INTERNATIONAL STANDARD

ISO/IEC 23360-1-4

First edition
2021-10

Linux Standard Base (LSB)—

Part 1-4: Languages specification

Reference number
ISO/IEC 23360-1-4:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by the Linux Foundation as Linux Standard Base (LSB): Languages specification and drafted in accordance with its editorial rules. It was assigned to Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*, and adopted by National Bodies.

This first edition of ISO/IEC 23360-1-4 cancels and replaces ISO/IEC 23360-1:2006, which has been technically revised.

This document is based on "The GNU Free Documentation License, version 1.1". The license is available at https://www.gnu.org/licenses/old-licenses/fdl-1.1.html.

A list of all parts in the ISO/IEC 23660 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Contents

Introduction

The LSB defines a binary interface for application programs that are compiled and packaged for LSB-conforming implementations on many different hardware architectures. A binary specification must include information specific to the computer processor architecture for which it is intended. To avoid the complexity of conditional descriptions, the specification has instead been divided into generic parts which are augmented by one of several architecture-specific parts, depending on the target processor architecture; the generic part will indicate when reference must be made to the architecture part, and vice versa.

This document should be used in conjunction with the documents it references. This document enumerates the system components it includes, but descriptions of those components may be included entirely or partly in this document, partly in other documents, or entirely in other reference documents. For example, the section that describes system service routines includes a list of the system routines supported in this interface, formal declarations of the data structures they use that are visible to applications, and a pointer to the underlying referenced specification for information about the syntax and semantics of each call. Only those routines not described in standards referenced by this document, or extensions to those standards, are described in the detail. Information referenced in this way is as much a part of this document as is the information explicitly included here.

The specification carries a version number of either the form `x.y` or `x.y.z`. This version number carries the following meaning:

1. The first number (`x`) is the major version number. Versions sharing the same major version number shall be compatible in a backwards direction; that is, a newer version shall be compatible with an older version. Any deletion of a library results in a new major version number. Interfaces marked as deprecated may be removed from the specification at a major version change.
2. The second number (`y`) is the minor version number. Libraries and individual interfaces may be added, but not removed. Interfaces may be marked as deprecated at a minor version change. Other minor changes may be permitted at the discretion of the LSB workgroup.
3. The third number (`z`), if present, is the editorial level. Only editorial changes should be included in such versions.

Since this specification is a descriptive Application Binary Interface, and not a source level API specification, it is not possible to make a guarantee of 100% backward compatibility between major releases. However, it is the intent that those parts of the binary interface that are visible in the source level API will remain backward compatible from version to version, except where a feature marked as "Deprecated" in one release may be removed from a future release. Implementors are strongly encouraged to make use of symbol versioning to permit simultaneous support of applications conforming to different releases of this specification.

LSB is a trademark of the Linux Foundation. Developers of applications or implementations interested in using the trademark should see the Linux Foundation Certification Policy for details.

I Introductory Elements

1 Scope

The LSB Languages specification defines components for runtime languages which are found on an LSB conforming system.

2 Normative References

The specifications listed below are referenced in whole or in part by the LSB Languages specification. Such references may be normative or informative; a reference to specification shall only be considered normative if it is explicitly cited as such. The LSB Languages specification may make normative references to a portion of these specifications (that is, to define a specific function or group of functions); in such cases, only the explicitly referenced portion of the specification is to be considered normative.

Table 2-1 Informative References

Name	Title	URL
ISO C (1999)	ISO/IEC 9899:1999 - Programming Languages -- C	
Perl Core Modules	Perl 5.8.8 Core Modules	http://perldoc.perl.org/5.8.8/index-modules-A.html
Perl Functions	Perl 5.8.8 Functions	http://perldoc.perl.org/5.8.8/perlfunc.html
Perl Language Reference	Perl 5.8.8 Language Reference	http://perldoc.perl.org/5.8.8/index-language.html
Perl Manual	Perl 5.8.8 Manual Page	http://perldoc.perl.org/5.8.8/perlrun.html
Perl Operators	Perl 5.8.8 Operators and Precedence	http://perldoc.perl.org/5.8.8/perlop.html
Perl Syntax	Perl 5.8.8 Syntax	http://perldoc.perl.org/5.8.8/perlsyn.html
Python Library Reference	Python Library Reference Release 2.4.2	http://www.python.org/doc/2.4.2/lib/lib.html
Python Reference Manual	Python Reference Manual Release 2.4.2	http://www.python.org/doc/2.4.2/ref/ref.html
Reference Manual for libxml2	Reference Manual for libxml2	http://xmlsoft.org/html/index.html
Reference Manual for libxslt	Reference Manual for libxslt	http://xmlsoft.org/xslt/html/index.html

3 Requirements

This specification describes runtime language interpreters which shall be found in specified locations. It also defines a number of runtime modules which shall be in an implementation-defined directory which the interpreters shall search by default.

3.1 Relevant Libraries

The libraries listed in Table 3-1 shall be available on a Linux Standard Base system, with the specified runtime names. This list may be supplemented or amended by the architecture-specific specification.

Table 3-1 Standard Library Names

Library	Runtime Name
libxml2	libxml2.so.2
libxslt	libxslt.so.1

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

4 Terms and Definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 2382, ISO 80000–2, and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

4.1

archLSB

Some LSB specification documents have both a generic, architecture-neutral part and an architecture-specific part. The latter describes elements whose definitions may be unique to a particular processor architecture. The term archLSB may be used in the generic part to refer to the corresponding section of the architecture-specific part.

4.2

Binary Standard, ABI

The total set of interfaces that are available to be used in the compiled binary code of a conforming application, including the run-time details such as calling conventions, binary format, C++ name mangling, etc.

4.3

Implementation-defined

Describes a value or behavior that is not defined by this document but is selected by an implementor. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementor shall document such a value or behavior so that it can be used correctly by an application.

4.4

Shell Script

A file that is read by an interpreter (e.g., awk). The first line of the shell script includes a reference to its interpreter binary.

4.5

Source Standard, API

The total set of interfaces that are available to be used in the source code of a conforming application. Due to translations, the Binary Standard and the Source Standard may contain some different interfaces.

4.6

Undefined

Describes the nature of a value or behavior not defined by this document which results from use of an invalid program construct or invalid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

4.7

Unspecified

Describes the nature of a value or behavior not specified by this document which results from use of a valid program construct or valid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

In addition, for the portions of this specification which build on IEEE Std 1003.1-2001, the definitions given in *IEEE Std 1003.1-2001, Base Definitions, Chapter 3* apply.

5 Documentation Conventions

Throughout this document, the following typographic conventions are used:

`function()`

the name of a function

command

the name of a command or utility

`CONSTANT`

a constant value

`parameter`

a parameter

`variable`

a variable

Throughout this specification, several tables of interfaces are presented. Each entry in these tables has the following format:

name

the name of the interface

(symver)

An optional symbol version identifier, if required.

[*`refno`*]

A reference number indexing the table of referenced specifications that follows this table.

For example,

forkpty(GLIBC_2.0) [SUSv4]

refers to the interface named `forkpty()` with symbol version `GLIBC_2.0` that is defined in the reference indicated by the tag `SUSv4`.

Note: For symbols with versions which differ between architectures, the symbol versions are defined in the architecture specific parts of of this module specification only. In the generic part, they will appear without symbol versions.

