify a set of features which the parser will then use to acquire an implementation from which it will create the DOMDocument to use when reading in an XML file.
Value Type	XMLCh*

setSecurityManager(Security Manager * const)	
Description	Certain valid XML and XML Schema constructs can force a processor to consume more system resources than an application may wish. In fact, certain features could be exploited by malicious document writers to produce a denial-of-service attack. This property allows applications to impose limits on the amount of resources the processor will consume while processing these constructs.

Value	An instance of the SecurityManager class (see `xercesc/util/SecurityManager`). This class's documentation describes the particular limits that may be set. Note that, when instantiated, default values for limits that should be appropriate in most settings are provided. The default implementation is not thread-safe; if thread-safety is required, the application should extend this class, overriding methods appropriately. The parser will not adopt the SecurityManager instance; the application is responsible for deleting it when it is finished with it. If no SecurityManager instance has been provided to the parser (the default) then processing strictly conforming to the relevant specifications will be performed.
Value Type	SecurityManager*

DOMBuilder

Constructing a DOMBuilder

DOMBuilder is a new interface introduced by the W3C DOM Level 3.0 Abstract Schemas and Load and Save Specification [54] . DOMBuilder provides the "Load" interface for parsing XML documents and building the corresponding DOM document tree from various input sources.

A DOMBuilder instance is obtained from the DOMImplementationLS interface by invoking its createDOMBuilder method. For example:

```
        #include <xercesc/dom/DOM.hpp>
        #include <xercesc/util/XMLString.hpp>
        #include <xercesc/util/PlatformUtils.hpp>

        int main (int argc, char* args[]) {

            try {
                XMLPlatformUtils::Initialize();
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Error during initialization! :\n"
                     << message << "\n";
                XMLString::release(&message);
                return 1;
            }


            XMLCh tempStr[100];
            XMLString::transcode("LS", tempStr, 99);
            DOMImplementation *impl =
   DOMImplementationRegistry::getDOMImplementation(tempStr);
```

```
         DOMBuilder* parser =
createDOMBuilder(DOMImplementationLS::MODE_SYNCHRONOUS, 0);

         // optionally you can set some features on this builder
         if (parser->canSetFeature(XMLUni::fgDOMValidation, true))
             parser->setFeature(XMLUni::fgDOMValidation, true);
         if (parser->canSetFeature(XMLUni::fgDOMNamespaces, true))
             parser->setFeature(XMLUni::fgDOMNamespaces, true);
         if (parser->canSetFeature((XMLUni::fgDOMDatatypeNormalization, true))
             parser->setFeature(XMLUni::fgDOMDatatypeNormalization, true);


         // optionally you can implement your DOMErrorHandler (e.g.
MyDOMErrorHandler)
         // and set it to the builder
         MyDOMErrorHandler* errHandler = new myDOMErrorHandler();
         parser->setErrorHandler(errHandler);

         char* xmlFile = "x1.xml";
         DOMDocument *doc = 0;

         try {
             doc = parser->parseURI(xmlFile);
         }
         catch (const XMLException& toCatch) {
             char* message = XMLString::transcode(toCatch.getMessage());
             cout << "Exception message is: \n"
                  << message << "\n";
             XMLString::release(&message);
             return -1;
         }
         catch (const DOMException& toCatch) {
             char* message = XMLString::transcode(toCatch.msg);
             cout << "Exception message is: \n"
                  << message << "\n";
             XMLString::release(&message);
             return -1;
         }
         catch (...) {
             cout << "Unexpected Exception \n" ;
             return -1;
         }

         parser->release();
         delete errHandler;
         return 0;
     }
```

Please refer to the API Documentation and the sample DOMCount for more detail.

How to interchange DOMInputSource and SAX InputSource?

DOM L3 has introduced a DOMInputSource which is similar to the SAX InputSource. The Xerces-C++ internals (XMLScanner, Reader, etc.) use the SAX InputSource to process the xml data. In order to support DOM L3, we need to provide a mechanism to allow the Xerces-C++ internals to talk to a DOMInputSource object. Similarly, Xerces-C++ provides some framework classes for specialized types of input source (i.e. LocalFileInputSource, etc.) that are derived from the SAX InputSource. In DOM L3, to allow users implementing their own DOMEntityResolver(s), which return a DOMInputSource, to utilize these framework classes, we need to provide a mechanism to map a SAX InputSource to a DOMInputSource. We are introducing to wrapper classes to interchange DOMInputSource and SAXInputSource.

Wrapper4DOMInputSource
Wraps a DOMInputSource object to a SAX InputSource.

```
        #include <xercesc/dom/DOMInputSource.hpp>
        #include <xercesc/framework/Wrapper4DOMInputSource.hpp>

        class DBInputSource: public DOMInputSource
        {
        ...
        };

        ...
        DOMInputSource *domIS = new DBInputSource;
        Wrapper4DOMInputSource domISWrapper(domIS);
        XercesDOMParser parser;

        parser.parse(domISWrapper);
```

Wrapper4InputSource
Wraps a SAX InputSource object to a DOMInputSource.

```
        #include <xercesc/framework/WrapperInputSource.hpp>
        #include <xercesc/framework/LocalFileInputSource.hpp>

        DOMInputSource* MyEntityResolver::resolveEntity(const XMLCh* const publicId,
                                                        const XMLCh* const systemId,
                                                        const XMLCh* const baseURI)
        {
            return new Wrapper4InputSource(new LocalFileInputSource(baseURI,
   systemId));
        }
```

Please refer to the API Documentation for more detail.

DOMBuilder Supported Features
The behavior of the DOMBuilder is dependant on the values of the following features. All of the features

below can be set using the function `DOMBuilder::setFeature(cons XMLCh* const, const bool)`. And can be queried using the function `bool DOMBuilder::getFeature(const XMLCh* const)`. User can also call `DOMBuilder::canSetFeature(const XMLCh* const, const bool)` to query whether setting a feature to a specific value is supported

DOM Features

cdata-sections	
true:	Keep CDATASection nodes in the document.
false:	Not Supported.
default:	true
note:	Setting this feature to false is not supported.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

comments	
true:	Keep Comment nodes in the document.
false:	Discard Comment nodes in the document.
default:	true
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

charset-overrides-xml-encoding	
true:	If a higher level protocol such as HTTP [IETF RFC 2616] provides an indication of the character encoding of the input stream being processed, that will override any encoding specified in the XML declaration or the Text declaration (see also [XML 1.0] 4.3.3 "Character Encoding in Entities"). Explicitly setting an encoding in the DOMInputSource overrides encodings from the protocol.
false:	Any character set encoding information from higher level protocols is ignored by the parser.
default:	true
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

datatype-normalization	
true:	Let the validation process do its datatype normalization that is defined in the used schema language.
false:	Disable datatype normalization. The XML 1.0 attribute value normalization always occurs though.
default:	false
note:	Note that setting this feature to true does not affect the DTD normalization operation which always takes place, in accordance to XML 1.0 (Second Edition) [3] .
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]
see:	XML 1.0 (Second Edition) [3] .

entities	

true:	Create EntityReference nodes in the DOM tree. The EntityReference nodes and their child nodes will be read-only.
false:	Do not create EntityReference nodes in the DOM tree. No EntityReference nodes will be created, only the nodes corresponding to their fully expanded sustitution text will be created.
default:	true
note:	This feature only affects the appearance of EntityReference nodes in the DOM tree. The document will always contain the entity reference child nodes.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

canonical-form	
true:	Not Supported.
false:	Do not canonicalize the document.
default:	false
note:	Setting this feature to true is not supported.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

infoset	
true:	Not Supported.
false:	No effect.
default:	false
note:	Setting this feature to true is not supported.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

namespaces	
true:	Perform Namespace processing
false:	Do not perform Namespace processing
default:	false
note:	If the validation is on, then the document must contain a grammar that supports the use of namespaces
see:	validation
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

namespace-declarations	
true:	Include namespace declaration attributes, specified or defaulted from the schema or the DTD, in the document.
false:	Not Supported.
default:	true
note:	Setting this feature to false is not supported.
see:	namespaces
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

supported-mediatypes-only	
true:	Not Supported.
false:	Don't check the media type, accept any type of data.
default:	false

note:	Setting this feature to true is not supported.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

validate-if-schema	
true:	When validation is true, the parser will validate the document only if a grammar is specified.
false:	Validation is determined by the state of the validation feature.
default:	false
see:	validation
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

validation	
true:	Report all validation errors.
false:	Do not report validation errors.
default:	false
note:	If this feature is set to true, the document must specify a grammar. If this feature is set to false and document specifies a grammar, that grammar might be parsed but no validation of the document contents will be performed.
see:	validate-if-schema
see:	http://apache.org/xml/features/nonvalidating/load-external-dtd
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

whitespace-in-element-content	
true:	Include text nodes that can be considered "ignorable whitespace" in the DOM tree.
false:	Do not include ignorable whitespace in the DOM tree.
default:	true
note:	The only way that the parser can determine if text is ignorable is by reading the associated grammar and having a content model for the document. When ignorable whitespace text nodes are included in the DOM tree, they will be flagged as ignorable; and the method DOMText::isIgnorableWhitespace() will return true for those text nodes.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

Xerces Features

http://apache.org/xml/features/validation/schema	
true:	Enable the parser's schema support.
false:	Disable the parser's schema support.
default:	false
note	If set to true, namespace processing must also be turned on.
see:	namespaces

http://apache.org/xml/features/validation/schema-full-checking	
true:	Enable full schema constraint checking, including checking which may be time-consuming or memory intensive. Currently, particle unique attribution constraint checking and particle derivation restriction checking are controlled by this option.
false:	Disable full schema constraint checking.
default:	false
note:	This feature checks the Schema grammar itself for additional errors that are time-consuming or memory intensive. It does **not** affect the level of checking performed on document instances that use Schema grammars.
see:	http://apache.org/xml/features/validation/schema

http://apache.org/xml/features/nonvalidating/load-external-dtd	
true:	Load the External DTD.
false:	Ignore the external DTD completely.
default:	true
note	This feature is ignored and DTD is always loaded when validation is on.
see:	validation

http://apache.org/xml/features/continue-after-fatal-error	
true:	Attempt to continue parsing after a fatal error.
false:	Stops parse on first fatal error.
default:	false
note:	The behavior of the parser when this feature is set to true is **undetermined**! Therefore use this feature with extreme caution because the parser may get stuck in an infinite loop or worse.

http://apache.org/xml/features/validation-error-as-fatal	
true:	The parser will treat validation error as fatal and will exit depends on the state of http://apache.org/xml/features/continue-after-fatal-error .
false:	The parser will report the error and continue processing.
default:	false

note:	Setting this true does not mean the validation error will be printed with the word "Fatal Error". It is still printed as "Error", but the parser will exit if http://apache.org/xml/features/continue-after-fatal-error is set to false.
see:	http://apache.org/xml/features/continue-after-fatal-error

http://apache.org/xml/features/validation/use-cachedGrammarInParse	
true:	Use cached grammar if it exists in the pool.
false:	Parse the schema grammar.
default:	false
note:	If http://apache.org/xml/features/validation/cache-grammarF is enabled, this feature is set to true automatically. Any setting to this feature by the users is a no-op.
see:	http://apache.org/xml/features/validation/cache-grammarF

http://apache.org/xml/features/validation/cache-grammarFromParse	
true:	Cache the grammar in the pool for re-use in subsequent parses.
false:	Do not cache the grammar in the pool
default:	false
note:	If set to true, the http://apache.org/xml/features/validation/use-cachedGran is also set to true automatically.
see:	http://apache.org/xml/features/validation/use-cachedGran

http://apache.org/xml/features/standard-uri-conformant	
true:	Force standard uri conformance.
false:	Do not force standard uri conformance.
default:	false
note:	If set to true, malformed uri will be rejected and fatal error will be issued.

http://apache.org/xml/features/calculate-src-ofs	
true:	Enable src offset calculation.
false:	Disable src offset calculation.
default:	false
note:	If set to true, the user can inquire about the current src offset within the input source. Setting it to false (default) improves the performance.

http://apache.org/xml/features/dom/user-adopts-DOMDocument	

true:	The caller will adopt the DOMDocument that is returned from the parse method and thus is responsible to call DOMDocument::release() to release the associated memory. The parser will not release it. The ownership is transferred from the parser to the caller.
false:	The returned DOMDocument from the parse method is owned by the parser and thus will be deleted when the parser is released.
default:	false
see:	DOMBuilder API Documentation, (DOMBuilder::parse and DOMBuilder::resetDocumentPool)

DOMBuilder Supported Properties

The behavior of the DOMBuilder is dependant on the values of the following properties. All of the properties below can be set using the function `DOMBuilder::setProperty(const XMLCh* const, void*)`. It takes a void pointer as the property value. Application is required to initialize this void pointer to a correct type. Please check the column "Value Type" below to learn exactly what type of property value each property expects for processing. Passing a void pointer that was initialized with a wrong type will lead to unexpected result. If the same property is set more than once, the last one takes effect.

Property values can be queried using the function `void* DOMBuilder::getFeature(const XMLCh* const)`. The parser owns the returned pointer, and the memory allocated for the returned pointer will be destroyed when the parser is released. To ensure accessibility of the returned information after the parser is released, callers need to copy and store the returned information somewhere else. Since the returned pointer is a generic void pointer, check the column "Value Type" below to learn exactly what type of object each property returns for replication.

Xerces Properties

http://apache.org/xml/properties/schema/external-schemaLocation

Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. Similar situation happens to <import> element in schema documents. This property allows the user to specify a list of schemas to use. If the targetNamespace of a schema specified using this method matches the targetNamespace of a schema occurring in the instance document in schemaLocation attribute, or if the targetNamespace matches the namespace attribute of <import> element, the schema specified by the user using this property will be used (i.e., the schemaLocation attribute in the instance document or on the <import> element will be effectively ignored).
Value	The syntax is the same as for schemaLocation attributes in instance documents: e.g, "http://www.example.com file_name.xsd". The user can specify more than one XML Schema in the list.
Value Type	XMLCh*

http://apache.org/xml/properties/schema/external-noNamespaceSchemaLocation	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. This property allows the user to specify the no target namespace XML Schema Location externally. If specified, the instance document's noNamespaceSchemaLocation attribute will be effectively ignored.
Value	The syntax is the same as for the noNamespaceSchemaLocation attribute that may occur in an instance document: e.g."file_name.xsd".
Value Type	XMLCh*

http://apache.org/xml/properties/scannerName	

Description	This property allows the user to specify the name of the XMLScanner to use for scanning XML documents. If not specified, the default scanner "IGXMLScanner" is used.
Value	The recognized scanner names are: 1."WFXMLScanner" - scanner that performs well-formedness checking only. 2. "DGXMLScanner" - scanner that handles XML documents with DTD grammar information. 3. "SGXMLScanner" - scanner that handles XML documents with XML schema grammar information. 4. "IGXMLScanner" - scanner that handles XML documents with DTD or/and XML schema grammar information. Users can use the predefined constants defined in XMLUni directly (fgWFXMLScanner, fgDGXMLScanner, fgSGXMLScanner, or fgIGXMLScanner) or a string that matches the value of one of those constants.
Value Type	XMLCh*
note:	See Use Specific Scanner for more programming details.

http://apache.org/xml/properties/parser-use-DOMDocument-from-Implementation	
Description	This property allows the user to specify a set of features which the parser will then use to acquire an implementation from which it will create the DOMDocument to use when reading in an XML file.
Value Type	XMLCh*

http://apache.org/xml/properties/security-manager	
Description	Certain valid XML and XML Schema constructs can force a processor to consume more system resources than an application may wish. In fact, certain features could be exploited by malicious document writers to produce a denial-of-service attack. This property allows applications to impose limits on the amount of resources the processor will consume while processing these constructs.