II Python Interpreter

6 Python Interpreter

6.1 Introduction

The Python intrepreter API is described in the Python Library Reference, with the following requirements for an LSB conforming runtime.

6.2 Python Interpreter Location

The Python interpreter binary, or a link to the binary, shall exist at `/usr/bin/python`.

6.3 Python Interpreter Version

The default installed Python version shall be 2.4.2 or greater.

6.4 Operators and Functions

Core Python operators, subroutines, and built-in functions shall be present and shall operate as defined in Python Reference Manual.

6.5 Python Modules

An LSB conforming implementation shall provide the Python modules as described in Table 6-1 with at least the behavior described as mandatory in the referenced underlying specification. Some Python modules may be marked as deprecated, and applications should avoid using these as they may be withdrawn in future releases of this specification.

Table 6-1 Python Modules

array [1]	csv [1]	imp [1]	posix [1]	sys [1]
binascii [1]	datetime [1]	itertools [1]	pwd [1]	syslog [1]
bisect [1]	errno [1]	locale [1]	random [1]	termios [1]
cPickle [1]	exceptions [1]	marshal [1]	re [1]	thread [1]
cStringIO [1]	fcntl [1]	mmap [1]	resource [1]	time [1]
cmath [1]	gc [1]	operator [1]	select [1]	unicodedata [1]
codecs [1]	grp [1]	os [1]	signal [1]	weakref [1]
collections [1]	heapq [1]	ossaudiodev [1]	socket [1]	zipimport [1]
crypt [1]	hotshot [1]	parser [1]	string [1]	zlib [1]

Referenced Specification(s)

[1]. Python Reference Manual

6.6 Python Interpreter Command

This section contains a description of the **python** command.

PYTHON

Name

`python` — an interpreted, interactive, object-oriented programming language

Synopsis

`python` `[-d ] [-E ] [-h ] [-i ] [ -m` *`module-name`* `] [-O ] [ -Q` *`argument`* `] [-S ] [-t ] [-u ] [-v ] [-V ] [ -W` *`argument`* `] [-x ] [ -c` *`command`* `|` *`script`* `| - ] [`*`arguments`*`]`

DESCRIPTION

Python is an interpreted, interactive, object-oriented programming language that combines remarkable power with very clear syntax. For an introduction to programming in Python you are referred to the Python Tutorial. The Python Library Reference documents built-in and standard types, constants, functions and modules. Finally, the Python Reference Manual describes the syntax and semantics of the core language in (perhaps too) much detail. (These documents may be located via the INTERNET RESOURCES below; they may be installed on your system as well.)

Python's basic power can be extended with your own modules written in C or C++. On most systems such modules may be dynamically loaded. Python is also adaptable as an extension language for existing applications. See the internal documentation for hints.

Documentation for installed Python modules and packages can be viewed by running the *pydoc* program.

COMMAND LINE OPTIONS

`-c` *command*

Specify the command to execute (see next section). This terminates the option list (following options are passed as arguments to the command).

`-d`

Turn on parser debugging output (for wizards only, depending on compilation options).

`-E`

Ignore environment variables like PYTHONPATH and PYTHONHOME that modify the behavior of the interpreter.

`-h`

Prints the usage for the interpreter executable and exits.

`-i`

When a script is passed as first argument or the `-c` option is used, enter interactive mode after executing the script or the command. It does not read the $PYTHONSTARTUP file. This can be useful to inspect global variables or a stack trace when a script raises an exception.

`-m` *module-name*

Searches *sys.path* for the named module and runs the corresponding .py file as a script.

`-O`

Turn on basic optimizations. This changes the filename extension for compiled (bytecode) files from .pyc to .pyo. Given twice, causes docstrings to be discarded.

`-Q` *argument*

Division control; see PEP 238. The argument must be one of "old" (the default, int/int and long/long return an int or long), "new" (new division semantics, i.e. int/int and long/long returns a float), "warn" (old division semantics with a warning for int/int and long/long), or "warnall" (old division semantics with a warning for all use of the division operator). For a use of "warnall", see the Tools/scripts/fixdiv.py script.

`-S`

Disable the import of the module *site* and the site-dependent manipulations of *sys.path* that it entails.

`-t`

Issue a warning when a source file mixes tabs and spaces for indentation in a way that makes it depend on the worth of a tab expressed in spaces. Issue an error when the option is given twice.

`-u`

Force stdin, stdout and stderr to be totally unbuffered. On systems where it matters, also put stdin, stdout and stderr in binary mode. Note that there is internal buffering in xreadlines(), readlines() and file-object iterators ("for line in sys.stdin") which is not influenced by this option. To work around this, you will want to use "sys.stdin.readline()" inside a "while 1:" loop.

`-v`

Print a message each time a module is initialized, showing the place (filename or built-in module) from which it is loaded. When given twice, print a message for each file that is checked for when searching for a module. Also provides information on module cleanup at exit.

`-V`

Prints the Python version number of the executable and exits.

`-W` *argument*

Warning control. Python sometimes prints warning message to *sys.stderr*. A typical warning message has the following form: *file:line: category: message.* By default, each warning is printed once for each source line where it occurs. This option controls how often warnings are printed. Multiple `-W` options may be given; when a warning matches more than one option, the action for the last matching option is performed. Invalid `-W` options are ignored (a warning message is printed about invalid options when the first warning is issued). Warnings can also be controlled from within a Python program using the *warnings* module.

The simplest form of *argument* is one of the following *action* strings (or a unique abbreviation): *ignore* to ignore all warnings; *default* to explicitly request the default behavior (printing each warning once per source line); *all* to print a warning each time it occurs (this may generate many messages if a warning is triggered repeatedly for the same source line, such as inside a loop); *module* to print each warning only only the first time it occurs in each module; *once* to print each warning only the first time it occurs in the program; or *error* to raise an exception instead of printing a warning message.

The full form of *argument* is *action:message:category:module:line*. Here, *action* is as explained above but only applies to messages that match the remaining fields. Empty fields match all values; trailing empty fields may be omitted. The *message* field matches the start of the warning message printed; this match is case-insensitive. The *category* field matches the warning category. This must be a class name; the match test whether the actual warning category of the message is a subclass of the specified warning category. The full class name must be given. The *module* field matches the (fully-qualified) module name; this match is case-sensitive. The *line* field matches the line number, where zero matches all line numbers and is thus equivalent to an omitted line number.

`-x`

Skip the first line of the source. This is intended for a DOS specific hack only. Warning: the line numbers in error messages will be off by one!

INTERPRETER INTERFACE

The interpreter interface resembles that of the UNIX shell: when called with standard input connected to a tty device, it prompts for commands and executes them until an EOF is read; when called with a file name argument or with a file as standard input, it reads and executes a *script* from that file; when called with `-c command`, it executes the Python statement(s) given as *command*. Here *command* may contain multiple statements separated by newlines. Leading whitespace is significant in Python statements! In non-interactive mode, the entire input is parsed before it is executed.

If available, the script name and additional arguments thereafter are passed to the script in the Python variable *sys.argv* , which is a list of strings (you must first *import sys* to be able to access it). If no script name is given, *sys.argv[0]* is an empty string; if `-c` is used, *sys.argv[0]* contains the string '*-c*'. Note that options interpreted by the Python interpreter itself are not placed in *sys.argv*.