Value	An instance of the SecurityManager class (see `xercesc/util/SecurityManager`). This class's documentation describes the particular limits that may be set. Note that, when instantiated, default values for limits that should be appropriate in most settings are provided. The default implementation is not thread-safe; if thread-safety is required, the application should extend this class, overriding methods appropriately. The parser will not adopt the SecurityManager instance; the application is responsible for deleting it when it is finished with it. If no SecurityManager instance has been provided to the parser (the default) then processing strictly conforming to the relevant specifications will be performed.
Value Type	SecurityManager*

DOMWriter

Constructing a DOMWriter

DOMWriter is a new interface introduced by the W3C DOM Level 3.0 Abstract Schemas and Load and Save Specification [54] . DOMWriter provides the "Save" interface for serializing (writing) a DOM document into XML data. The XML data can be written to various type of output stream.

A DOMWriter instance is obtained from the DOMImplementationLS interface by invoking its createDOMWriter method. For example:

```
        #include <xercesc/dom/DOM.hpp>
        #include <xercesc/util/XMLString.hpp>
        #include <xercesc/util/PlatformUtils.hpp>

        int serializeDOM(DOMNode* node) {

            XMLCh tempStr[100];
            XMLString::transcode("LS", tempStr, 99);
            DOMImplementation *impl =
   DOMImplementationRegistry::getDOMImplementation(tempStr);
            DOMWriter* theSerializer =
   createDOMWriter();

            // optionally you can set some features on this serializer
            if (theSerializer->canSetFeature(XMLUni::fgDOMWRTDiscardDefaultContent,
   true))
                theSerializer->setFeature(XMLUni::fgDOMWRTDiscardDefaultContent,
   true);

            if (theSerializer->canSetFeature(XMLUni::fgDOMWRTFormatPrettyPrint,
   true))
                 theSerializer->setFeature(XMLUni::fgDOMWRTFormatPrettyPrint, true);
```

```
         // optionally you can implement your DOMWriterFilter (e.g.
MyDOMWriterFilter)
         // and set it to the serializer
         DOMWriterFilter* myFilter = new myDOMWriterFilter();
         theSerializer->setFilter(myFilter);

         // optionally you can implement your DOMErrorHandler (e.g.
MyDOMErrorHandler)
         // and set it to the serializer
         DOMErrorHandler* errHandler = new myDOMErrorHandler();
         theSerializer->setErrorHandler(myErrorHandler);

         // StdOutFormatTarget prints the resultant XML stream
         // to stdout once it receives any thing from the serializer.
         XMLFormatTarget *myFormTarget = new StdOutFormatTarget();

         try {
             // do the serialization through DOMWriter::writeNode();
             theSerializer->writeNode(myFormTarget, *node);
         }
         catch (const XMLException& toCatch) {
             char* message = XMLString::transcode(toCatch.getMessage());
             cout << "Exception message is: \n"
                  << message << "\n";
             XMLString::release(&message);
             return -1;
         }
         catch (const DOMException& toCatch) {
             char* message = XMLString::transcode(toCatch.msg);
             cout << "Exception message is: \n"
                  << message << "\n";
             XMLString::release(&message);
             return -1;
         }
         catch (...) {
             cout << "Unexpected Exception \n" ;
             return -1;
         }


         theSerializer->release();
         delete myErrorHandler;
         delete myFilter;
         delete myFormTarget;
         return 0;
     }
```

Please refer to the API Documentation and the sample DOMPrint for more detail.

How does DOMWriter handle built-in entity Reference in node value?

Say for example you parse the following xml document using XercesDOMParser or DOMBuilder

```
<root>
<Test attr=" > ' &lt; &gt; &amp; &quot; &apos; "></Test>
<Test attr=' > " &lt; &gt; &amp; &quot; &apos; '></Test>
<Test> >  " ' &lt; &gt; &amp; &quot; &apos; </Test>
<Test><![CDATA[< > & " ' &lt; &gt; &amp; &quot; &apos; ] ]></Test>
</root>
```

According to XML 1.0 spec, 4.4 XML Processor Treatment of Entities and References, the parser will expand the entity reference as follows

```
<root>
<Test attr=" > ' < > & " ' "></Test>
<Test attr=' > " < > & " ' '></Test>
<Test> >  " ' < > & " ' </Test>
<Test><![CDATA[< > & " ' &lt; &gt; &amp; &quot; &apos; ] ]></Test>
</root>
```

and pass such DOMNode to DOMWriter for serialization. From DOMWriter perspective, it does not know what the original string was. All it sees is above DOMNode from the parser. But since the DOMWriter is supposed to generate something that is parsable if sent back to the parser, it cannot print such string as is. Thus the DOMWriter is doing some "touch up", just enough, to get the string parsable.

So for example since the appearance of < and & in text value will lead to not well-form XML error, the DOMWriter fixes them to &lt; and &amp; respectively; while the >, ' and " in text value are ok to the parser, so DOMWriter does not do anything to them. Similarly the DOMWriter fixes some of the characters for the attribute value but keep everything in CDATA.

So the string that is generated by DOMWriter will look like this

```
<root>
<Test attr=" > ' &lt; > &amp; &quot; ' "/>
<Test attr=" > &quot; &lt; > &amp; &quot; ' "/>
<Test> >  " ' &lt; > &amp; " ' </Test>
<Test><![CDATA[< > & " ' &lt; &gt; &amp; &quot; &apos; ] ]></Test>
</root>
```

To summarize, here is the table that summarize how built-in entity refernece are handled for different Node Type:

Input/Output	**<**	**>**	**&**	**"**	**'**	**&lt;**	**&gt;**	**&amp;**	**&quot;**	**&apos;**
Attribute	N/A	>	N/A	&quot;	'	&lt;	>	&amp;	&quot;	'
Text	N/A	>	N/A	"	'	&lt;	>	&amp;	"	'
CDATA	<	>	&	"	'	&lt;	&gt;	&amp;	&quot;	&apos;

DOMWriter Supported Features

The behavior of the DOMWriter is dependant on the values of the following features. All of the features below can be set using the function `DOMWriter::setFeature(cons XMLCh* const, bool)`. And can be queried using the function `bool DOMWriter::getFeature(const XMLCh* const)`. User can also call `DOMWriter::canSetFeature(const XMLCh* const, bool)` to query whether setting a feature to a specific value is supported

DOM Features

discard-default-content	
true:	Use whatever information available to the implementation (i.e. XML schema, DTD, the specified flag on Attr nodes, and so on) to decide what attributes and content should be discarded or not.
false:	Keep all attributes and all content.
default:	true
note:	Note that the specified flag on Attr nodes in itself is not always reliable, it is only reliable when it is set to false since the only case where it can be set to false is if the attribute was created by the implementation. The default content won't be removed if an implementation does not have any information available.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

entities	
true:	EntityReference nodes are serialized as an entity reference of the form entityName;" in the output.
false:	EntityReference nodes are serialized as expanded sustitution text, unless the corresponding entity definition is not found.
default:	true
note:	This feature only affects the output XML stream. The dom tree to be serialized will not be changed.
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

canonical-form	
true:	Not Supported.
false:	Do not canonicalize the output.
default:	false
note:	Setting this feature to true is not supported.
see:	format-pretty-print
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

format-pretty-print	
true:	Formatting the output by adding whitespace to produce a pretty-printed, indented, human-readable form. The exact form of the transformations is not specified by this specification.
false:	Don't pretty-print the result.
default:	false
note:	Setting this feature to true will set the feature canonical-form to false.
see:	canonical-form

see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

normalize-characters	
true:	Not Supported.
false:	Do not perform character normalization.
note:	Setting this feature to true is not supported.
default:	false
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

split-cdata-sections	
true:	Split CDATA sections containing the CDATA section termination marker ']]>', or unrepresentable characters in the output encoding. When a CDATA section is split a warning is issued.
false:	Signal an error if a CDATASection contains CDATA section termination marker ']]>', or an unrepresentable character.
default:	true
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

validation	
true:	Not Supported.
false:	Do not report validation errors.
note:	Setting this feature to true is not supported.
default:	false
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

whitespace-in-element-content	
true:	Include text nodes that can be considered "ignorable whitespace" in the DOM tree.
false:	Not Supported.
note:	Setting this feature to false is not supported.
default:	true
see:	DOM Level 3.0 Abstract Schemas and Load and Save Specification [54]

Xerces Features

byte-order-mark	
true:	Enable the writing of the Byte-Order-Mark (BOM), in the resultant XML stream.
false:	Disable the writing of BOM.

note:	The BOM is written at the beginning of the resultant XML stream, if and only if a DOMDocumentNode is rendered for serialization, and the output encoding is among the encodings listed here (alias acceptable), UTF-16, UTF-16LE, UTF-16BE, UCS-4, UCS-4LE, and UCS-4BE. In the case of UTF-16/UCS-4, the system directive, ENDIANMODE_LITTLE and ENDIANMODE_BIG (which denotes the host machine's endian mode), is refered to determine the appropriate BOM to be written.
default:	false
see:	XML 1.0 Appendix F [55] for more information about BOM.

Deprecated - Java-like DOM

Earlier, Xerces-C++ has provided a set of C++ DOM interfaces that is very similar in design and use, to the Java DOM API bindings. Currently, such interface has been deprecated. See this document for its programming details.

32 DOM Level 3 Support

Disclaimer

The Xerces-C++ 2.5.0 contains a partial implementation of the W3C Document Object Model Level 3. This implementation is experimental. The DOM Level 3 specification is still in working draft stage: you should not consider this implementation complete or correct.

The limitations of this implementation are detailed below. Please, read this document before using Xerces-C++ 2.5.0.

Introduction

The Xerces-C++ 2.5.0 contains an experimental implementation of a subset of the W3C DOM Level 3 as specified in

- DOM Level 3.0 Core Specification [13] , Version 1.0 W3C Working Draft 26 February 2003 and
- Document Object Model (DOM) Level 3 Load and Save Specification [14] , Version 1.0 W3C Working Draft 26 February 2003

Implementation of DOM Level 3 Core

The following are implemented in Xerces-C++ 2.5.0.

- `DOMImplementationRegistry`, `DOMImplementationSource`
- `DOMException`: VALIDATION_ERR
- `DOMDocument`: set/getActualEncoding, set/getEncoding, set/getVersion, set/getStandalone, set/getDocumentURI, set/getStrictErrorChecking, renameNode, normalizeDocument
- `DOMEntity`: set/getActualEncoding, set/getEncoding, set/getVersion
- `DOMErrorHandler`, `DOMError`, `DOMLocator`
- `DOMNode`: set/getUserData, isSameNode, isEqualNode, compareTreePosition, lookupNamespaceURI, lookupNamespacePrefix, isDefaultNamespace, baseURI
- `DOMText`: getIsWhitespaceInElementContent
- `DOMAttr`: isID, getTypeInfo
- `DOMElement`: setIdAttributeNode, setIdAttributeNS, setIdAttribute, getTypeInfo
- `DOMUserDataHandler`
- `DOMConfiguration`

Implementation of DOM Level 3 Load and Save

The following are implemented in Xerces-C++ 2.5.0.

- `DOMImplementationLS`
- `DOMBuilder`: does not provide implementation of parseWithContext() and `DOMBuilderFilter`.
- `DOMEntityResolver`

- `DOMInputSource`
- `DOMWriter` and `DOMWriterFilter`

Implementation of DOM Level 3 Abstract Schemas

The entire section has not been implemented in Xerces-C++ 2.5.0.

33 Deprecated - Java-like DOM

Deprecated - Java-like DOM

Earlier, Xerces-C++ has provided a set of C++ DOM interfaces that is very similar in design and use, to the Java DOM API bindings. It uses smart pointer interface and uses reference counting in memory management.

Currently, such interface has been deprecated and is provided just as a viable alternative for those users who like the idea of Java-like smart pointer design. Please note that such interface may not be kept up to date to the latest W3C DOM specification.

Users are recommended to migrate to the Apache Recommended DOM C++ binding.

Using this set of deprecated API

Accessing API from application code

```

        // C++
        #include <xercesc/dom/deprecated/DOM.hpp>
```

```
    // Compared to Java
        import org.w3c.dom.*
```

The header file <dom/deprecated/DOM.hpp> includes all the individual headers for this set of deprecated DOM API classes.

Class Names

The C++ class names are prefixed with "DOM_". The intent is to prevent conflicts between DOM class names and other names that may already be in use by an application or other libraries that a DOM based application must link with.

The use of C++ namespaces would also have solved this conflict problem, but for the fact that many compilers do not yet support them.

```
    DOM_Document   myDocument;   // C++
        DOM_Node       aNode;
        DOM_Text       someText;
```

```
    Document       myDocument;   // Compared to Java
        Node           aNode;
        Text           someText;
```

If you wish to use the Java class names in C++, then you need to typedef them in C++. This is not advisable for the general case - conflicts really do occur - but can be very useful when converting a body of existing Java code to C++.

```
    typedef DOM_Document  Document;
        typedef DOM_Node      Node;

        Document   myDocument;        // Now C++ usage is
                                      // indistinguishable from Java
        Node       aNode;
```

Objects and Memory Management

This deprecated C++ DOM implementation uses automatic memory management, implemented using reference counting. As a result, the C++ code for most DOM operations is very similar to the equivalent Java code, right down to the use of factory methods in the DOM document class for nearly all object creation, and the lack of any explicit object deletion.

Consider the following code snippets

```
    // This is C++
        DOM_Node       aNode;
        aNode = someDocument.createElement("ElementName");
        DOM_Node docRootNode = someDoc.getDocumentElement();
        docRootNode.AppendChild(aNode);
```

```
    // This is Java
        Node       aNode;
        aNode = someDocument.createElement("ElementName");
        Node docRootNode = someDoc.getDocumentElement();
        docRootNode.AppendChild(aNode);
```

The Java and the C++ are identical on the surface, except for the class names, and this similarity remains true for most DOM code.

However, Java and C++ handle objects in somewhat different ways, making it important to understand a little bit of what is going on beneath the surface.

In Java, the variable `aNode` is an object reference , essentially a pointer. It is initially == null, and references an object only after the assignment statement in the second line of the code.

In C++ the variable `aNode` is, from the C++ language's perspective, an actual live object. It is constructed when the first line of the code executes, and DOM_Node::operator = () executes at the second line. The C++ class DOM_Node essentially a form of a smart-pointer; it implements much of the behavior of a Java Object Reference variable, and delegates the DOM behaviors to an implementation class that lives behind the scenes.

Key points to remember when using the C++ DOM classes:

- Create them as local variables, or as member variables of some other class. Never "new" a DOM object into the heap or make an ordinary C pointer variable to one, as this will greatly confuse the automatic memory management.
- The "real" DOM objects - nodes, attributes, CData sections, whatever, do live on the heap, are created with the create... methods on class DOM_Document. DOM_Node and the other DOM classes serve as reference variables to the underlying heap objects.
- The visible DOM classes may be freely copied (assigned), passed as parameters to functions, or returned by value from functions.

- Memory management of the underlying DOM heap objects is automatic, implemented by means of reference counting. So long as some part of a document can be reached, directly or indirectly, via reference variables that are still alive in the application program, the corresponding document data will stay alive in the heap. When all possible paths of access have been closed off (all of the application's DOM objects have gone out of scope) the heap data itself will be automatically deleted.
- There are restrictions on the ability to subclass the DOM classes.

String Type - DOMString

Class DOMString provides the mechanism for passing string data to and from the DOM API. DOMString is not intended to be a completely general string class, but rather to meet the specific needs of the DOM API.

The design derives from two primary sources: from the DOM's CharacterData interface and from class `java.lang.string`.

Main features are:

- It stores Unicode text.
- Automatic memory management, using reference counting.
- DOMStrings are mutable - characters can be inserted, deleted or appended.

When a string is passed into a method of the DOM, when setting the value of a Node, for example, the string is cloned so that any subsequent alteration or reuse of the string by the application will not alter the document contents. Similarly, when strings from the document are returned to an application via the DOM API, the string is cloned so that the document can not be inadvertently altered by subsequent edits to the string.

> ***Note:*** *The ICU classes are a more general solution to UNICODE character handling for C++ applications. ICU is an Open Source Unicode library, available at the IBM DeveloperWorks website [22] .*

Equality Testing

The DOMString equality operators (and all of the rest of the DOM class conventions) are modeled after the Java equivalents. The equals() method compares the content of the string, while the == operator checks whether the string reference variables (the application program variables) refer to the same underlying string in memory. This is also true of DOM_Node, DOM_Element, etc., in that operator == tells whether the variables in the application are referring to the same actual node or not. It's all very Java-like

- bool operator == () is true if the DOMString variables refer to the same underlying storage.
- bool equals() is true if the strings contain the same characters.

Here is an example of how the equality operators work:

```
    DOMString a = "Hello";
        DOMString b = a;
        DOMString c = a.clone();
        if (b == a)           //  This is true
        if (a == c)           //  This is false
        if (a.equals(c))       //  This is true
        b = b + " World";
        if (b == a)           // Still true, and the string's
                              //    value is "Hello World"
        if (a.equals(c))      // false.  a is "Hello World";
                              //    c is still "Hello".
```

Downcasting

Application code sometimes must cast an object reference from DOM_Node to one of the classes deriving from DOM_Node, DOM_Element, for example. The syntax for doing this in C++ is different from that in Java.

```
    // This is C++
        DOM_Node       aNode = someFunctionReturningNode();
        DOM_Element    el = (DOM_Element &) aNode;
```

```
    // This is Java
        Node       aNode = someFunctionReturningNode();
        Element    el = (Element) aNode;
```

The C++ cast is not type-safe; the Java cast is checked for compatible types at runtime. If necessary, a type-check can be made in C++ using the node type information:

```
    // This is C++

        DOM_Node       aNode = someFunctionReturningNode();
        DOM_Element    el;    // by default, el will == null.

        if (anode.getNodeType() == DOM_Node::ELEMENT_NODE)
           el = (DOM_Element &) aNode;
        else
           // aNode does not refer to an element.
           // Do something to recover here.
```

Subclassing

The C++ DOM classes, DOM_Node, DOM_Attr, DOM_Document, etc., are not designed to be subclassed by an application program.

As an alternative, the DOM_Node class provides a User Data field for use by applications as a hook for extending nodes by referencing additional data or objects. See the API description for DOM_Node for details.

DOMParser

Constructing a DOMParser

In order to use Xerces-C++ to parse XML files using the deprecated DOM, you will need to create an instance of the DOMParser class. The example below shows the code you need in order to create an instance of the DOMParser.

```

        #include <xercesc/dom/deprecated/DOMParser.hpp>
        #include <xercesc/dom/deprecated/DOM.hpp>
        #include <xercesc/sax/HandlerBase.hpp>
        #include <xercesc/util/XMLString.hpp>

        int main (int argc, char* args[]) {

            try {
                XMLPlatformUtils::Initialize();
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Error during initialization! :\n"
                     << message << "\n";
                XMLString::release(&message);
                return 1;
            }

            char* xmlFile = "x1.xml";
            DOMParser* parser = new DOMParser();
            parser->setValidationScheme(DOMParser::Val_Always);    // optional.
            parser->setDoNamespaces(true);    // optional

            ErrorHandler* errHandler = (ErrorHandler*) new HandlerBase();
            parser->setErrorHandler(errHandler);

            try {
                parser->parse(xmlFile);
            }
            catch (const XMLException& toCatch) {
                char* message = XMLString::transcode(toCatch.getMessage());
                cout << "Exception message is: \n"
                     << message << "\n";
                XMLString::release(&message);
                return -1;
            }
            catch (const DOM_DOMException& toCatch) {
                cout << "Exception message is: \n"
                     << toCatch.code << "\n";
                return -1;
            }
            catch (...) {
                cout << "Unexpected Exception \n" ;
                return -1;
            }

            delete parser;
            delete errHandler;
            return 0;
        }
```

DOMParser Supported Features

The behavior of the DOMParser is dependant on the values of the following features. All of the features below are set using the "setter" methods (e.g. `setDoNamespaces`), and are queried using the corresponding "getter" methods (e.g. `getDoNamespaces`). The following only gives you a quick summary of supported features. Please refer to API Documentation for complete detail.

None of these features can be modified in the middle of a parse, or an exception will be thrown.

void setCreateEntityReferenceNodes(const bool)	
true:	Create EntityReference nodes in the DOM tree. The EntityReference nodes and their child nodes will be read-only.
false:	Do not create EntityReference nodes in the DOM tree. No EntityReference nodes will be created, only the nodes corresponding to their fully expanded sustitution text will be created.
default:	true
note:	This feature only affects the appearance of EntityReference nodes in the DOM tree. The document will always contain the entity reference child nodes.

void setExpandEntityReferences(const bool) (deprecated) please use setCreateEntityReferenceNodes	
true:	Do not create EntityReference nodes in the DOM tree. No EntityReference nodes will be created, only the nodes corresponding to their fully expanded sustitution text will be created.
false:	Create EntityReference nodes in the DOM tree. The EntityReference nodes and their child nodes will be read-only.
default:	false
see:	setCreateEntityReferenceNodes

void setIncludeIgnorableWhitespace(const bool)	
true:	Include text nodes that can be considered "ignorable whitespace" in the DOM tree.
false:	Do not include ignorable whitespace in the DOM tree.
default:	true
note:	The only way that the parser can determine if text is ignorable is by reading the associated grammar and having a content model for the document. When ignorable whitespace text nodes are included in the DOM tree, they will be flagged as ignorable; and the method DOMText::isIgnorableWhitespace() will return true for those text nodes.

void setDoNamespaces(const bool)	

true:	Perform Namespace processing
false:	Do not perform Namespace processing
default:	false
note:	If the validation scheme is set to Val_Always or Val_Auto, then the document must contain a grammar that supports the use of namespaces
see:	setValidationScheme

void setDoValidation(const bool) (deprecated) please use setValidationScheme	
true:	Report all validation errors.
false:	Do not report validation errors.
default:	see the default of setValidationScheme
see:	setValidationScheme

void setValidationScheme(const ValSchemes)	
Val_Auto:	The parser will report validation errors only if a grammar is specified.
Val_Always:	The parser will always report validation errors.
Val_Never:	Do not report validation errors.
default:	Val_Auto
note:	If set to Val_Always, the document must specify a grammar. If this feature is set to Val_Never and document specifies a grammar, that grammar might be parsed but no validation of the document contents will be performed.
see:	setLoadExternalDTD

void setDoSchema(const bool)	
true:	Enable the parser's schema support.
false:	Disable the parser's schema support.
default:	false
note	If set to true, namespace processing must also be turned on.
see:	setDoNamespaces

void setValidationSchemaFullChecking(const bool)	

true:	Enable full schema constraint checking, including checking which may be time-consuming or memory intensive. Currently, particle unique attribution constraint checking and particle derivation restriction checking are controlled by this option.
false:	Disable full schema constraint checking .
default:	false
note:	This feature checks the Schema grammar itself for additional errors that are time-consuming or memory intensive. It does **not** affect the level of checking performed on document instances that use Schema grammars.
see:	setDoSchema

void setLoadExternalDTD(const bool)	
true:	Load the External DTD .
false:	Ignore the external DTD completely.
default:	true
note	This feature is ignored and DTD is always loaded if the validation scheme is set to Val_Always or Val_Auto.
see:	setValidationScheme

void setExitOnFirstFatalError(const bool)	
true:	Stops parse on first fatal error.
false:	Attempt to continue parsing after a fatal error.
default:	true
note:	The behavior of the parser when this feature is set to false is **undetermined**! Therefore use this feature with extreme caution because the parser may get stuck in an infinite loop or worse.

void setValidationConstraintFatal(const bool)	
true:	The parser will treat validation error as fatal and will exit depends on the state of setExitOnFirstFatalError
false:	The parser will report the error and continue processing.
default:	false
note:	Setting this true does not mean the validation error will be printed with the word "Fatal Error". It is still printed as "Error", but the parser will exit if setExitOnFirstFatalError is set to true.

see:	setExitOnFirstFatalError

void useCachedGrammarInParse(const bool)	
true:	Use cached grammar if it exists in the pool.
false:	Parse the schema grammar.
default:	false
note:	The getter function for this method is called isUsingCachedGrammarInParse
note:	If the grammar caching option is enabled, this option is set to true automatically. Any setting to this option by the users is a no-op.
see:	cacheGrammarFromParse

void cacheGrammarFromParse(const bool)	
true:	cache the grammar in the pool for re-use in subsequent parses.
false:	Do not cache the grammar in the pool
default:	false
note:	The getter function for this method is called isCachingGrammarFromParse
note:	If set to true, the useCachedGrammarInParse is also set to true automatically.
see:	useCachedGrammarInParse

void setStandardUriConformant(const bool)	
true:	Force standard uri conformance.
false:	Do not force standard uri conformance.
default:	false
note:	If set to true, malformed uri will be rejected and fatal error will be issued.

void setCalculateSrcOfs(const bool)	
true:	Enable src offset calculation.
false:	Disable src offset calculation.
default:	false
note:	If set to true, the user can inquire about the current src offset within the input source. Setting it to false (default) improves the performance.

void setExternalSchemaLocation(const XMLCh* const)	

Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. Similar situation happens to <import> element in schema documents. This property allows the user to specify a list of schemas to use. If the targetNamespace of a schema specified using this method matches the targetNamespace of a schema occurring in the instance document in schemaLocation attribute, or if the targetNamespace matches the namespace attribute of <import> element, the schema specified by the user using this property will be used (i.e., the schemaLocation attribute in the instance document or on the <import> element will be effectively ignored).
Value	The syntax is the same as for schemaLocation attributes in instance documents: e.g, "http://www.example.com file_name.xsd". The user can specify more than one XML Schema in the list.
Value Type	XMLCh*

void setExternalNoNamespaceSchemaLocation(const XMLCh* const)	
Description	The XML Schema Recommendation explicitly states that the inclusion of schemaLocation/ noNamespaceSchemaLocation attributes in the instance document is only a hint; it does not mandate that these attributes must be used to locate schemas. This property allows the user to specify the no target namespace XML Schema Location externally. If specified, the instance document's noNamespaceSchemaLocation attribute will be effectively ignored.
Value	The syntax is the same as for the noNamespaceSchemaLocation attribute that may occur in an instance document: e.g."file_name.xsd".
Value Type	XMLCh*

void useScanner(const XMLCh* const)	

Description	This property allows the user to specify the name of the XMLScanner to use for scanning XML documents. If not specified, the default scanner "IGXMLScanner" is used.
Value	The recognized scanner names are: 1."WFXMLScanner" - scanner that performs well-formedness checking only. 2. "DGXMLScanner" - scanner that handles XML documents with DTD grammar information. 3. "SGXMLScanner" - scanner that handles XML documents with XML schema grammar information. 4. "IGXMLScanner" - scanner that handles XML documents with DTD or/and XML schema grammar information. Users can use the predefined constants defined in XMLUni directly (fgWFXMLScanner, fgDGXMLScanner, fgSGXMLScanner, or fgIGXMLScanner) or a string that matches the value of one of those constants.
Value Type	XMLCh*
note:	See Use Specific Scanner for more programming details.

34 Schema

Introduction

This package contains an implementation of the W3C XML Schema Language, a recommendation of the Worldwide Web Consortium available in three parts: XML Schema: Primer [41] and XML Schema: Structures [42] and XML Schema: Datatypes [43] . We consider this implementation complete except for the limitations cited below.

We would very much appreciate feedback on the package via the Xerces-C++ mailing list xerces-c-dev@xml.apache.org [21] , and we encourage the submission of bugs as described in Bug-Reporting page. Please read this document before using this package.

Limitations

- Due to the way in which the parser constructs content models for elements with complex content, specifying large values for the `minOccurs` or `maxOccurs` attributes may cause a stack overflow or very poor performance in the parser. Large values for `minOccurs` should be avoided, and `unbounded` should be used instead of a large value for `maxOccurs`.