In interactive mode, the primary prompt is >>>; the second prompt (which appears when a command is not complete) is .... The prompts can be changed by assignment to *sys.ps1* or *sys.ps2*. The interpreter quits when it reads an EOF at a prompt. When an unhandled exception occurs, a stack trace is printed and control returns to the primary prompt; in non-interactive mode, the interpreter exits after printing the stack trace. The interrupt signal raises the *KeyboardInterrupt* exception; other UNIX signals are not caught (except that SIGPIPE is sometimes ignored, in favor of the *IOError* exception). Error messages are written to stderr.

FILES AND DIRECTORIES

These are subject to difference depending on local installation conventions; ${prefix} and ${exec_prefix} are installation-dependent and should be interpreted as for GNU software; they may be the same. The default for both is `/usr/local`.

${exec_prefix}/bin/python

Recommended location of the interpreter.

${prefix}/lib/python<version> ${exec_prefix}/lib/python<version>

Recommended locations of the directories containing the standard modules.

${prefix}/include/python<version> ${exec_prefix}/include/python<version>

Recommended locations of the directories containing the include files needed for developing Python extensions and embedding the interpreter.

`~/.pythonrc.py`

User-specific initialization file loaded by the *user* module; not used by default or by most applications.

ENVIRONMENT VARIABLES

PYTHONHOME

Change the location of the standard Python libraries. By default, the libraries are searched in ${prefix}/lib/python<version> and ${exec_prefix}/lib/python<version>, where ${prefix} and ${exec_prefix} are installation-dependent directories, both defaulting to `/usr/local`. When $PYTHONHOME is set to a single directory, its value replaces both ${prefix} and ${exec_prefix}. To specify different values for these, set $PYTHONHOME to ${prefix}:${exec_prefix}.

PYTHONPATH

Augments the default search path for module files. The format is the same as the shell's $PATH: one or more directory pathnames separated by colons. Non-existent directories are silently ignored. The default search path is installation dependent, but generally begins with ${prefix}/lib/python<version> (see PYTHONHOME above). The default search path is always appended to $PYTHONPATH. If a script argument is given, the directory containing the script is inserted in the path in front of $PYTHONPATH. The search path can be manipulated from within a Python program as the variable *sys.path* .

PYTHONSTARTUP

If this is the name of a readable file, the Python commands in that file are executed before the first prompt is displayed in interactive mode. The file is executed in the same name space where interactive commands are executed so that objects defined or imported in it can be used without qualification in the interactive session. You can also change the prompts *sys.ps1* and *sys.ps2* in this file.

PYTHONY2K

Set this to a non-empty string to cause the *time* module to require dates specified as strings to include 4-digit years, otherwise 2-digit years are converted based on rules described in the *time* module documentation.

PYTHONOPTIMIZE

If this is set to a non-empty string it is equivalent to specifying the `-O` option. If set to an integer, it is equivalent to specifying `-O` multiple times.

PYTHONDEBUG

If this is set to a non-empty string it is equivalent to specifying the `-d` option. If set to an integer, it is equivalent to specifying `-d` multiple times.

PYTHONINSPECT

If this is set to a non-empty string it is equivalent to specifying the `-i` option.

PYTHONUNBUFFERED

If this is set to a non-empty string it is equivalent to specifying the `-u` option.

PYTHONVERBOSE

If this is set to a non-empty string it is equivalent to specifying the `-v` option. If set to an integer, it is equivalent to specifying `-v` multiple times.

AUTHOR

The Python Software Foundation: http://www.python.org/psf

INTERNET RESOURCES

Main website: http://www.python.org/ Documentation: http://docs.python.org/ Community website: http://starship.python.net/ Developer resources: http://www.python.org/dev/ FTP: ftp://ftp.python.org/pub/python/ Module repository: http://www.vex.net/parnassus/ Newsgroups: comp.lang.python, comp.lang.python.announce

LICENSING

Python is distributed under an Open Source license. See the file "LICENSE" in the Python source distribution for information on terms & conditions for accessing and otherwise using Python and for a DISCLAIMER OF ALL WARRANTIES.

III Perl Interpreter

7 Perl Interpreter

7.1 Introduction

The Perl intrepreter API is described in the Perl Language Reference, with the following requirements for an LSB conforming runtime.

7.2 Perl Interpreter Location

The Perl interpreter binary, or a link to the binary, shall exist at `/usr/bin/perl`.

7.3 Perl Interpreter Version

The default installed Perl version shall be 5.8.8 or greater.

7.4 Perl Operators and Functions

Core Perl operators, subroutines, and built-in functions shall be present and shall operate as defined in Perl Syntax, Perl Operators and Perl Functions.

7.5 Perl Modules

An LSB conforming implementation shall provide the Perl modules as described in Table 7-1 with at least the behavior described as mandatory in the referenced underlying specification. Some Perl modules may be marked as deprecated, and applications should avoid using these as they may be withdrawn in future releases of this specification.

Table 7-1 Perl Modules

AnyDBM_File [1]	Encode::JP [1]	I18N::Collate [1]	Net::Time [1]	Test::More [1]
Attribute::Handlers [1]	Encode::JP::H2Z [1]	I18N::LangTags [1]	Net::hostent [1]	Test::Simple [1]
AutoLoader [1]	Encode::JP::JIS7 [1]	I18N::LangTags::Detect [1]	Net::netent [1]	Text::Abbrev [1]
AutoSplit [1]	Encode::KR [1]	I18N::LangTags::List [1]	Net::protoent [1]	Text::Balanced [1]
B::Concise [1]	Encode::KR::2022_KR [1]	I18N::Langinfo [1]	Net::servent [1]	Text::ParseWords [1]
B::Debug [1]	Encode::MIME::Header [1]	IO [1]	O [1]	Text::Soundex [1]
B::Deparse [1]	Encode::Symbol [1]	IO::Dir [1]	Opcode [1]	Text::Tabs [1]
B::Lint [1]	Encode::TW [1]	IO::File [1]	POSIX [1]	Text::Wrap [1]
B::Showlex [1]	Encode::Unicode [1]	IO::Handle [1]	PerlIO [1]	Tie::Array [1]
B::Terse [1]	Encode::Unicode::UTF7 [1]	IO::Pipe [1]	PerlIO::encoding [1]	Tie::File [1]

B::Xref [1]	English [1]	IO::Poll [1]	PerlIO::scalar [1]	Tie::Handle [1]
Benchmark [1]	Env [1]	IO::Seekable [1]	PerlIO::via [1]	Tie::Hash [1]
CGI [1]	Exporter [1]	IO::Select [1]	PerlIO::via::QuotedPrint [1]	Tie::Memoize [1]
CGI::Apache [1]	Exporter::Heavy [1]	IO::Socket [1]	Pod::Checker [1]	Tie::RefHash [1]
CGI::Carp [1]	ExtUtils::Command [1]	IO::Socket::INET [1]	Pod::Find [1]	Tie::Scalar [1]
CGI::Cookie [1]	ExtUtils::Command::MM [1]	IO::Socket::UNIX [1]	Pod::Functions [1]	Tie::SubstrHash [1]
CGI::Pretty [1]	ExtUtils::Install [1]	IPC::Msg [1]	Pod::Html [1]	Time::HiRes [1]
CGI::Push [1]	ExtUtils::Installed [1]	IPC::Open2 [1]	Pod::InputObjects [1]	Time::Local [1]
CGI::Util [1]	ExtUtils::Liblist [1]	IPC::Open3 [1]	Pod::LaTeX [1]	Time::gmtime [1]
CPAN [1]	ExtUtils::Liblist::Kid [1]	IPC::Semaphore [1]	Pod::Man [1]	Time::localtime [1]
CPAN::FirstTime [1]	ExtUtils::MM_Unix [1]	IPC::SysV [1]	Pod::ParseLink [1]	Time::tm [1]
CPAN::Nox [1]	ExtUtils::MY [1]	List::Util [1]	Pod::ParseUtils [1]	Unicode::Collate [1]
Carp [1]	ExtUtils::MakeMaker [1]	Locale::Country [1]	Pod::Parser [1]	Unicode::Normalize [1]
Carp::Heavy [1]	ExtUtils::MakeMaker::Config [1]	Locale::Currency [1]	Pod::Perldoc::ToChecker [1]	Unicode::UCD [1]
Class::Struct [1]	ExtUtils::Manifest [1]	Locale::Language [1]	Pod::Perldoc::ToMan [1]	User::grent [1]
Cwd [1]	ExtUtils::Mkbootstrap [1]	Locale::Maketext [1]	Pod::Perldoc::ToNroff [1]	User::pwent [1]
DB [1]	ExtUtils::Mksymlists [1]	Locale::Script [1]	Pod::Perldoc::ToPod [1]	attributes [1]
DBM_Filter [1]	ExtUtils::Packlist [1]	MIME::Base64 [1]	Pod::Perldoc::ToText [1]	autouse [1]
DBM_Filter::encode [1]	ExtUtils::testlib [1]	MIME::QuotedPrint [1]	Pod::PlainText [1]	base [1]
DBM_Filter::int32 [1]	Fatal [1]	Math::BigFloat [1]	Pod::Select [1]	bigint [1]

DBM_Filter::null [1]	Fcntl [1]	Math::BigInt [1]	Pod::Text [1]	bignum [1]
DBM_Filter::utf8 [1]	File::Basename [1]	Math::BigInt::Calc [1]	Pod::Text::Color [1]	bigrat [1]
Data::Dumper [1]	File::CheckTree [1]	Math::BigInt::CalcEmu [1]	Pod::Text::Overstrike [1]	blib [1]
Devel::PPPort [1]	File::Compare [1]	Math::BigRat [1]	Pod::Text::Termcap [1]	bytes [1]
Devel::Peek [1]	File::Copy [1]	Math::Complex [1]	Pod::Usage [1]	charnames [1]
Devel::SelfStubber [1]	File::DosGlob [1]	Math::Trig [1]	SDBM_File [1]	constant [1]
Digest [1]	File::Find [1]	Memoize [1]	Safe [1]	diagnostics [1]
Digest::MD5 [1]	File::Glob [1]	Memoize::AnyDBM_File [1]	Scalar::Util [1]	fields [1]
Digest::base [1]	File::Path [1]	Memoize::Expire [1]	Search::Dict [1]	filetest [1]
Digest::file [1]	File::Spec [1]	Memoize::ExpireFile [1]	SelectSaver [1]	if [1]
DirHandle [1]	File::Spec::Functions [1]	Memoize::ExpireTest [1]	SelfLoader [1]	integer [1]
Dumpvalue [1]	File::Spec::Unix [1]	Memoize::SDBM_File [1]	Socket [1]	less [1]
Encode [1]	File::Temp [1]	Memoize::Storable [1]	Storable [1]	locale [1]
Encode::Alias [1]	File::stat [1]	NEXT [1]	Sys::Hostname [1]	open [1]
Encode::Byte [1]	FileCache [1]	Net::Cmd [1]	Sys::Syslog [1]	overload [1]
Encode::CJKConstants [1]	FileHandle [1]	Net::Config [1]	Term::ANSIColor [1]	re [1]
Encode::CN [1]	Filter::Simple [1]	Net::Domain [1]	Term::Complete [1]	sigtrap [1]
Encode::CN::HZ [1]	Filter::Util::Call [1]	Net::FTP [1]	Term::ReadLine [1]	sort [1]
Encode::Config [1]	FindBin [1]	Net::NNTP [1]	Test::Builder [1]	strict [1]
Encode::EBCDIC [1]	GDBM_File [1]	Net::Netrc [1]	Test::Builder::Module [1]	subs [1]

Encode::Encoder [1]	Getopt::Long [1]	Net::POP3 [1]	Test::Builder::Tester [1]	utf8 [1]
Encode::Encoding [1]	Getopt::Std [1]	Net::Ping [1]	Test::Builder::Tester::Color [1]	warnings [1]
Encode::Guess [1]	Hash::Util [1]	Net::SMTP [1]	Test::Harness [1]	warnings::register [1]

Referenced Specification(s)

[1]. Perl Language Reference

7.6 Perl Interpreter Command

The **perl** command is described in Perl Manual.

IV XML2 library

8 Libraries

8.1 Interfaces for libxml2

Table 8-1 defines the library name and shared object name for the libxml2 library

Table 8-1 libxml2 Definition

Library:	libxml2
SONAME:	libxml2.so.2

The behavior of the interfaces in this library is specified by the following specifications:

[libXML2] Reference Manual for libxml2

8.1.1 The XML C parser and toolkit for XML processing

8.1.1.1 Interfaces for The XML C parser and toolkit for XML processing

An LSB conforming implementation shall provide the generic functions for The XML C parser and toolkit for XML processing specified in Table 8-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 8-2 libxml2 - The XML C parser and toolkit for XML processing Function Interfaces

UTF8ToHtml(LIBXML2_2.4.30) [libXML2]	UTF8Toisolat1(LIBXML2_2.4.30) [libXML2]	__docbDefaultSAXHandler [libXML2]
__htmlDefaultSAXHandler [libXML2]	__oldXMLWDcompatibility [libXML2]	__xmlBufferAllocScheme [libXML2]
__xmlDefaultBufferSize [libXML2]	__xmlDefaultSAXHandler [libXML2]	__xmlDefaultSAXLocator [libXML2]
__xmlDeregisterNodeDefaultValue [libXML2]	__xmlDoValidityCheckingDefaultValue [libXML2]	__xmlGenericError [libXML2]
__xmlGenericErrorContext [libXML2]	__xmlGetWarningsDefaultValue [libXML2]	__xmlIndentTreeOutput [libXML2]
__xmlKeepBlanksDefaultValue [libXML2]	__xmlLastError [libXML2]	__xmlLineNumbersDefaultValue [libXML2]
__xmlLoadExtDtdDefaultValue [libXML2]	__xmlOutputBufferCreateFilenameValue [libXML2]	__xmlParserDebugEntities [libXML2]
__xmlParserInputBufferCreateFilenameValue [libXML2]	__xmlParserVersion [libXML2]	__xmlPedanticParserDefaultValue [libXML2]
__xmlRegisterNodeDefaultValue [libXML2]	__xmlSaveNoEmptyTags [libXML2]	__xmlStructuredError [libXML2]