Interpretation of Areas that are Unclear or Implementation-Dependent

keyref

We have interpreted the specs as requiring <keyref> Identity Constraints to refer to <key> or <unique> identity constraints within the scope of the elements to which the <keyref> is attached. This interpretation is at variance with the Schema Primer, which contains an example with a <keyref> declared on an element used inside the element of its corresponding <key>.

out-of-bound float values

For float data, the specs does not explicitly specify how to deal with out-of-bound data. Xerces converts these values as below

Values in range	Values converted
less than -3.402823466e+38	-INF
greater than -1.175494351e-38 and less than -0	-0
greater than +0 and less than +1.175494351e-38	+0
greater than +3.402823466e+38	+INF

The effect of this conversion would invalidate an instance data, for example, "1.1e-39", of a data type derived from float, with minExclusive value '+0', since "1.1e-39" is converted to "+0", which is the same as the minExclusive.

out-of-bound double values

Similarly, Xerces converts double values as below

Values in range	Values converted
less than -1.7976931348623158e+308	-INF
greater than -2.2250738585072014e-308 and less than -0	-0
greater than +0 and less than +2.2250738585072014e-308	+0
greater than +1.7976931348623158e+308	+INF

Usage

Here is an example how to turn on schema processing in DOMParser (default is off). Note that you must also turn on namespace support (default is off) for schema processing.

```
    // Instantiate the DOM parser.
    DOMParser parser;
    parser.setDoNamespaces(true);
    parser.setDoSchema(true);
    parser.parse(xmlFile);
```

Usage in SAXParser is similar, please refer to the sample program 'samples/SAXCount/SAXCount.cpp' for further reference.

Here is an example how to turn on schema processing in SAX2XMLReader (default is on). Note that namespace must be on (default is on) as well.

```
    SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();
    parser->setFeature(XMLUni::fgSAX2CoreNameSpaces, true);
    parser->setFeature(XMLUni::fgXercesSchema, true);
    parser->parse(xmlFile);
```

Review the sample file, 'samples/data/personal-schema.xml' and 'samples/data/personal.xsd' for an example of an XML Schema grammar.

Associating Schema Grammar with instance document

Schema grammars can be associated with instance documents in two ways.

Specifying Schema Grammar through method calls:

An application developer may associate schemas with instance documents through methods `setExternalSchemaLocation` if they use namespaces, and `setExternalNoNamespaceSchemaLocation` otherwise. (For SAX2XMLReader, use the properties: "http://apache.org/xml/properties/schema/external-schemaLocation" and "http://apache.org/xml/properties/schema/external-noNamespaceSchemaLocation")

Here is an example with no target namespace:

```

    // Instantiate the DOM parser.
    DOMParser parser;
    parser.setDoNamespaces(true);
    parser.setDoSchema(true);
    parser.setExternalNoNamespaceSchemaLocation("personal.xsd");
    parser.parse("test.xml");

    // Instantiate the SAX2 XMLReader.
    SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();
    XMLCh* propertyValue = XMLString::transcode("personal.xsd");
    ArrayJanitor<XMLCh> janValue(propertyValue);

    parser->setProperty(
           XMLUni::fgXercesSchemaExternalNoNameSpaceSchemaLocation,
           propertyValue);
    parser.parse("test.xml");
```

Here is an example with a target namespace. Note that it is an error to specify a different namespace in `setExternalSchemaLocation` than the target namespace defined in the Schema.

```
    // Instantiate the DOM parser.
    DOMParser parser;
    parser.setDoNamespaces(true);
    parser.setDoSchema(true);
    parser.setExternalSchemaLocation("http://my.com personal.xsd http://my2.com
   test2.xsd");
    parser.parse("test.xml");

    // Instantiate the SAX2 XMLReader.
    SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();
    XMLCh* propertyValue = XMLString::transcode("http://my.com personal.xsd
   http://my2.com test2.xsd");
    ArrayJanitor<XMLCh> janValue(propertyValue);

    parser->setProperty(
           XMLCh XMLUni::fgXercesSchemaExternalSchemaLocation,
           propertyValue);
    parser.parse("test.xml");
```

Specifying Schema Grammar through attributes in the instance document:
If schema grammar was not specified externally through methods, then each instance document that uses XML Schema grammars must specify the location of the grammars it uses by using an xsi:schemaLocation attribute if they use namespaces, and xsi:noNamespaceSchemaLocation attribute otherwise.

Here is an example with no target namespace:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <personnel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:noNamespaceSchemaLocation='personal.xsd'>
    ...
    </personnel>
```

Here is an example with a target namespace. Note that it is an error to specify a different namespace in xsi:schemaLocation attribute than the target namespace defined in the Schema.

```
 <?xml version="1.0" encoding="UTF-8"?>
 <personnel xmlns="http://my.com"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="http://my.com personal.xsd http://my2.com
test2.xsd">
 ...
 </personnel>
```

35 Programming Guide

Version Macro

Xerces-C++ has defined a numeric preprocessor macro, _XERCES_VERSION, for users to introduce into their code to perform conditional compilation where the version of Xerces is detected in order to enable or disable version specific capabilities. For example,

```
    #if _XERCES_VERSION >= 20304
      // code specific to Xerces-C++ version 2.3.4
    #else
      // old code here...
    #endif
```

The minor and revision (patch level) numbers have two digits of resolution which means that '3' becomes '03' and '4' becomes '04' in this example.

There are also other string macro, or constants to represent the Xerces-C++ version. Please refer to the header xercesc/util/XercesVersion.hpp for further details.

Schema Support

Xerces-C++ contains an implementation of the W3C XML Schema Language. See the Schema page for details.

Progressive Parsing

In addition to using the *parse()* method to parse an XML File. You can use the other two parsing methods, *parseFirst()* and *parseNext()* to do 'progressive parsing', so that you don't have to depend upon throwing an exception to terminate the parsing operation.

Calling parseFirst() will cause the DTD (both internal and external subsets), and any pre-content, i.e. everything up to but not including the root element, to be parsed. Subsequent calls to parseNext() will cause one more pieces of markup to be parsed, and spit out from the core scanning code to the parser (and hence either on to you if using SAX or into the DOM tree if using DOM).

You can quit the parse any time by just not calling parseNext() anymore and breaking out of the loop. When you call parseNext() and the end of the root element is the next piece of markup, the parser will continue on to the end of the file and return false, to let you know that the parse is done. So a typical progressive parse loop will look like this:

```
    // Create a progressive scan token
    XMLPScanToken token;
    if (!parser.parseFirst(xmlFile, token))
    {
      cerr << "scanFirst() failed\n" << endl;
      return 1;
    }

    //
    // We started ok, so lets call scanNext()
    // until we find what we want or hit the end.
    //
    bool gotMore = true;
    while (gotMore && !handler.getDone())
      gotMore = parser.parseNext(token);
```

In this case, our event handler object (named 'handler' surprisingly enough) is watching form some criteria and will return a status from its getDone() method. Since the handler sees the SAX events coming out of the SAXParser, it can tell when it finds what it wants. So we loop until we get no more data or our handler indicates that it saw what it wanted to see.

When doing non-progressive parses, the parser can easily know when the parse is complete and insure that any used resources are cleaned up. Even in the case of a fatal parsing error, it can clean up all per-parse resources. However, when progressive parsing is done, the client code doing the parse loop might choose to stop the parse before the end of the primary file is reached. In such cases, the parser will not know that the parse has ended, so any resources will not be reclaimed until the parser is destroyed or another parse is started.

This might not seem like such a bad thing; however, in this case, the files and sockets which were opened in order to parse the referenced XML entities will remain open. This could cause serious problems. Therefore, you should destroy the parser instance in such cases, or restart another parse immediately. In a future release, a reset method will be provided to do this more cleanly.

Also note that you must create a scan token and pass it back in on each call. This insures that things don't get done out of sequence. When you call parseFirst() or parse(), any previous scan tokens are invalidated and will cause an error if used again. This prevents incorrect mixed use of the two different parsing schemes or incorrect calls to parseNext().

Preparsing Grammar and Grammar Caching

Xerces-C++ 2.5.0 provides a new function to pre-parse the grammar so that users can check for any syntax or error before using the grammar. Users can also optionally cache these pre-parsed grammars for later use during actual parsing.

Here is an example:

```
    XercesDOMParser parser;

    // enbale schema processing
    parser.setDoSchema(true);
    parser.setDONamespaces(true);

    // Let's preparse the schema grammar (.xsd) and cache it.
    Grammar* grammar = parser.loadGrammar(xmlFile, Grammar::SchemaGrammarType,
  true);
```

Besides caching pre-parsed schema grammars, users can also cache any grammars encountered during an xml document parse.

Here is an example:

```
    SAXParser parser;

    // Enable grammar caching by setting cacheGrammarFromParse to true.
    // The parser will cache any encountered grammars if it does not
    // exist in the pool.
    // If the grammar is DTD, no internal subset is allowed.
    parser.cacheGrammarFromParse(true);

    // Let's parse our xml file (DTD grammar)
    parser.parse(xmlFile);

    // We can get the grammar where the root element was declared
    // by calling the parser's method getRootGrammar;
    // Note: The parser owns the grammar, and the user should not delete it.
    Grammar* grammar = parser.getRootGrammar();
```

We can use any previously cached grammars when parsing new xml documents. Here are some examples on how to use those cached grammars:

```
    /**
      * Caching and reusing XML Schema (.xsd) grammar
      * Parse an XML document and cache its grammar set. Then,  use the cached
      * grammar set in subsequent parses.
      */

    XercesDOMParser parser;

    // Enable schema processing
    parser.setDoSchema(true);
    parser.setDoNamespaces(true);

    // Enable grammar caching
    parser.cacheGrammarFromParse(true);

    // Let's parse the XML document. The parser will cache any grammars encounterd.
    parser.parse(xmlFile);

    // No need to enable re-use by setting useCachedGrammarInParse to true. It is
    // automatically enabled with grammar caching.
    for (int i=0; i< 3; i++)
        parser.parse(xmlFile);

    // This will flush the grammar pool
    parser.resetCachedGrammarPool();
```

```
    /**
      * Caching and reusing DTD grammar
      * Preparse a grammar and cache it in the pool. Then, we use the cached grammar
      * when parsing XML documents.
      */

    SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();

    // Load grammar and cache it
    parser->loadGrammar(dtdFile, Grammar::DTDGrammarType, true);

    // enable grammar reuse
    parser->setFeature(XMLUni::fgXercesUseCachedGrammarInParse, true);

    // Parse xml files
    parser->parse(xmlFile1);
    parser->parse(xmlFile2);
```

There are some limitations about caching and using cached grammars:

- When caching/reusing DTD grammars, no internal subset is allowed.
- When preparsing grammars with caching option enabled, if a grammar, in the result set, already exists in the pool (same NS for schema or same system id for DTD), the entire set will not be cached.
- When parsing an XML document with the grammar caching option enabled, the reuse option is also automatically enabled. We will only parse a grammar if it does not exist in the pool.

Loadable Message Text

The Xerces-C++ supports loadable message text. Although the current drop just supports English, it is capable to support other languages. Anyone interested in contributing any translations should contact us. This would be an extremely useful service.

In order to support the local message loading services, all the error messages are captured in an XML file in the src/xercesc/NLS/ directory. There is a simple program, in the tools/NLS/Xlat/ directory, which can spit out that text in various formats. It currently supports a simple 'in memory' format (i.e. an array of strings), the Win32 resource format, and the message catalog format. The 'in memory' format is intended for very simple installations or for use when porting to a new platform (since you can use it until you can get your own local message loading support done.)

In the src/xercesc/util/ directory, there is an XMLMsgLoader class. This is an abstraction from which any number of message loading services can be derived. Your platform driver file can create whichever type of message loader it wants to use on that platform. Xerces-C++ currently has versions for the in memory format, the Win32 resource format, the message catalog format, and ICU message loader. Some of the platforms can support multiple message loaders, in which case a #define token is used to control which one is used. You can set this in your build projects to control the message loader type used.

Pluggable Transcoders

Xerces-C++ also supports pluggable transcoding services. The XMLTransService class is an abstract

API that can be derived from, to support any desired transcoding service. XMLTranscoder is the abstract API for a particular instance of a transcoder for a particular encoding. The platform driver file decides what specific type of transcoder to use, which allows each platform to use its native transcoding services, or the ICU service if desired.

Implementations are provided for Win32 native services, ICU services, and the *iconv* services available on many Unix platforms. The Win32 version only provides native code page services, so it can only handle XML code in the intrinsic encodings ASCII, UTF-8, UTF-16 (Big/Small Endian), UCS4 (Big/Small Endian), EBCDIC code pages IBM037, IBM1047 and IBM1140 encodings, ISO-8859-1 (aka Latin1) and Windows-1252. The ICU version provides all of the encodings that ICU supports. The *iconv* version will support the encodings supported by the local system. You can use transcoders we provide or create your own if you feel ours are insufficient in some way, or if your platform requires an implementation that Xerces-C++ does not provide.

Porting Guidelines

All platform dependent code in Xerces has been isolated to a couple of files, which should ease the porting effort. Here are the basic steps that should be followed to port Xerces.

1. The directory `src/xercesc/util/Platforms` contains the platform sensitive files while `src/xercesc/util/Compilers` contains all development environment sensitive files. Each operating system has a file of its own and each development environment has another one of its own too.

   As an example, the Win32 platform as a `Win32Defs.hpp` file and the Visual C++ environment has a `VCPPDefs.hpp` file. These files set up certain define tokens, typedefs, constants, etc... that will drive the rest of the code to do the right thing for that platform and development environment. AIX/CSet have their own `AIXDefs.hpp` and `CSetDefs.hpp` files, and so on. You should create new versions of these files for your platform and environment and follow the comments in them to set up your own. Probably the comments in the Win32 and Visual C++ will be the best to follow, since that is where the main development is done.
2. Next, edit the file `XercesDefs.hpp`, which is where all of the fundamental stuff comes into the system. You will see conditional sections in there where the above per-platform and per-environment headers are brought in. Add the new ones for your platform under the appropriate conditionals.
3. Now edit `AutoSense.hpp`. Here we set canonical Xerces internal `#define` tokens which indicate the platform and compiler. These definitions are based on known platform and compiler defines.

   `AutoSense.hpp` is included in `XercesDefs.hpp` and the canonical platform and compiler settings thus defined will make the particular platform and compiler headers to be the included at compilation.

   It might be a little tricky to decipher this file so be careful. If you are using say another compiler on Win32, probably it will use similar tokens so that the platform will get picked up already using what is already there.
4. Once this is done, you will then need to implement a version of the *platform utilities* for your platform. Each operating system has a file which implements some methods of the XMLPlatformUtils class, specific to that operating system. These are not terribly complex, so it should not be a lot of work. The Win32 version is called `Win32PlatformUtils.cpp`, the AIX version is `AIXPlatformUtils.cpp` and so on. Create one for your platform, with the correct name, and empty out all of the implementation so that just the empty shells of the methods are there (with dummy returns where needed to make the compiler happy.) Once you've done that, you can

start to get it to build without any real implementation.

5. Once you have the system building, then start implementing your own platform utilities methods. Follow the comments in the Win32 version as to what they do, the comments will be improved in subsequent versions, but they should be fairly obvious now. Once you have these implementations done, you should be able to start debugging the system using the demo programs.

Other concerns are:

- Does ICU compile on your platform? If not, then you'll need to create a transcoder implementation that uses your local transcoding services. The iconv transcoder should work for you, though perhaps with some modifications.
- What message loader will you use? To get started, you can use the "in memory" one, which is very simple and easy. Then, once you get going, you may want to adapt the message catalog message loader, or write one of your own that uses local services.
- What should I define XMLCh to be? Please refer to What should I define XMLCh to be? for further details.

That is the work required in a nutshell!

Using C++ Namespace

Xerces-C++ 2.5.0 now supports C++ Namespace.

The macro `XERCES_HAS_CPP_NAMESPACE` is defined in each Compiler Definition file if C++ Namespace is supported.

For example in header `xercesc/util/Compilers/GCCDefs.hpp`, the C++ Namespace is enabled:

```
// ---------------------------------------------------------------------------
// Indicate that we support C++ namespace
// Do not define it if the compile cannot handle C++ namespace
// ---------------------------------------------------------------------------
#define XERCES_HAS_CPP_NAMESPACE
```

If C++ Namespace support is ENABLED (all the binary distributions of Xerces-C++ 2.5.0 are built with C++ Namespace enabled), users' applications must namespace qualified all the Xerces-C++ classes/data/variables with `"xercesc::"` or add the `"using namespace xercesc"` clause. Users also need to ensure all forward declarations are properly qualified or scope. For example

```
#include <stdio.h>
#include <stdlib.h>
#include <xercesc/sax/HandlerBase.hpp>

// indicate using Xerces-C++ namespace in general
using namespace xercesc;

// need to properly scope any forward declarations
namespace xercesc {
class AttributeList;
}

// or namespace qualifier the forward declarations
```

```
    class xercesc::ErrorHandler;

    class MySAXHandlers : public HandlerBase
    {
    public:
        // -----------------------------------------------------------------------
        //  Handlers for the SAX DocumentHandler interface
        // -----------------------------------------------------------------------
        void startElement(const XMLCh* const name, AttributeList& attributes);
        void characters(const XMLCh* const chars, const unsigned int length);
    :
    :
    };
```

Header `"xercesc/util/XercesDefs.hpp"` has defined the following macros

```
    #if defined(XERCES_HAS_CPP_NAMESPACE)
        #define XERCES_CPP_NAMESPACE_BEGIN    namespace xercesc_2_5 {
        #define XERCES_CPP_NAMESPACE_END    }
        #define XERCES_CPP_NAMESPACE_USE    using namespace xercesc_2_5;
        #define XERCES_CPP_NAMESPACE_QUALIFIER    xercesc_2_5::

        namespace xercesc_2_5 { }
        namespace xercesc = xercesc_2_5;
    #else
        #define XERCES_CPP_NAMESPACE_BEGIN
        #define XERCES_CPP_NAMESPACE_END
        #define XERCES_CPP_NAMESPACE_USE
        #define XERCES_CPP_NAMESPACE_QUALIFIER
    #endif
```

Users can also make use of these pre-defined macro in their applications. For example:

```
    #include <stdio.h>
    #include <stdlib.h>
    #include <xercesc/sax/HandlerBase.hpp>

    // indicate using Xerces-C++ namespace in general
    XERCES_CPP_NAMESPACE_USE

    // need to properly scope any forward declarations
    XERCES_CPP_NAMESPACE_BEGIN
    class AttributeList;
    XERCES_CPP_NAMESPACE_END

    // or namespace qualifier the forward declarations
    class XERCES_CPP_NAMESPACE_QUALIFIER ErrorHandler;

    class MySAXHandlers : public HandlerBase
```

```
{
public:
    // -----------------------------------------------------------------------
    //  Handlers for the SAX DocumentHandler interface
    // -----------------------------------------------------------------------
    void startElement(const XMLCh* const name, AttributeList& attributes);
    void characters(const XMLCh* const chars, const unsigned int length);
:
:
};
```

NOTE: `"namespace xercesc"` and `"namespace xercesc_2_5"` are equivalent in this release.

For those users who want to selectively pick which version of API to use, they can do something like this:

```
#if _XERCES_VERSION == 20300
  // code specific to Xerces-C++ version 2.3.0
  new xercesc_2_3::SAXParser();
#elif _XERCES_VERSION == 20200
  // code specific to Xerces-C++ version 2.2.0
  new xercesc_2_2::SAXParser();
#else
  // old code here...
  new SAXParser();
#endif
```

But for those who just want to call the latest API, then they should use `xercesc::` or the macro `XERCES_CPP_NAMESPACE_QUALIFIER` for source compatibility:

```
//use the current namespace, xercesc
new xercesc::SAXParser();

//or use the macro
new XERCES_CPP_NAMESPACE_QUALIFIER SAXParser();
```

`xercesc` is a generic namespace name which will be assigned to `xercesc_YY_ZZ` in each specific release, where "YY" is the Major Release Number and "ZZ" is the Minor Version Number.

Specify Locale for Message Loader

The Xerces-C++ has implemented mechanism to support NLS, though the current drop has only English version message file, it is capable to support other languages once the translated version of the target language is available.

Application can specify the locale for the message loader in their very first invocation to XMLPlatformUtils::Initialize() by supplying a parameter for the target locale intended. The defaul locale is "en_US".

```


...
```

```
        // Initialize the parser system
        try
        {
             XMLPlatformUtils::Initialize("fr_FR");
        }

        catch ()
        {
        }
    ..
```

Specify Location for Message Loader

The Xerces-C++ searches for message files at the default message directory, $XERCESCROOT/msg.

Application can specify an alternative location for the message files in their very first invocation to XMLPlatformUtils::Initialize() by supplying a parameter for the alternative location intended.

```


    ...
        // Initialize the parser system
        try
        {
             XMLPlatformUtils::Initialize("en_US",
   "/usr/application_root/msg_home");
        }

        catch ()
        {
        }
    ..
```

Pluggable Panic Handler

The Xerces-C++ reports, through the method panic(), any panic encountered, to the panic handler installed, which in turn takes whatever action appropriate, to handle the panic.

The Xerces-C++ allows application plugging a customized panic handler (class implementing the interface PanicHandler), in its very first invocation to XMLPlatformUtils::Initialize() by supplying a parameter for the panic handler intended.

In the absence of such a plugged panic handler, Xerces-C++ default panic handler is installed and used, which aborts program whenever a panic is seen.

```


    ...
        // Initialize the parser system
        try
        {
             PanicHandler* ph = new MyPanicHandler();
```

```
            XMLPlatformUtils::Initialize("en_US"
                                         , "/usr/application_root/msg_home"
                                         , ph);
        }

        catch ()
        {
        }
    ..
```

Pluggable Memory Manager

Certain applications wish to maintain precise control over memory allocation. This enables them to recover more easily from crashes of individual components, as well as to allocate memory more efficiently than a general-purpose OS-level procedure with no knowledge of the characteristics of the program making the requests for memory. As of Xerces-C 2.3.0 this is supported via the Pluggable Memory Handler.

Users that have no particular memory management requirements (beyond that components don't leak memory or attempt to read from or write to areas of memory they haven't been assigned), should notice no behavioural changes in the parser, so long as their code conforms to Xerces-C best practices (e.g., avoids implicit destruction of objects related to the parser after XMLPlatformUtils::Terminate() has been called; see the FAQ entry describing a reason why applications may suddenly start segfaulting with Xerces-C 2.3.0 for details.). Such users can ignore this subsection and continue using the parser as they always had.

Users who wish to implement their own MemoryManager, an interface found in xercesc/framework/MemoryManager.hpp, need implement only two methods:

```
    // This method allocates requested memory.
    // the parameter is the requested memory size
    // A pointer to the allocated memory is returned.
    virtual void* allocate(size_t size) = 0;

    // This method deallocates memory
    // The parameter is a pointer to the allocated memory to be deleted
    virtual void deallocate(void* p) = 0;
```

To maximize the amount of flexibility that applications have in terms of controlling memory allocation, a MemoryManager instance may be set as part of the call to XMLPlatformUtils::Initialize() to allow for static initialization to be done with the given MemoryHandler; a (possibly different) MemoryManager may be passed in to the constructors of all Xerces parser objects as well, and all dynamic allocations within the parsers will make use of this object. Assuming that MyMemoryHandler is a class that implements the MemoryManager interface, here is a bit of pseudocode which illustrates these ideas:

```
    MyMemoryHandler *mm_for_statics = new MyMemoryHandler();
    MyMemoryHandler *mm_for_particular_parser = new MyMemoryManager();

    // initialize the parser information; try/catch
    // removed for brevity
    XMLPlatformUtils::Initialize(XMLUni::fgXercescDefaultLocale, 0,0,
            mm_for_statics);

    // create a parser object
    XercesDOMParser *parser = new
            XercesDomParser(mm_for_particular_parser);

    // ...
    delete parser;
    XMLPlatformUtils::Terminate();
```

Notice that, to maintain backward compatibility, the MemoryManager parameter is positioned last in the list of parameters to XMLPlatformUtils::Initialize(); this means that all other parameters must be specified with their defaults as found in Xerces code if all other aspects of standard behaviour are to be preserved.

If a user provides a MemoryManager object to the parser, then the user owns that object. It is also important to note that Xerces default implementation simply uses the global new and delete.

Finally, there are two platform/compiler-related limitations of our memory handling facilities that certain users will need to be aware of:

- The compiler shipped with HPUX 11 does not understand "placement" delete operators. These versions of delete have the same signature as our "placement" new operators but will only be invoked when an exception is thrown during the construction of an object. Since the HP compiler does not permit delete to be overridden twice within a class, we cannot provide a placement delete; hence, in the few cases in which an exception may be thrown during object construction by Xerces, destructors of objects created during that construction will not be called.
- There is a bug in versions of GCC older than 2.96 which makes it impossible to have the pluggable memory manager create elements in the `RefHash3KeysIdPool` template hashtable. Therefore, on this compiler, we must use global new for this purpose. These elements will be properly destroyed under this compiler; the limitation is that, since the pluggable memory manager cannot be used, these particular elements will not be destroyed if the user destroys their memory manager directly. Note that this hashtable is not used that often in Xerces.

Managing Security Vulnerabilities

The purpose of the SecurityManager class is to permit applications a means to have the parser reject documents whose processing would otherwise consume large amounts of system resources. Malicious use of such documents could be used to launch a denial-of-service attack against a system running the parser. Initially, the SecurityManager only knows about attacks that can result from exponential entity expansion; this is the only known attack that involves processing a single XML document. Other, simlar attacks can be launched if arbitrary schemas may be parsed; there already exist means (via use of the EntityResolver interface) by which applications can deny processing of untrusted schemas. In future, the SecurityManager will be expanded to take these other exploits into account.

The SecurityManager class is very simple: It will contain getters and setters corresponding to each known variety of exploit. These will reflect limits that the application may impose on the parser with respect to the processing of various XML constructs. When an instance of SecurityManager is instantiated, default values for these limits will be provided that should suit most applications.

By default, Xerces-C is a wholly conformant XML parser; that is, no security-related considerations will be observed by default. An application must set an instance of the SecurityManager class on a Xerces parser in order to make that parser behave in a security-conscious manner. i.e.:

```
SAXParser *myParser = new SAXParser();
SecurityManager *myManager = new SecurityManager();
myManager->setEntityExpansionLimit(100000); // larger than default
myParser->setSecurityManager(myManager);
// ... use the parser
```

Note that SecurityManager instances may be set on all kinds of Xerces parsers; please see the documentation for the individual parsers for details.

Note also that the application always owns the SecurityManager instance. The default SecurityManager that Xerces provides is not thread-safe; although it only uses primitive operations at the moment, users may need to extend the class with a thread-safe implementation on some platforms.

Use Specific Scanner

For performance and modularity, the Xerces-C++ has implemented a mechanism to allow users to specify the scanner to use when scanning an XML document. Such mechanism will enable the creation of special purpose scanners that can be easily plugged in.

Xerces-C++ supports the following scanners:

WFXMLScanner

The WFXMLScanner is a non-validating scanner which performs well-formedness check only. It does not do any DTD/XMLSchema processing. If the XML document contains a DOCTYPE, it will be silently ignored (i.e. no warning message is issued). Similiarly, any schema specific attributes (e.g. schemaLocation), will be treated as normal element attributes. Setting grammar specific features/properties will have no effect on its behavior (e.g. setLoadExternalDTD(true) is ignored).

```
// Create a DOM parser
XercesDOMParser parser;

// Specify scanner name
parser.useScanner(XMLUni::fgWFXMLScanner);

// Specify other parser features, e.g.
parser.setDoNamespaces(true);
```

DGXMLScanner

The DGXMLScanner handles XML documents with DOCTYPE information. It does not do any XMLSchema processing, which means that any schema specific attributes (e.g. schemaLocation), will be treated as normal element attributes. Setting schema grammar specific features/properties will have no effect on its behavior (e.g. setDoSchema(true) is ignored).

```
// Create a SAX parser
SAXParser parser;
```

```
    // Specify scanner name
    parser.useScanner(XMLUni::fgDGXMLScanner);

    // Specify other parser features, e.g.
    parser.setLoadExternalDTD(true);
```

SGXMLScanner
The SGXMLScanner handles XML documents with XML schema grammar information. If the XML document contains a DOCTYPE, it will be ignored. Namespace and schema processing features are on by default, and setting them to off has not effect.

```
    // Create a SAX2 parser
    SAX2XMLReader* parser = XMLReaderFactory::createXMLReader();

    // Specify scanner name
    parser->setProperty(XMLUni::fgXercesScannerName, (void
   *)XMLUni::fgSGXMLScanner);

    // Specify other parser features, e.g.
    parser->setFeature(XMLUni::fgXercesSchemaFullChecking, false);
```

IGXMLScanner
The IGXMLScanner is an integrated scanner and handles XML documents with DTD and/or XML schema grammar. This is the default scanner used by the various parsers if no scanner is specified.