__xmlSubstituteEntitiesDefaultValue [libXML2]	__xmlTreeIndentString [libXML2]	docbDefaultSAXHandlerInit(LIBXML2_2.4.30) [libXML2]
htmlAttrAllowed(LIBXML2_2.5.2) [libXML2]	htmlAutoCloseTag(LIBXML2_2.4.30) [libXML2]	htmlCreateFileParserCtxt(LIBXML2_2.4.30) [libXML2]
htmlCreateMemoryParserCtxt(LIBXML2_2.5.7) [libXML2]	htmlCreatePushParserCtxt(LIBXML2_2.4.30) [libXML2]	htmlCtxtReadDoc(LIBXML2_2.6.0) [libXML2]
htmlCtxtReadFd(LIBXML2_2.6.0) [libXML2]	htmlCtxtReadFile(LIBXML2_2.6.0) [libXML2]	htmlCtxtReadIO(LIBXML2_2.6.0) [libXML2]
htmlCtxtReadMemory(LIBXML2_2.6.0) [libXML2]	htmlCtxtReset(LIBXML2_2.6.0) [libXML2]	htmlCtxtUseOptions(LIBXML2_2.6.0) [libXML2]
htmlDefaultSAXHandlerInit(LIBXML2_2.4.30) [libXML2]	htmlDocContentDumpFormatOutput(LIBXML2_2.4.30) [libXML2]	htmlDocContentDumpOutput(LIBXML2_2.4.30) [libXML2]
htmlDocDump(LIBXML2_2.4.30) [libXML2]	htmlDocDumpMemory(LIBXML2_2.4.30) [libXML2]	htmlElementAllowedHere(LIBXML2_2.5.2) [libXML2]
htmlElementStatusHere(LIBXML2_2.5.2) [libXML2]	htmlEncodeEntities(LIBXML2_2.4.30) [libXML2]	htmlEntityLookup(LIBXML2_2.4.30) [libXML2]
htmlEntityValueLookup(LIBXML2_2.4.30) [libXML2]	htmlFreeParserCtxt(LIBXML2_2.4.30) [libXML2]	htmlGetMetaEncoding(LIBXML2_2.4.30) [libXML2]
htmlHandleOmittedElem(LIBXML2_2.4.30) [libXML2]	htmlInitAutoClose(LIBXML2_2.4.30) [libXML2]	htmlIsAutoClosed(LIBXML2_2.4.30) [libXML2]
htmlIsBooleanAttr(LIBXML2_2.4.30) [libXML2]	htmlIsScriptAttribute(LIBXML2_2.4.30) [libXML2]	htmlNewDoc(LIBXML2_2.4.30) [libXML2]
htmlNewDocNoDtD(LIBXML2_2.4.30) [libXML2]	htmlNodeDump(LIBXML2_2.4.30) [libXML2]	htmlNodeDumpFile(LIBXML2_2.4.30) [libXML2]
htmlNodeDumpFileFormat(LIBXML2_2.4.30) [libXML2]	htmlNodeDumpFormatOutput(LIBXML2_2.4.30) [libXML2]	htmlNodeDumpOutput(LIBXML2_2.4.30) [libXML2]
htmlParseCharRef(LIBXML2_2.4.30) [libXML2]	htmlParseChunk(LIBXML2_2.4.30) [libXML2]	htmlParseDoc(LIBXML2_2.4.30) [libXML2]
htmlParseDocument(LIBXML2_2.4.30) [libXML2]	htmlParseElement(LIBXML2_2.4.30) [libXML2]	htmlParseEntityRef(LIBXML2_2.4.30) [libXML2]
htmlParseFile(LIBXML2_2.4.30) [libXML2]	htmlReadDoc(LIBXML2_2.6.0) [libXML2]	htmlReadFd(LIBXML2_2.6.0) [libXML2]

htmlReadFile(LIBXML2_2.6.0) [libXML2]	htmlReadIO(LIBXML2_2.6.0) [libXML2]	htmlReadMemory(LIBXML2_2.6.0) [libXML2]
htmlSAXParseDoc(LIBXML2_2.4.30) [libXML2]	htmlSAXParseFile(LIBXML2_2.4.30) [libXML2]	htmlSaveFile(LIBXML2_2.4.30) [libXML2]
htmlSaveFileEnc(LIBXML2_2.4.30) [libXML2]	htmlSaveFileFormat(LIBXML2_2.4.30) [libXML2]	htmlSetMetaEncoding(LIBXML2_2.4.30) [libXML2]
htmlTagLookup(LIBXML2_2.4.30) [libXML2]	initGenericErrorDefaultFunc(LIBXML2_2.4.30) [libXML2]	inputPop(LIBXML2_2.4.30) [libXML2]
inputPush(LIBXML2_2.4.30) [libXML2]	isolat1ToUTF8(LIBXML2_2.4.30) [libXML2]	namePop(LIBXML2_2.4.30) [libXML2]
namePush(LIBXML2_2.4.30) [libXML2]	nodePop(LIBXML2_2.4.30) [libXML2]	nodePush(LIBXML2_2.4.30) [libXML2]
valuePop(LIBXML2_2.4.30) [libXML2]	valuePush(LIBXML2_2.4.30) [libXML2]	xmlACatalogAdd(LIBXML2_2.4.30) [libXML2]
xmlACatalogDump(LIBXML2_2.4.30) [libXML2]	xmlACatalogRemove(LIBXML2_2.4.30) [libXML2]	xmlACatalogResolve(LIBXML2_2.4.30) [libXML2]
xmlACatalogResolvePublic(LIBXML2_2.4.30) [libXML2]	xmlACatalogResolveSystem(LIBXML2_2.4.30) [libXML2]	xmlACatalogResolveURI(LIBXML2_2.4.30) [libXML2]
xmlAddAttributeDecl(LIBXML2_2.4.30) [libXML2]	xmlAddChild(LIBXML2_2.4.30) [libXML2]	xmlAddChildList(LIBXML2_2.4.30) [libXML2]
xmlAddDocEntity(LIBXML2_2.4.30) [libXML2]	xmlAddDtdEntity(LIBXML2_2.4.30) [libXML2]	xmlAddElementDecl(LIBXML2_2.4.30) [libXML2]
xmlAddEncodingAlias(LIBXML2_2.4.30) [libXML2]	xmlAddID(LIBXML2_2.4.30) [libXML2]	xmlAddNextSibling(LIBXML2_2.4.30) [libXML2]
xmlAddNotationDecl(LIBXML2_2.4.30) [libXML2]	xmlAddPrevSibling(LIBXML2_2.4.30) [libXML2]	xmlAddRef(LIBXML2_2.4.30) [libXML2]
xmlAddSibling(LIBXML2_2.4.30) [libXML2]	xmlAllocOutputBuffer(LIBXML2_2.4.30) [libXML2]	xmlAllocParserInputBuffer(LIBXML2_2.4.30) [libXML2]
xmlAttrSerializeTxtContent(LIBXML2_2.6.6) [libXML2]	xmlAutomataCompile(LIBXML2_2.4.30) [libXML2]	xmlAutomataGetInitState(LIBXML2_2.4.30) [libXML2]
xmlAutomataIsDeterminist(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewAllTrans(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewCountTrans(LIBXML2_2.4.30) [libXML2]

xmlAutomataNewCountTrans2(LIBXML2_2.6.14) [libXML2]	xmlAutomataNewCountedTrans(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewCounter(LIBXML2_2.4.30) [libXML2]
xmlAutomataNewCounterTrans(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewEpsilon(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewNegTrans(LIBXML2_2.6.21) [libXML2]
xmlAutomataNewOnceTrans(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewOnceTrans2(LIBXML2_2.6.14) [libXML2]	xmlAutomataNewState(LIBXML2_2.4.30) [libXML2]
xmlAutomataNewTransition(LIBXML2_2.4.30) [libXML2]	xmlAutomataNewTransition2(LIBXML2_2.5.7) [libXML2]	xmlAutomataSetFinalState(LIBXML2_2.4.30) [libXML2]
xmlBoolToText(LIBXML2_2.4.30) [libXML2]	xmlBufferAdd(LIBXML2_2.4.30) [libXML2]	xmlBufferAddHead(LIBXML2_2.4.30) [libXML2]
xmlBufferCCat(LIBXML2_2.4.30) [libXML2]	xmlBufferCat(LIBXML2_2.4.30) [libXML2]	xmlBufferContent(LIBXML2_2.4.30) [libXML2]
xmlBufferCreate(LIBXML2_2.4.30) [libXML2]	xmlBufferCreateSize(LIBXML2_2.4.30) [libXML2]	xmlBufferCreateStatic(LIBXML2_2.6.0) [libXML2]
xmlBufferDump(LIBXML2_2.4.30) [libXML2]	xmlBufferEmpty(LIBXML2_2.4.30) [libXML2]	xmlBufferFree(LIBXML2_2.4.30) [libXML2]
xmlBufferGrow(LIBXML2_2.4.30) [libXML2]	xmlBufferLength(LIBXML2_2.4.30) [libXML2]	xmlBufferResize(LIBXML2_2.4.30) [libXML2]
xmlBufferSetAllocationScheme(LIBXML2_2.4.30) [libXML2]	xmlBufferShrink(LIBXML2_2.4.30) [libXML2]	xmlBufferWriteCHAR(LIBXML2_2.4.30) [libXML2]
xmlBufferWriteChar(LIBXML2_2.4.30) [libXML2]	xmlBufferWriteQuotedString(LIBXML2_2.4.30) [libXML2]	xmlBuildQName(LIBXML2_2.5.7) [libXML2]
xmlBuildRelativeURI(LIBXML2_2.6.11) [libXML2]	xmlBuildURI(LIBXML2_2.4.30) [libXML2]	xmlByteConsumed(LIBXML2_2.6.6) [libXML2]
xmlC14NDocDumpMemory(LIBXML2_2.4.30) [libXML2]	xmlC14NDocSave(LIBXML2_2.4.30) [libXML2]	xmlC14NDocSaveTo(LIBXML2_2.4.30) [libXML2]
xmlC14NExecute(LIBXML2_2.4.30) [libXML2]	xmlCanonicPath(LIBXML2_2.5.4) [libXML2]	xmlCatalogAdd(LIBXML2_2.4.30) [libXML2]
xmlCatalogAddLocal(LIBXML2_2.4.30) [libXML2]	xmlCatalogCleanup(LIBXML2_2.4.30) [libXML2]	xmlCatalogConvert(LIBXML2_2.4.30) [libXML2]
xmlCatalogDump(LIBXML2_2.4.30) [libXML2]	xmlCatalogFreeLocal(LIBXML2_2.4.30) [libXML2]	xmlCatalogGetDefaults(LIBXML2_2.4.30) [libXML2]

xmlCatalogIsEmpty(LIBXML2_2.4.30) [libXML2]	xmlCatalogLocalResolve(LIBXML2_2.4.30) [libXML2]	xmlCatalogLocalResolveURI(LIBXML2_2.4.30) [libXML2]
xmlCatalogRemove(LIBXML2_2.4.30) [libXML2]	xmlCatalogResolve(LIBXML2_2.4.30) [libXML2]	xmlCatalogResolvePublic(LIBXML2_2.4.30) [libXML2]
xmlCatalogResolveSystem(LIBXML2_2.4.30) [libXML2]	xmlCatalogResolveURI(LIBXML2_2.4.30) [libXML2]	xmlCatalogSetDebug(LIBXML2_2.4.30) [libXML2]
xmlCatalogSetDefaultPrefer(LIBXML2_2.4.30) [libXML2]	xmlCatalogSetDefaults(LIBXML2_2.4.30) [libXML2]	xmlCharEncCloseFunc(LIBXML2_2.4.30) [libXML2]
xmlCharEncFirstLine(LIBXML2_2.4.30) [libXML2]	xmlCharEncInFunc(LIBXML2_2.4.30) [libXML2]	xmlCharEncOutFunc(LIBXML2_2.4.30) [libXML2]
xmlCharStrdup(LIBXML2_2.4.30) [libXML2]	xmlCharStrndup(LIBXML2_2.4.30) [libXML2]	xmlCheckFilename(LIBXML2_2.4.30) [libXML2]
xmlCheckHTTPInput(LIBXML2_2.6.0) [libXML2]	xmlCheckUTF8(LIBXML2_2.4.30) [libXML2]	xmlCheckVersion(LIBXML2_2.4.30) [libXML2]
xmlCleanupCharEncodingHandlers(LIBXML2_2.4.30) [libXML2]	xmlCleanupEncodingAliases(LIBXML2_2.4.30) [libXML2]	xmlCleanupGlobals(LIBXML2_2.5.8) [libXML2]
xmlCleanupInputCallbacks(LIBXML2_2.4.30) [libXML2]	xmlCleanupMemory(LIBXML2_2.6.5) [libXML2]	xmlCleanupOutputCallbacks(LIBXML2_2.4.30) [libXML2]
xmlCleanupParser(LIBXML2_2.4.30) [libXML2]	xmlCleanupThreads(LIBXML2_2.4.30) [libXML2]	xmlClearNodeInfoSeq(LIBXML2_2.4.30) [libXML2]
xmlClearParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlConvertSGMLCatalog(LIBXML2_2.4.30) [libXML2]	xmlCopyAttributeTable(LIBXML2_2.4.30) [libXML2]
xmlCopyChar(LIBXML2_2.4.30) [libXML2]	xmlCopyCharMultiByte(LIBXML2_2.4.30) [libXML2]	xmlCopyDoc(LIBXML2_2.4.30) [libXML2]
xmlCopyDocElementContent(LIBXML2_2.6.18) [libXML2]	xmlCopyDtd(LIBXML2_2.4.30) [libXML2]	xmlCopyElementTable(LIBXML2_2.4.30) [libXML2]
xmlCopyEntitiesTable(LIBXML2_2.4.30) [libXML2]	xmlCopyEnumeration(LIBXML2_2.4.30) [libXML2]	xmlCopyError(LIBXML2_2.6.0) [libXML2]
xmlCopyNamespace(LIBXML2_2.4.30) [libXML2]	xmlCopyNamespaceList(LIBXML2_2.4.30) [libXML2]	xmlCopyNode(LIBXML2_2.4.30) [libXML2]