```
    // Create a DOMBuilder parser
    DOMBuilder *parser =
   createDOMBuilder(DOMImplementationLS::MODE_SYNCHRONOUS, 0);

    // Specify scanner name - This is optional as IGXMLScanner is the default
    parser->setProperty(XMLUni::fgXercesScannerName, (void
   *)XMLUni::fgIGXMLScanner);

    // Specify other parser features, e.g.
    parser->setFeature(XMLUni::fgDOMNamespaces, doNamespaces);
    parser->setFeature(XMLUni::fgXercesSchema, doSchema);
```

36 Migration

Migration Archive

For migration information to Xerces-C++ 2.4.0 or earlier, please refer to Migration Archive.

Migrating from Xerces-C++ 2.4.0 to Xerces-C++ 2.5.0

The following section is a discussion of the technical differences between Xerces-C++ 2.4.0 code base and the Xerces-C++ 2.5.0.

Topics discussed are:

- New features in Xerces-C++ 2.5.0
- Public API Changes
  - New Public API
  - Modified Public API
  - Deprecated/Removed Public API

New features in Xerces-C++ 2.5.0

- Fix duplicate attribute detection when namespaces are disabled
- Stricter use of static memory manager for static data only
- PSVI bug fix and enhencement
- ThreadTest with grammar caching
- Re-pluggable Panic Handler
- Enhenced mutex creation to impove thread safety
- Intrinsic transcoding support for 390.
- Canonical Representation Support
- New sample SCMPrint
- New sample PSVIWriter
- New test XSerializerTest

Public API Changes

The following lists the public API changes between the Xerces-C++ 2.4.0; and the Xerces-C++ 2.5.0 releases of the parser.

New Public API

-

Modified Public API

-

Deprecated/Removed Public API

-

37 Migration Archive

Migrating to earlier Releases

Migrating from Xerces-C++ 2.3.0 to Xerces-C++ 2.4.0

The following section is a discussion of the technical differences between Xerces-C++ 2.3.0 code base and the Xerces-C++ 2.4.0.

Topics discussed are:

New features in Xerces-C++ 2.4.0

- PSVI
- Performance enhancement
- Stateless Grammar
- Grammar Serialization/Deserialiation

Public API Changes

The following lists the public API changes between the Xerces-C++ 2.3.0; and the Xerces-C++ 2.4.0 releases of the parser.

New Public API

- PSVI related
- Grammar serialization/deserialization related

Modified Public API

.
Deprecated/Removed Public API

- XMLAttDef: getProvided, getDOMTypeInfoUri, getDOMTypeInfoName, setProvided
- XMLAttDefList: hasMoreElements, nextElement, Reset
- DTDAttDefList: hasMoreElements, nextElement, Reset
- SchemaAttDefList: hasMoreElements, nextElement, Reset
- XMLElementDecl: LookupOpts
- XMLNumber family: toString
- ENTITYDatatypeValidator: setEntityDeclPool
- IDDatatypeValidator: setIDRefList
- IDREFDatatypeValidator: setIDRefList
- GeneralAttributeCheck: setIDRefList
- SchemaGrammar: getIDRefList
- SchemaElementDecl: all non thread safe methods
- SchemaAttDef: getters
- DTDGrammar: getRootElemId

Migrating from Xerces-C++ 2.2.0 to Xerces-C++ 2.3.0

The following section is a discussion of the technical differences between Xerces-C++ 2.2.0 code base and the Xerces-C++ 2.3.0.

Topics discussed are:

- New features in Xerces-C++ 2.3.0
- Public API Changes
  - New Public API
  - Modified Public API
  - Deprecated/Removed Public API

New features in Xerces-C++ 2.3.0

- Experimental Implementation of Namespaces in XML 1.1
- Experimental Implementation of XML 1.1: in DOMWriter
- More Schema 1.0 Errata Implementation
- More DOM L3 Core Support
  - DOMConfiguration
  - Document Normalization
- Plugable Memory Manager
- Plugable Security Manager
- Plugable Panic Handler
- Logical Path Resolution

Public API Changes

The following lists the public API changes between the Xerces-C++ 2.2.0; and the Xerces-C++ 2.3.0 releases of the parser.

New Public API

- To support additional DOM L3 functions, the following are added:
- DOMDocument: getDOMConfiguration
- DOMConfiguration class for document normalization.

Modified Public API

.

Deprecated/Removed Public API

- DOMDocument canSetNormalizationFeature, setNormalizationFeature, getNormalizationFeature, getErrorHandler, setErrorHandler removed

Migrating from Xerces-C++ 2.1.0 to Xerces-C++ 2.2.0

The following section is a discussion of the technical differences between Xerces-C++ 2.1.0 code base and the Xerces-C++ 2.2.0.

Topics discussed are:

- New features in Xerces-C++ 2.2.0
- Using C++ Namespace
- Public API Changes
  - New Public API
  - Modified Public API
  - Deprecated/Removed Public API

New features in Xerces-C++ 2.2.0

- C++ Namespace Support
- Schema 1.0 Errata Implementation
- Experimental Implementation of XML 1.1
- More DOM L3 Core Support:
  - DOMNode: baseURI
  - DOMAttr: isId, getTypeInfo
  - DOMElement: setIdAttribute, setIdAttributeNS, setIdAttributeNode, getTypeInfo
- DOM Message: make use of the non-standard extension DOMImplementation::loadDOMExceptionMsg to load the default error text message for the correspond Exception Code.
- New feature XMLPlatformUtils::Initialize(const char* const locale) to set the locale for message loader. See Specify locale for Message Loader for details
- Support Build with ICU Message Loader, or Message Catalog Message Loader
- RPM for Linux
- 390: Uniconv390 support
- 390: support record-oriented MVS datasets with the DOM Level 3 serialization APIs
- Support for Linux/390
- Performance: Break Scanner for different functionalities and many other performance improvement
- New feature, "http://apache.org/xml/features/dom/byte-order-mark", allows user to enable DOMWriter to write Byte-Order-Mark in the output XML stream, See Xercesc Feature: Byte Order Mark for details

Using C++ Namespace

Xerces-C++ 2.2.0 now supports C++ Namespace. All Xerces-C++ classes/data/variables are defined in the `"namespace xercesc"` if C++ Namespace support is ENABLED.

All the binary distributions of Xerces-C++ 2.2.0 are now built with C++ Namespace enabled. Therefore users' applications that links with the distributed binary packages must namespace qualified all the Xerces-C++ classes/data/variables with `"xercesc::"` or add the `"using namespace xercesc"` clause.

See the Programming Guide Using C++ Namespace for more details.

Public API Changes

The following lists the public API changes between the Xerces-C++ 2.1.0; and the Xerces-C++ 2.2.0 releases of the parser.

New Public API

- To support additional DOM L3 functions, the following are added:
  - DOMAttr: isId, getTypeInfo
  - DOMElement: setIdAttribute, setIdAttributeNS, setIdAttributeNode, getTypeInfo
  - Added DOMTypeInfo class for getTypeInfo class in DOMElement and DOMAttr
  - Added getDOMTypeInfoUri, getDOMTypeInfoName to XMLAttDef and XMLElementDecl for use in building DOMTypeInfo
- Added a non-standard extension DOMImplementation::loadDOMExceptionMsg to load the default error message for the corresponding DOMException code.
- XMLAttr: Added a constructor and a set method to allow creating/setting of XMLAttr using a rawname.
- Added XMLUri::getUriText to return the URI as a string specification.
- Add XMLString::fixURI to transform an absolute path filename to standard URI form.
- Added XMLString::equals for faster string comparison.
- To allow users to tell the parser to force standard uri conformance, the following are added:
  - XercesDOMParser/DOMParser/SAXParser: get/setStandardUriConformant
  - and DOMBuilder/SAX2XMLReader will recognize the feature http://apache.org/xml/features/standard-uri-conformant
- Add XMLURL::hasInvalidChar() to indicate if the URL has invalid char as per RFC standard
- To allow users to enable/disable src offset calculation, the following are added:
  - XercesDOMParser/DOMParser/SAXParser: get/setCalculateSrcOfs
  - and DOMBuilder/SAX2XMLReader will recognize the feature http://apache.org/xml/features/calculate-src-ofst
- To allow users to select the scanner when scanning XML documents, the following are added:
  - XercesDOMParser/DOMParser/SAXParser: useScanner
  - and DOMBuilder/SAX2XMLReader will recognize the property http://apache.org/xml/properties/scannerName
- Added getSrcOffset to XercesDOMParser/DOMParser/SAXParser/DOMBuilder/SAX2XMLReader to allow users to get the current src offset within the input source.

Modified Public API

- The following DOM functions are being added a const modifier.
  - DOMImplementation::hasFeature
  - DOMNode: isSameNode, isEqualNode, compareTreePosition
- XMLPlatformUtils::Initialize() takes a parameter specifying locale for message loader, with default value "en_US".
- To fix [Bug 13641], the QName copy constructor is corrected to take a reference as parameter, i.e. QName(const QName& qname).
- To fix [Bug 12232], the QName operator== has been added a const modified.
- Move XMLUri copy constructor and operator = as public.
- Move XMLUri::isURIString as public.
- For validation purpose, added two more default parameters to XMLValidator::validateAttrValue.
- To fix [Bug 15802], the getURIText of DOMParser/XercesDOMParser/SAXParser/SAX2XMLReader are being added a const modifier.

Deprecated/Removed Public API

- No Deprecated Public API in this release.

Migrating from Xerces-C++ 2.0.0 to Xerces-C++ 2.1.0

The following section is a discussion of the technical differences between Xerces-C++ 2.0.0 code base and the Xerces-C++ 2.1.0.

Topics discussed are:

- New features in Xerces-C++ 2.1.0
- Public API Changes
  - New Public API
  - Modified Public API
  - Deprecated/Removed Public API

New features in Xerces-C++ 2.1.0

- 64 bit binaries distribution on Windows IA64 and Linux IA64
- Support for Cygwin environment
- DOM Level 3 DOMNode: compareTreePosition, lookupNamespaceURI, lookupNamespacePrefix and isDefaultNamespace
- plus many more bug fixes

Public API Changes

The following lists the public API changes between the Xerces-C++ 2.0.0; and the Xerces-C++ 2.1.0 releases of the parser.

New Public API

- To fix bug 7087, XMLEnumerator is added a virtual destructor.
- To fix bug 11448, XMLNotationDecl::get/setBaseURI, and XMLEntityDecl::get/setBaseURI are added.

Modified Public API

- DOMNodeList: item, and getLength have been added a const modifier.
- DOMNode: lookupNamespacePrefix, isDefaultNamespace, and lookupNamespaceURI have been added a const modifier.

Deprecated/Removed Public API

- No Deprecated Public API in this release.

Migrating from Xerces-C++ 1.7.0 to Xerces-C++ 2.0.0

The following section is a discussion of the technical differences between Xerces-C++ 1.7.0 code base and the Xerces-C++ 2.0.0.

Topics discussed are:

- New features in Xerces-C++ 2.0.0
- Unix Library Name change
- DOM Reorganization
- Reuse Grammar becomes Grammar Caching
- Public API Changes
  - New Public API
  - Modified Public API

- Deprecated/Removed Public API

New features in Xerces-C++ 2.0.0

- 64 bit binaries distribution
- Follow Unix Shared Library Naming Convention
- Apache Recommended DOM C++ Binding
- Experimental DOM Level 3 subset support, including DOMWriter and DOMBuilder
- Grammar preparsing and Grammar caching
- Optionally ignore loading of external DTD
- Project files for Microsoft Visual C++ .Net
- Codewarrior 8 support
- Option to enable/disable strict IANA encoding name checking
- plus many more bug fixes and performance enhancement

Unix Library Name Change

The Xerces-C++ UNIX Library now follows the Unix Shared Library Naming Convention (libname.so.soname). It is now called:

- AIX
  - libxerces-c25.0.so
  - symbolic link: libxerces-c.so ---- > libxerces-c25.so
  - symbolic link: libxerces-c25.so ---- > libxerces-c25.0.so
- Solaris / Linux
  - libxerces-c.so.25.0
  - symbolic link: libxerces-c.so ---- > libxerces-c.so.25
  - symbolic link: libxerces-c.so.25 ---- > libxerces-c.so.25.0
- HP-UX
  - libxerces-c.sl.25.0
  - symbolic link: libxerces-c.sl ---- > libxerces-c.sl.25
  - symbolic link: libxerces-c.sl.25 ---- > libxerces-c.sl.25.0

DOM Reorganization

1. The old Java-like DOM is now deprecated, and all the associated files, including the headers and DOMParser files are moved to `src/xercesc/dom/deprecated`. Users of the old Java-like DOM are required to change all their #include lines to pick up the headers. For example

```
//old code
#include <xercesc/dom/DOM.hpp>
#include <xercesc/dom/DOM_Document.hpp>
#include <xercesc/parsers/DOMParser.hpp>

void test(char* xmlFile) {
    DOMParser parser;
    parser.parse(xmlFile);
    DOM_Document doc = parser.getDocument();
    :
    return;
}
```

should now change to

```
 //new code
 #include <xercesc/dom/deprecated/DOM.hpp>          //<==== change this include
line
 #include <xercesc/dom/deprecated/DOM_Document.hpp> //<==== change this include
line
 #include <xercesc/dom/deprecated/DOMParser.hpp>    //<==== change this include
line

 // the rest is the same
 void test(char* xmlFile) {
     DOMParser parser;
     parser.parse(xmlFile);
     DOM_Document doc = parser.getDocument();
     :
     return;
 }
```

2. The Experimental IDOM is now renamed, and becomes the Apache Recommended DOM C++ Binding. The following changes are made:

- class names are renamed from IDOM_XXXX to DOMXXXX, e.g. IDOM_Document to DOMDocument
- and thus header files are renamed from IDOM_XXXX.hpp to DOMXXXX.hpp and are moved to `src/xercesc/dom`
- the IDOMParser is renamed to XercesDOMParser. And thus the header file is renamed as well
- the rest is the same, see Apache Recommended DOM C++ binding and DOM Programming Guide for more programming information

Users of IDOM are required to change all their #include lines and do a global rename of IDOMParser to XercesDOMParesr, and IDOM_XXXX to DOMXXXX. For example

```
 //old code
 #include <xercesc/idom/IDOM.hpp>
 #include <xercesc/idom/IDOM_Document.hpp>
 #include <xercesc/parsers/IDOMParser.hpp>

 void test(char* xmlFile) {
     IDOMParser parser;
     parser.parse(xmlFile);
     IDOM_Document* doc = parser.getDocument();
     :
     return;
 }
```

should now change to

```
 //new code
 #include <xercesc/dom/DOM.hpp>                      //<==== change this include line
 #include <xercesc/dom/DOMDocument.hpp>              //<==== change this include line
```

```
    #include <xercesc/parsers/XercesDOMParser.hpp>  //<==== change this include line

    void test(char* xmlFile) {
        XercesDOMParser parser;                           //<==== rename the
   IDOMParser
        parser.parse(xmlFile);
        DOMDocument* doc = parser.getDocument();          //<==== rename the
   IDOM_XXXX
        :
        return;
    }
```

Reuse Grammar becomes Grammar Caching

The Xerces-C++ 2.0.0 extends the "Reuse Grammar" support by replacing it with a new feature called "Grammar Caching" which provides more flexibility in reusing grammars. Users who used to do the following:

```


          XercesDOMParser parser;

          // this is the first parse, just usual code as you do normal parse
          // "firstXmlFile" has a grammar (schema or DTD) specified.
          parser.parse(firstXmlFile);

          // this is the second parse, by setting second parameter to true,
          // the parser will reuse the grammar in the last parse
          // (i.e. the one in  "firstXmlFile")
          // to validate the second "anotherXmlFile".  Any grammar that is
          // specified in anotherXmlFile is IGNORED.
          //
          // Note: The anotherXmlFile cannot have any DTD internal subset.
          parser.parse(anotherXmlFile, true);

```

should now use the features cacheGrammarFromParse and useCachedGrammarFromParse:

```

          XercesDOMParser parser;

          // By setting cacheGrammarFromParse to true,
          // the parser will cache any grammars encountered in the
          // follow-on xml files, if not cached already
          parser.cacheGrammarFromParse(true);

          parser.parse(firstXmlFile);

          // By setting useCachedGrammarFromParse to true,
          // the parser will use all the previous cached grammars
          // to validate the follow-on xml files if the cached
```

```
        // grammar matches the one specified in anotherXmlFile.
        //
        // Note: The follow-on xml files cannot have any DTD internal subset.
        parser.useCachedGrammarFromParse(true);

        parser.parse(anotherXmlFile);

        // This will flush the cached grammar pool
        parser.resetCachedGrammarPool();
```

Note there are a number of differences between "Reuse Grammar" and "Grammar Caching"

1. "Reuse Grammar" ignores any grammar that is specified in anotherXmlFile and simply reuse whatever stored in previous parse; while "Grammar Caching" will use the cached grammar only if it matches the one specified in the anotherXmlFile. If not match, then the new grammar is parsed.
2. "Reuse Grammar" can only reuse the grammar from previous parse; while "Grammar Caching" can selectively cache many grammars from different parses and collect them all in a pool indexed by targetNamespace (for Schema) or system id (for DTD).
3. Plus "Grammar Caching" has much more functionalities other than above (like "Pre-parsing Grammar"). Please refer to Preparsing Grammar and Grammar Caching for more programming details.

Public API Changes

The following lists the public API changes between the Xerces-C++ 1.7.0; and the Xerces-C++ 2.0.0 releases of the parser.

New Public API

- To support DOM Level 3, the following are added (see the API documentation page for details).
  · DOMNode functions set/getUserData, isSameNode isEqualNode.
  · DOMDocument functions renameNode, get/setActualEncoding, get/setEncoding, get/setVersion, get/setStandalone, get/setDocumentURI.
  · DOMEntity functions get/setActualEncoding, get/setEncoding, get/setVersion.
  · classes AbstractDOMParser, DOMError, DOMErrorHandler, and DOMLocator.
  · classes DOMUserDataHandler, DOMImplementationRegistry and DOMImplementationSource.
  · classes DOMBuilder, DOMEntityResolver, DOMImplementationLS, DOMInputSource, Wrapper4DOMInputSource and Wrapper4InputSource.
  · classes DOMWriter, DOMWriterFilter, LocalFileFormatTarget, StdOutFormatTarget, and MemBufFormatTarget
- To support DOMWriter, the following PlatformUtils functions are added
  · openFileToWrite, writeBufferToFile
- To have Apache Recommended DOM C++ Binding, the following are added (see Apache Recommended DOM C++ binding).
  · function release() to fix Memory Management problem
  · classes DOMDocumentRange and DOMDocumentTraversal
  · XMLSize_t is used to represent unsigned integral type in DOM
  · IDOM_XXXX classes are renamed to DOMXXXX, and IDOMParser is renamed to XercesDOMParser as described in DOM Reorganization
  · XercesDOMParser::adoptDocument is added so that document can optionally live outside the parser.

- To support optionally load external DTD, the following are added:
  - XercesDOMParser::set/getLoadExternalDTD
  - DOMParser::set/getLoadExternalDTD
  - SAXParser::set/getLoadExternalDTD
  - and SAX2XMLReader will recognize the feature http://apache.org/xml/features/nonvalidating/load-external-dtd
- To support Preparsing Grammar and Grammar Caching, the following are added:
  - XercesDOMParser/DOMParser/SAXParser functions loadGrammar, resetCachedGrammarPool, cacheGrammarFromParse, isCachingGrammarFromParse, useCachedGrammarInParse, isUsingCachedGrammarInParse.
  - SAX2XMLReader functions loadGrammar, resetCachedGrammarPool, and will recognize the features http://apache.org/xml/features/validation/cache-grammarFromParse and http://apache.org/xml/features/validation/use-cachedGrammarInParse.
- To support access to Grammar info, the following are added:
  - XercesDOMParser/DOMParser/SAXParser/SAX2XMLReader functions getRootGrammar, getGrammar, getURIText.
- To support strict IANA encoding name checking, the following are added:
  - class EncodingValidator.
  - PlatformUtils functions strictIANAEncoding, isStrictIANAEncoding.
  - XMLTransService functions strictIANAEncoding, isStrictIANAEncoding.

Modified Public API

- SAXParser::getScanner() is moved from public to protected.
- Grammar::getGrammarType has been added a const modifier.
- Xerces features are renamed from XMLUni::fgSAX2XercesXXXX to XMLUni::fgXercesXXXX so that they can be shared with DOM parser.
- With the new Grammar Caching introduced, the the last parameter "reuseGrammar" in the following API is dropped. Users should now use the "Grammar Caching" feature as described in Reuse Grammar becomes Grammar Caching .
  - (in Parser, SAXParser, DOMParser, and XercesDOMParser)
  - parse(const InputSource& source, const bool reuseGrammar = false);
  - parse(const XMLCh* const systemId, const bool reuseGrammar = false);
  - parse(const char* const systemId, const bool reuseGrammar = false);
  - (in SAXParser, DOMParser, and XercesDOMParser)
  - parseFirst(const InputSource& source, XMLPScanToken& toFill, const bool reuseGrammar = false);
  - parseFirst(const XMLCh* const systemId, XMLPScanToken& toFill, const bool reuseGrammar = false);
  - parseFirst(const char* const systemId, XMLPScanToken& toFill, const bool reuseGrammar = false);

Deprecated/Removed Public API

- The old Java-like DOM is now deprecated as described in DOM Reorganization
- SAX2XMLReader::setValidationConstraint. For consistency, SAX2XMLReader users should set the feature http://apache.org/xml/features/validation-error-as-fatal" instead.
- SAX2XMLReader::setExitOnFirstFatalError. For consistency, SAX2XMLReader users should set the feature "http://apache.org/xml/features/continue-after-fatal-error" instead.
- With the new Grammar Caching introduced, the following features will not be recognized by the SAX2XMLReader:

· http://apache.org/xml/features/validation/reuse-grammar
· http://apache.org/xml/features/validation/reuse-validator

Migrating from Xerces-C++ 1.6.0 to 1.7.0

The following section is a discussion of the technical differences between Xerces-C++ 1.6.0 code base and the Xerces-C++ 1.7.0 code base.

New features in Xerces-C++ 1.7.0

- Support SAX2-ext's DeclHandler.
- Directory sane_include reorganization: add sub-directory 'xercesc' to src / include folder. See "Directory change in Xerces-C++ 1.7.0" below for detail.
- More IDOM test cases - port IDOMMemTest, and merge ThreadTest and IThreadTest.
- Support IconvFBSD in multi-threading environment.
- Use IDOM in schema processing for faster performance.
- Add Project files for BCB6.
- Port to Caldera (SCO) OpenServer.
- Support building with new MacOSURLAccessCF NetAccessor that doesn't require Carbon but can allow Xerces to live solely within CoreServices layer.

Directory change in Xerces-C++ 1.7.0

- A new directory, **src/xercesc** is created to be the new parent directory of all src's direct subdirectories.
- And in the binary package, all the headers are distributed in **include/xercesc** directory.
- Migration considerations:
  · Windows application,
  either change the **include directories** setting to "..\..\..\..\..\src**xercesc**"
  C/C++- >Preprocessor),
  or
  change the relevant #include instances in the source/header files, accordingly, eg
  #include <util/XMLString.hpp> be changed to
  #include <**xercesc**/util/XMLString.hpp>
  · Unix application,
  either change the **include search path** in the Makefile to " -I <installroot>/include/**xercesc**",
  or
  change the relevant #include instances in the source/header files as shown above.

Public API Changes in Xerces-C++ 1.7.0

The following lists the public API changes between the Xerces-C++ 1.7.0 and the Xerces-C++ 1.7.0 releases of the parser.

New Public API

- Added SAX2-ext's DeclHandler class. See the API documentation page for details.
- To support SAX2-ext's DeclHandler, the following new methods are added in classes DefaultHandler and SAX2XMLReader:
  · void DefaultHandler::elementDecl(const XMLCh* const name, const XMLCh* const model)
  · void DefaultHandler::attributeDecl(const XMLCh* const eName, const XMLCh* const aName, const XMLCh* const type, const XMLCh* const mode, const XMLCh* const value)
  · void DefaultHandler::internalEntityDecl(const XMLCh* const name, const XMLCh* const value)
  · void DefaultHandler::externalEntityDecl(const XMLCh* const name, const XMLCh* const

publicId, const XMLCh* const systemId)
  - DeclHandler* SAX2XMLReader::getDeclarationHandler() const
  - void SAX2XMLReader::setDeclarationHandler(DeclHandler* const handler)
- To conform to DOM Level 2 specification, the following methods are added:
  - DOM_Node DOM_NodeIterator::getRoot()
  - DOM_Node DOM_TreeWalker::getRoot()
  - bool DOM_Node::hasAttributes() const
  - bool DOM_Element::hasAttribute(const DOMString &name) const
  - bool DOM_Element::hasAttributeNS(const DOMString &namespaceURI, const DOMString localName) const
  - IDOM_Node* IDOM_NodeIterator::getRoot()
  - IDOM_Node* IDOM_TreeWalker::getRoot()
  - bool IDOM_Node::hasAttributes() const
  - bool IDOM_Element::hasAttribute(const XMLCh* name) const
  - bool IDOM_Element::hasAttributeNS(const XMLCh* namespaceURI, const XMLCh* localName) const
- To fix [Bug 5570], a copy constructor is added to DOM_Range

Modified Public API
- To conform to the SAX2 specification, the namespace-prefixes feature in SAX2 is set to off as default.
- To fix [Bug 6330], the Base64::encode and Base64::decode have been modified as follows
  - static XMLByte* Base64::encode(const XMLByte* const inputData, const unsigned int inputLength, unsigned int* outputLength);
  - static XMLByte* Base64::decode(const XMLByte* const inputData, unsigned int* outputLength);
  - static XMLCh* decode(const XMLCh* const inputData, unsigned int* outputLength);
- To conform to DOM Level 2 specification, the DOM_Node::supports and IDOM_Node::supports are modified to
  - bool DOM_Node::isSupported(const DOMString &feature, const DOMString &version) const
  - bool IDOM_Node::isSupported(const XMLCh* feature, const XMLCh* version) const

Deprecated Public API
- No Deprecated Public API in this release.

Migrating from Xerces-C++ 1.5.2 to 1.6.0

The following section is a discussion of the technical differences between Xerces-C++ 1.5.2 code base and the Xerces-C++ 1.6.0 code base.

New features in Xerces-C++ 1.6.0

- Full Schema support is available in this release. See the Schema page for details.
- New sample SEnumVal to show how to enumerate the markup decls in a Schema Grammar is added.

Public API Changes in Xerces-C++ 1.6.0

The following lists the public API changes between the Xerces-C++ 1.5.2 and the Xerces-C++ 1.6.0 releases of the parser.

New Public API
- It should not be a fatal error if a schema InputSource is not found. Add the following new methods:
  - const bool InputSource::getIssueFatalErrorIfNotFound() const
  - void InputSource::setIssueFatalErrorIfNotFound(const bool flag

- Allow code to take advantage of the fact that the length of the prefix and local name are known when constructing the QName. Add the following new methods:
  - void QName::setNPrefix(const XMLCh*, const unsigned int)
  - void QName::setNLocalPart(const XMLCh*, const unsigned int)
- To support schemaLocation and noNamespaceSchemaLocation to be specified outside the instance document, the following new methods are added:
  - XMLCh* DOMParser::getExternalSchemaLocation() const
  - XMLCh* DOMParser::getExternalNoNamespaceSchemaLocation() const
  - void DOMParser::setExternalSchemaLocation(const XMLCh* const schemaLocation)
  - void DOMParser::setExternalNoNamespaceSchemaLocation(const char* const noNamespaceSchemaLocation)
  - XMLCh* IDOMParser::getExternalSchemaLocation() const
  - XMLCh* IDOMParser::getExternalNoNamespaceSchemaLocation() const
  - void IDOMParser::setExternalSchemaLocation(const XMLCh* const schemaLocation)
  - void IDOMParser::setExternalNoNamespaceSchemaLocation(const char* const noNamespaceSchemaLocation)
  - XMLCh* SAXParser::getExternalSchemaLocation() const
  - XMLCh* SAXParser::getExternalNoNamespaceSchemaLocation() const
  - void SAXParser::setExternalSchemaLocation(const XMLCh* const schemaLocation)
  - void SAXParser::setExternalNoNamespaceSchemaLocation(const char* const noNamespaceSchemaLocation)
  - and the following properties are recognized by SAX2XMLReader:
    - http://apache.org/xml/properties/schema/external-schemaLocation
    - http://apache.org/xml/properties/schema/external-noNamespaceSchemaLocation
- To support identity constraints, the following new method is added:
  - QName* XMLAttr::getAttName() const

Modified Public API
- To support attribute constraint checking, the constant values in XMLAttDef::DefAttTypes have been re-ordered.

Deprecated Public API
- Root Element check is moved from XMLValidator to XMLScanner. Thus XMLValidator::checkRootElement() is deprecated.

Migrating from Xerces-C++ 1.4.0 to 1.5.2

The following section is a discussion of the technical differences between Xerces-C++ 1.4.0 code base and the Xerces-C++ 1.5.2 code base.

New features in Xerces-C++ 1.5.2

Schema subset support and an experimental IDOM are available in this release.

Schema Subset Support
- New function "setDoSchema" is added to DOM/SAX parser.
- New feature "http://apache.org/xml/features/validation/schema" is recognized by SAX2XMLReader.
- New classes such as SchemaValidator, TraverseSchema ... are added.
- The Scanner is enhanced to process schema.
- New sample data files personal-schema.xml and personal.xsd.
- New command line option "-s" for samples.

See the Schema page for details.

Experimental IDOM

The experimental IDOM API is a new design of the C++ DOM API. If you would like to migrate from DOM to the experimental IDOM, please refer to IDOM programming guide. Please note that this experimental IDOM API is only a prototype and is subject to change.

Changes required to migrate to Xerces-C++ 1.5.2

There are some architectural changes between the Xerces-C++ 1.4.0 and the Xerces-C++ 1.5.2 releases of the parser, and as a result, some code has undergone restructuring as shown below.

Validator directory Reorganization

- common content model files such as DFAContentModel ... are moved to a new directory called src/validators/common
- DTD related files are moved to a new directory called src/validators/DTD
- new directory src/validators/Datatype is created to store all datatype validators
- new directory src/validators/schema is created to store Schema related files

DTDValidator

DTDValidator was design to scan, validate and store the DTD in Xerces-C++ 1.4.0 or earlier. In Xerces-C++ 1.5.2, this process is broken down into three components:

- new class DTDScanner - to scan the DTD
- new class DTDGrammar - to store the DTD Grammar
- DTDValidator - to validate the DTD only

Migrating from XML4C 2.x to Xerces-C++ 1.4.0

The following section is a discussion of the technical differences between XML4C 2.x code base and the new Xerces-C++ 1.4.0 code base.

Summary of changes required to migrate from XML4C 2.x to Xerces-C++ 1.4.0

There are some major architectural changes between the 2.3.x and Xerces-C++ 1.4.0 releases of the parser, and as a result the code has undergone significant restructuring. The list below mentions the public api's which existed in 2.3.x and no longer exist in Xerces-C++ 1.4.0. It also mentions the Xerces-C++ 1.4.0 api which will give you the same functionality. Note: This list is not exhaustive. The API docs (and ultimately the header files) supplement this information.

- `parsers/[Non]Validating[DOM/SAX]parser.hpp`

  These files/classes have all been consolidated in the new version to just two files/classes: `[DOM/SAX]Parser.hpp`. Validation is now a property which may be set before invoking the `parse`. Now, the `setDoValidation()` method controls the validation processing.
- The `framework/XMLDocumentTypeHandler.hpp` been replaced with `validators/DTD/DocTypeHandler.hpp`.
- The following methods now have different set of parameters because the underlying base class methods have changed in the 3.x release. These methods belong to one of `XMLDocumentHandler`, `XMLErrorReporter` or `DocTypeHandler` interfaces.
  - `[Non]Validating[DOM/SAX]Parser::docComment`
  - `[Non]Validating[DOM/SAX]Parser::doctypePI`
  - `[Non]ValidatingSAXParser::elementDecl`
  - `[Non]ValidatingSAXParser::endAttList`
  - `[Non]ValidatingSAXParser::entityDecl`

  - `[Non]ValidatingSAXParser::notationDecl`
  - `[Non]ValidatingSAXParser::startAttList`
  - `[Non]ValidatingSAXParser::TextDecl`
  - `[Non]ValidatingSAXParser::docComment`
  - `[Non]ValidatingSAXParser::docPI`
  - `[Non]Validating[DOM/SAX]Parser::endElement`
  - `[Non]Validating[DOM/SAX]Parser::startElement`
  - `[Non]Validating[DOM/SAX]Parser::XMLDecl`
  - `[Non]Validating[DOM/SAX]Parser::error`
- The following methods/data members changed visibility from `protected` in 2.3.x to `private` (with public setters and getters, as appropriate).
  - `[Non]ValidatingDOMParser::fDocument`
  - `[Non]ValidatingDOMParser::fCurrentParent`
  - `[Non]ValidatingDOMParser::fCurrentNode`
  - `[Non]ValidatingDOMParser::fNodeStack`
- The following files have moved, possibly requiring changes in the `#include` statements.
  - `MemBufInputSource.hpp`
  - `StdInInputSource.hpp`
  - `URLInputSource.hpp`
- All the DTD validator code was moved from `internal` to separate `validators/DTD` directory.
- The error code definitions which were earlier in `internal/ErrorCodes.hpp` are now split up into the following files:
  - `framework/XMLErrorCodes.hpp` - Core XML errors
  - `framework/XMLValidityCodes.hpp` - DTD validity errors
  - `util/XMLExceptMsgs.hpp` - C++ specific exception codes.

The Samples

The sample programs no longer use any of the unsupported util/xxx classes. They only existed to allow us to write portable samples. But, since we feel that the wide character APIs are supported on a lot of platforms these days, it was decided to go ahead and just write the samples in terms of these. If your system does not support these APIs, you will not be able to build and run the samples. On some platforms, these APIs might perhaps be optional packages or require runtime updates or some such action.

More samples have been added as well. These highlight some of the new functionality introduced in the new code base. And the existing ones have been cleaned up as well.

The new samples are:

1. PParse - Demonstrates 'progressive parse' (see below)
2. StdInParse - Demonstrates use of the standard in input source
3. EnumVal - Shows how to enumerate the markup decls in a DTD Validator

Parser Classes

In the XML4C 2.x code base, there were the following parser classes (in the src/parsers/ source directory): NonValidatingSAXParser, ValidatingSAXParser, NonValidatingDOMParser, ValidatingDOMParser. The non-validating ones were the base classes and the validating ones just derived from them and turned on the validation. This was deemed a little bit overblown, considering the tiny amount of code required to turn on validation and the fact that it makes people use a pointer to the parser in most cases (if they needed to support either validating or non-validating versions.)

The new code base just has SAXParer and DOMParser classes. These are capable of handling both validating and non-validating modes, according to the state of a flag that you can set on them. For instance, here is a code snippet that shows this in action.