xmlCopyNodeList(LIBXML2_2.4.30) [libXML2]	xmlCopyNotationTable(LIBXML2_2.4.30) [libXML2]	xmlCopyProp(LIBXML2_2.4.30) [libXML2]
xmlCopyPropList(LIBXML2_2.4.30) [libXML2]	xmlCreateDocParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlCreateEntityParserCtxt(LIBXML2_2.4.30) [libXML2]
xmlCreateEnumeration(LIBXML2_2.4.30) [libXML2]	xmlCreateFileParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlCreateIOParserCtxt(LIBXML2_2.4.30) [libXML2]
xmlCreateIntSubset(LIBXML2_2.4.30) [libXML2]	xmlCreateMemoryParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlCreatePushParserCtxt(LIBXML2_2.4.30) [libXML2]
xmlCreateURI(LIBXML2_2.4.30) [libXML2]	xmlCreateURLParserCtxt(LIBXML2_2.6.2) [libXML2]	xmlCtxtGetLastError(LIBXML2_2.6.0) [libXML2]
xmlCtxtReadDoc(LIBXML2_2.6.0) [libXML2]	xmlCtxtReadFd(LIBXML2_2.6.0) [libXML2]	xmlCtxtReadFile(LIBXML2_2.6.0) [libXML2]
xmlCtxtReadIO(LIBXML2_2.6.0) [libXML2]	xmlCtxtReadMemory(LIBXML2_2.6.0) [libXML2]	xmlCtxtReset(LIBXML2_2.6.0) [libXML2]
xmlCtxtResetLastError(LIBXML2_2.6.0) [libXML2]	xmlCtxtResetPush(LIBXML2_2.6.1) [libXML2]	xmlCtxtUseOptions(LIBXML2_2.6.0) [libXML2]
xmlCurrentChar(LIBXML2_2.4.30) [libXML2]	xmlDOMWrapFreeCtxt(LIBXML2_2.6.20) [libXML2]	xmlDOMWrapNewCtxt(LIBXML2_2.6.20) [libXML2]
xmlDebugCheckDocument(LIBXML2_2.6.15) [libXML2]	xmlDebugDumpAttr(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpAttrList(LIBXML2_2.4.30) [libXML2]
xmlDebugDumpDTD(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpDocument(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpDocumentHead(LIBXML2_2.4.30) [libXML2]
xmlDebugDumpEntities(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpNode(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpNodeList(LIBXML2_2.4.30) [libXML2]
xmlDebugDumpOneNode(LIBXML2_2.4.30) [libXML2]	xmlDebugDumpString(LIBXML2_2.4.30) [libXML2]	xmlDefaultSAXHandlerInit(LIBXML2_2.4.30) [libXML2]
xmlDelEncodingAlias(LIBXML2_2.4.30) [libXML2]	xmlDeregisterNodeDefault(LIBXML2_2.5.0) [libXML2]	xmlDetectCharEncoding(LIBXML2_2.4.30) [libXML2]
xmlDictCleanup(LIBXML2_2.6.18) [libXML2]	xmlDictCreate(LIBXML2_2.6.0) [libXML2]	xmlDictCreateSub(LIBXML2_2.6.5) [libXML2]
xmlDictExists(LIBXML2_2.6.17) [libXML2]	xmlDictFree(LIBXML2_2.6.0) [libXML2]	xmlDictLookup(LIBXML2_2.6.0) [libXML2]

xmlDictOwns(LIBXML2_2.6.0) [libXML2]	xmlDictQLookup(LIBXML2_2.6.0) [libXML2]	xmlDictReference(LIBXML2_2.6.0) [libXML2]
xmlDictSize(LIBXML2_2.6.0) [libXML2]	xmlDocCopyNode(LIBXML2_2.4.30) [libXML2]	xmlDocCopyNodeList(LIBXML2_2.6.15) [libXML2]
xmlDocDump(LIBXML2_2.4.30) [libXML2]	xmlDocDumpFormatMemory(LIBXML2_2.4.30) [libXML2]	xmlDocDumpFormatMemoryEnc(LIBXML2_2.4.30) [libXML2]
xmlDocDumpMemory(LIBXML2_2.4.30) [libXML2]	xmlDocDumpMemoryEnc(LIBXML2_2.4.30) [libXML2]	xmlDocFormatDump(LIBXML2_2.4.30) [libXML2]
xmlDocGetRootElement(LIBXML2_2.4.30) [libXML2]	xmlDocSetRootElement(LIBXML2_2.4.30) [libXML2]	xmlDumpAttributeDecl(LIBXML2_2.4.30) [libXML2]
xmlDumpAttributeTable(LIBXML2_2.4.30) [libXML2]	xmlDumpElementDecl(LIBXML2_2.4.30) [libXML2]	xmlDumpElementTable(LIBXML2_2.4.30) [libXML2]
xmlDumpEntitiesTable(LIBXML2_2.4.30) [libXML2]	xmlDumpEntityDecl(LIBXML2_2.4.30) [libXML2]	xmlDumpNotationDecl(LIBXML2_2.4.30) [libXML2]
xmlDumpNotationTable(LIBXML2_2.4.30) [libXML2]	xmlElemDump(LIBXML2_2.4.30) [libXML2]	xmlEncodeEntitiesReentrant(LIBXML2_2.4.30) [libXML2]
xmlEncodeSpecialChars(LIBXML2_2.4.30) [libXML2]	xmlExpCtxtNbCons(LIBXML2_2.6.21) [libXML2]	xmlExpCtxtNbNodes(LIBXML2_2.6.21) [libXML2]
xmlExpDump(LIBXML2_2.6.21) [libXML2]	xmlExpExpDerive(LIBXML2_2.6.21) [libXML2]	xmlExpFree(LIBXML2_2.6.21) [libXML2]
xmlExpFreeCtxt(LIBXML2_2.6.21) [libXML2]	xmlExpGetLanguage(LIBXML2_2.6.21) [libXML2]	xmlExpGetStart(LIBXML2_2.6.21) [libXML2]
xmlExpIsNillable(LIBXML2_2.6.21) [libXML2]	xmlExpMaxToken(LIBXML2_2.6.21) [libXML2]	xmlExpNewAtom(LIBXML2_2.6.21) [libXML2]
xmlExpNewCtxt(LIBXML2_2.6.21) [libXML2]	xmlExpNewOr(LIBXML2_2.6.21) [libXML2]	xmlExpNewRange(LIBXML2_2.6.21) [libXML2]
xmlExpNewSeq(LIBXML2_2.6.21) [libXML2]	xmlExpParse(LIBXML2_2.6.21) [libXML2]	xmlExpRef(LIBXML2_2.6.21) [libXML2]
xmlExpStringDerive(LIBXML2_2.6.21) [libXML2]	xmlExpSubsume(LIBXML2_2.6.21) [libXML2]	xmlFileClose(LIBXML2_2.4.30) [libXML2]
xmlFileMatch(LIBXML2_2.4.30) [libXML2]	xmlFileOpen(LIBXML2_2.4.30) [libXML2]	xmlFileRead(LIBXML2_2.4.30) [libXML2]