```
    void ParseThis(const  XMLCh* const fileToParse,
                   const bool validate)
    {
      //
      // Create a SAXParser. It can now just be
      // created by value on the stack if we want
      // to parse something within this scope.
      //
      SAXParser myParser;

      // Tell it whether to validate or not
      myParser.setDoValidation(validate);

      // Parse and catch exceptions...
      try
      {
        myParser.parse(fileToParse);
      }
        ...
    };
```

We feel that this is a simpler architecture, and that it makes things easier for you. In the above example, for instance, the parser will be cleaned up for you automatically upon exit since you don't have to allocate it anymore.

Moved Classes to src/framework

Some of the classes previously in the src/internal/ directory have been moved to their more correct location in the src/framework/ directory. These are classes used by the outside world and should have been framework classes to begin with. Also, to avoid name classes in the absence of C++ namespace support, some of these clashes have been renamed to make them more XML specific and less likely to clash. More classes might end up being moved to framework as well.

So you might have to change a few include statements to find these classes in their new locations. And you might have to rename some of the names of the classes, if you used any of the ones whose names were changed.

Util directory Reorganization

The src/util directory was becoming somewhat of a dumping ground of platform and compiler stuff. So we reworked that directory to better spread things out. The new scheme is:

util - The platform independent utility stuff

- MsgLoaders - Holds the msg loader implementations
  1. ICU
  2. InMemory
  3. MsgCatalog
  4. Win32
- Compilers - All the compiler specific files
- Transcoders - Holds the transcoder implementations
  1. Iconv

2. ICU
3. Win32

- Platforms
  1. AIX
  2. HP-UX
  3. Linux
  4. Solaris
  5. ....
  6. Win32

This organization makes things much easier to understand. And it makes it easier to find which files you need and which are optional. Note that only per-platform files have any hard coded references to specific message loaders or transcoders. So if you don't include the ICU implementations of these services, you don't need to link in ICU or use any ICU headers. The rest of the system works only in terms of the abstraction APIs.

38 Feedback Procedures

Questions or Comments

Please browse through this bundled documentation completely. Most of the common questions have been answered in the FAQ's. Specifically, do read the answer to " Is there any kind of support available for Xerces-C++?". Browsing this documentation, may be the quickest way to get an answer. Of course, if all else fails, as mentioned in the link above, you can post a question to the Xerces-C++ mailing list [21] .

See Bug Reporting if you would like to report a defect (greatly appreciated!).

Acknowledgements

Ever since this source code base was initially created, many people have helped to port the code to different platforms, and provided patches for both new features and bug fixes.

Listed below are some names (in alphabetical order) of people to whom we would like to give special thanks.

- Nadav Aharoni
- Curt Arnold
- Edward Avis
- Anupam Bagchi
- Torbjörn Bäckström
- Abe Backus
- Frank Balluffi
- Matthew Baker
- Devin Barnhart
- James Berry
- David Bertoni
- John Bellardo
- Arundhati Bhowmick
- Edward Bortner
- Sean Bright
- Phil Brown
- Robert Buck
- Sumit Chawla
- Nick Chiang
- Chih Hsiang Chou
- Radovan Chytracek
- Hiram Clawson

- John Clayton
- Todd Collins
- Michael Crawford
- Murray Cumming
- Helmut Eiken
- Mark Everline
- Simon Fell
- Paul Ferguson
- Pierpaolo Fumagalli
- Gary Gale
- Max Gotlib
- Petr Gotthard
- Susan Hardenbrook
- Jeff Harrell
- Andy Heninger
- William L. Hopper
- Michael Huedepohl
- Rahul Jain
- Tom Jordahl
- Christopher Just
- Martin Kalen
- Joe Kesselman
- Artur Klauser
- Bob Kline
- Richard Ko
- Paul Kramer
- Volker Krause
- Arnaud LeHors
- Andy Levine
- Jeff Lewis
- Matt Lovett
- Sean MacRoibeaird
- Alberto Massari
- Don Mastrovito
- David McCreedy
- Jordan Naftolin
- Tinny Ng
- David Nickerson
- Khaled Noaman
- Michael Ottati
- Kevin Philips
- Mike Pogue
- Joe Polastre
- John Ponzo
- Shengkai Qu
- Gareth Reakes
- Jim Reitz
- Dean Roddey

- John Roper
- Steven Rosenthal
- Erik Rydgren
- Bill Schindler
- Erik Schroeder
- Christian Schuhegger
- John Smirl
- Andrei Smirnov
- Gereon Steffens
- Jason Stewart
- Rick J. Stevens
- Roman Sulzhyk
- Linda M. Swan
- Pieter Van-Dyck
- Curtis Walker
- John Warrier
- Tom Watson
- Mark Weaver
- Roger Webster
- Robert Weir
- Carolyn Weiss
- Kari Whitcomb
- Dietrich Wolf
- Kirk Wylie
- Peter A. Volchek
- Grace Yan
- PeiYong Zhang
- Henry Zongaro

39 Bug Reporting

How to report bugs

Please report bugs to Bugzilla [28] , the Apache bug database. Pick the product "Xerces-C++: Apache XML parsers in C++" using the following components:

Component	Description
DOM	Items specific to DOM
SAX/SAX2	Items specific to SAX or SAX2
Non-Validating Parser	General Parsing Problem
Validating Parser (DTD)	DTD related parser issue
Validating Parser (Schema)	Schema related parser issue
Utilities	Items related to utilities like MessageLoader, Transcoder, NetAccessors, Platform specific utilities
Build	Problem with build, makefile, project files
Documentation	Documentation bugs such as FAQ, Programming Guide
Samples/Tests	Samples or test cases related issues
Miscellaneous	Items not covered in other categories

A copy of your bug report is sent automatically to the discussion list Xerces-C++ mailing list [21] .

Search first

Check the Bugzilla [28] database before submitting your bug report to avoid creating a duplicate report. Even the bug has been reported already, you may add a comment to the existing report since your contribution may lead to a quicker identification/resolution to the bug reported.

Write good bug report

Writing a useful bug report, which makes the bug reproducible, is the first step towards the resolution of the bug. Specifics about the bug, like

- Xerces-C++ version number
- Platform
- Operating system and version number
- Compiler and version number
- The XML document (or excerpt) that failed
- The C++ application code that failed
- Whether you built the Xerces-C++ library yourself or used the binary distribution
- What happened

are all necessary information to allow developer to reproduce, identify, evaluate and eventually, fix the bug, which is the very purpose of your reporting of the bug.

And please use the Keyword `"PatchAvailable"` if a patch is attached

40 PDF Documentation

PDF Documentation

You can get the entire Xerces-C++ documentation in PDF format, xerces-c.pdf [56] (or in zipped format xerces-c.pdf.tar.gz [57]), for printing and offline reference.

> ***Note:*** *A word of caution! The tools to create the PDF documentation are still experimental. So the resulting PDF document is not perfect. We would be glad to receive your comments on the Xerces-C++ mailing list [21] .*

Appendix A
Links Reference

[1] http://www.w3.org/XML/

[2] http://www.w3.org/TR/REC-xml

[3] http://www.w3.org/TR/2000/REC-xml-20001006

[4] http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/

[5] http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/

[6] http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/

[7] http://sax.sourceforge.net/

[8] http://www.w3.org/TR/1999/REC-xml-names-19990114/

[9] http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[10] http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[11] http://www.w3.org/TR/xml11/

[12] http://www.w3.org/TR/xml-names11/

[13] http://www.w3.org/TR/DOM-Level-3-Core/

[14] http://www.w3.org/TR/DOM-Level-3-LS/

[15] http://xml.apache.org/LICENSE

[16] http://www.cygwin.com

[17] http://www.gnu.org

[18] http://www.gnu.org/software/gcc/gcc.html

[19] http://www.gnu.org/software/autoconf/autoconf.html

[20] http://www.gnu.org/software/make/make.html

[21] mailto:xerces-c-dev@xml.apache.org

[22] http://oss.software.ibm.com/icu/

[23] http://oss.software.ibm.com/developerworks/opensource/

[24] http://oss.software.ibm.com/icu/download/index.html

[25] http://oss.software.ibm.com/cvs/icu/~checkout~/icu/readme.html

[26] http://www.rpm.org/RPM-HOWTO

[27] http://marc.theaimsgroup.com/?l=xerces-c-dev

[28] http://nagoya.apache.org/bugzilla/

[29] http://www.x.org/terms.htm

[30] http://www.alphaworks.ibm.com/tech/xml4c

[31] http://xml.apache.org/xerces-c/index.html

[32] http://xml.apache.org/dist/xerces-c/

[33] http://xml.apache.org/

[34] http://xml.apache.org/cocoon/index.html

[35] http://www.stack.nl/~dimitri/doxygen/

[36] http://www.research.att.com/sw/tools/graphviz/

[37] http://www.gnu.org/software/tar/tar.html

[38] http://www.gnu.org/manual/tar/html_node/tar_117.html#SEC112

[39] http://sunsolve.sun.com

[40] http://www.research.att.com/sw/tools/graphviz/license/index.html

[41] http://www.w3.org/TR/xmlschema-0/

[42] http://www.w3.org/TR/xmlschema-1/

[43] http://www.w3.org/TR/xmlschema-2/

[44] http://www.w3.org/TR/2003/WD-DOM-Level-3-Core-20030226/DOM3-Core.html#core-ID-3A0ED0A4

[45] http://www.w3.org/TR/REC-xml#charsets

[46] http://www.iana.org/assignments/character-sets

[47] http://rhn.redhat.com/errata/RHBA-2002-055.html

[48] http://xml.apache.org/xerces2-j/index.html

[49] http://www.oasis-open.org/cover/xml.html

[50] http://www.w3.org/TR/xhtml1/xhtml1.zip

[51] http://xml.apache.org/xalan-c/overview.html

[52] http://software.decisionsoft.com

[53] http://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20020409/

[54] http://www.w3.org/TR/2002/WD-DOM-Level-3-ASLS-20020409/

[55] http://www.w3.org/TR/REC-xml#sec-guessing

[56] http://xml.apache.org/xerces-c/pdf/xerces-c.pdf

[57] http://xml.apache.org/xerces-c/pdf/xerces-c.pdf.tar.gz