xmlFindCharEncodingHandler(LIBXML2_2.4.30) [libXML2]	xmlFreeAttributeTable(LIBXML2_2.4.30) [libXML2]	xmlFreeAutomata(LIBXML2_2.4.30) [libXML2]
xmlFreeCatalog(LIBXML2_2.4.30) [libXML2]	xmlFreeDoc(LIBXML2_2.4.30) [libXML2]	xmlFreeDocElementContent(LIBXML2_2.6.18) [libXML2]
xmlFreeDtd(LIBXML2_2.4.30) [libXML2]	xmlFreeElementTable(LIBXML2_2.4.30) [libXML2]	xmlFreeEntitiesTable(LIBXML2_2.4.30) [libXML2]
xmlFreeEnumeration(LIBXML2_2.4.30) [libXML2]	xmlFreeIDTable(LIBXML2_2.4.30) [libXML2]	xmlFreeInputStream(LIBXML2_2.4.30) [libXML2]
xmlFreeMutex(LIBXML2_2.4.30) [libXML2]	xmlFreeNode(LIBXML2_2.4.30) [libXML2]	xmlFreeNodeList(LIBXML2_2.4.30) [libXML2]
xmlFreeNotationTable(LIBXML2_2.4.30) [libXML2]	xmlFreeNs(LIBXML2_2.4.30) [libXML2]	xmlFreeNsList(LIBXML2_2.4.30) [libXML2]
xmlFreeParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlFreeParserInputBuffer(LIBXML2_2.4.30) [libXML2]	xmlFreePattern(LIBXML2_2.6.3) [libXML2]
xmlFreePatternList(LIBXML2_2.6.3) [libXML2]	xmlFreeProp(LIBXML2_2.4.30) [libXML2]	xmlFreePropList(LIBXML2_2.4.30) [libXML2]
xmlFreeRMutex(LIBXML2_2.4.30) [libXML2]	xmlFreeRefTable(LIBXML2_2.4.30) [libXML2]	xmlFreeStreamCtxt(LIBXML2_2.6.18) [libXML2]
xmlFreeTextReader(LIBXML2_2.4.30) [libXML2]	xmlFreeTextWriter(LIBXML2_2.6.0) [libXML2]	xmlFreeURI(LIBXML2_2.4.30) [libXML2]
xmlFreeValidCtxt(LIBXML2_2.5.8) [libXML2]	xmlGcMemGet(LIBXML2_2.5.7) [libXML2]	xmlGcMemSetup(LIBXML2_2.5.7) [libXML2]
xmlGetBufferAllocationScheme(LIBXML2_2.4.30) [libXML2]	xmlGetCharEncodingHandler(LIBXML2_2.4.30) [libXML2]	xmlGetCharEncodingName(LIBXML2_2.4.30) [libXML2]
xmlGetCompressMode(LIBXML2_2.4.30) [libXML2]	xmlGetDocCompressMode(LIBXML2_2.4.30) [libXML2]	xmlGetDocEntity(LIBXML2_2.4.30) [libXML2]
xmlGetDtdAttrDesc(LIBXML2_2.4.30) [libXML2]	xmlGetDtdElementDesc(LIBXML2_2.4.30) [libXML2]	xmlGetDtdEntity(LIBXML2_2.4.30) [libXML2]
xmlGetDtdNotationDesc(LIBXML2_2.4.30) [libXML2]	xmlGetDtdQAttrDesc(LIBXML2_2.4.30) [libXML2]	xmlGetDtdQElementDesc(LIBXML2_2.4.30) [libXML2]
xmlGetEncodingAlias(LIBXML2_2.4.30) [libXML2]	xmlGetExternalEntityLoader(LIBXML2_2.4.30) [libXML2]	xmlGetGlobalState(LIBXML2_2.4.30) [libXML2]

xmlGetID(LIBXML2_2.4.30) [libXML2]	xmlGetIntSubset(LIBXML2_2.4.30) [libXML2]	xmlGetLastChild(LIBXML2_2.4.30) [libXML2]
xmlGetLastError(LIBXML2_2.6.0) [libXML2]	xmlGetLineNo(LIBXML2_2.4.30) [libXML2]	xmlGetNoNsProp(LIBXML2_2.5.2) [libXML2]
xmlGetNodePath(LIBXML2_2.4.30) [libXML2]	xmlGetNsList(LIBXML2_2.4.30) [libXML2]	xmlGetNsProp(LIBXML2_2.4.30) [libXML2]
xmlGetParameterEntity(LIBXML2_2.4.30) [libXML2]	xmlGetPredefinedEntity(LIBXML2_2.4.30) [libXML2]	xmlGetProp(LIBXML2_2.4.30) [libXML2]
xmlGetRefs(LIBXML2_2.4.30) [libXML2]	xmlGetThreadId(LIBXML2_2.4.30) [libXML2]	xmlGetUTF8Char(LIBXML2_2.4.30) [libXML2]
xmlHasFeature(LIBXML2_2.6.21) [libXML2]	xmlHasNsProp(LIBXML2_2.4.30) [libXML2]	xmlHasProp(LIBXML2_2.4.30) [libXML2]
xmlHashAddEntry(LIBXML2_2.4.30) [libXML2]	xmlHashAddEntry2(LIBXML2_2.4.30) [libXML2]	xmlHashAddEntry3(LIBXML2_2.4.30) [libXML2]
xmlHashCopy(LIBXML2_2.4.30) [libXML2]	xmlHashCreate(LIBXML2_2.4.30) [libXML2]	xmlHashCreateDict(LIBXML2_2.6.18) [libXML2]
xmlHashFree(LIBXML2_2.4.30) [libXML2]	xmlHashLookup(LIBXML2_2.4.30) [libXML2]	xmlHashLookup2(LIBXML2_2.4.30) [libXML2]
xmlHashLookup3(LIBXML2_2.4.30) [libXML2]	xmlHashQLookup(LIBXML2_2.6.0) [libXML2]	xmlHashQLookup2(LIBXML2_2.6.0) [libXML2]
xmlHashQLookup3(LIBXML2_2.6.0) [libXML2]	xmlHashRemoveEntry(LIBXML2_2.4.30) [libXML2]	xmlHashRemoveEntry2(LIBXML2_2.4.30) [libXML2]
xmlHashRemoveEntry3(LIBXML2_2.4.30) [libXML2]	xmlHashScan(LIBXML2_2.4.30) [libXML2]	xmlHashScan3(LIBXML2_2.4.30) [libXML2]
xmlHashScanFull(LIBXML2_2.4.30) [libXML2]	xmlHashScanFull3(LIBXML2_2.4.30) [libXML2]	xmlHashSize(LIBXML2_2.4.30) [libXML2]
xmlHashUpdateEntry(LIBXML2_2.4.30) [libXML2]	xmlHashUpdateEntry2(LIBXML2_2.4.30) [libXML2]	xmlHashUpdateEntry3(LIBXML2_2.4.30) [libXML2]
xmlIOFTPClose(LIBXML2_2.4.30) [libXML2]	xmlIOFTPMatch(LIBXML2_2.4.30) [libXML2]	xmlIOFTPOpen(LIBXML2_2.4.30) [libXML2]
xmlIOFTPRead(LIBXML2_2.4.30) [libXML2]	xmlIOHTTPClose(LIBXML2_2.4.30) [libXML2]	xmlIOHTTPMatch(LIBXML2_2.4.30) [libXML2]
xmlIOHTTPOpen(LIBXML2_2.4.30) [libXML2]	xmlIOHTTPOpenW(LIBXML2_2.4.30) [libXML2]	xmlIOHTTPRead(LIBXML2_2.4.30) [libXML2]

xmlIOParseDTD(LIBXML2_2.4.30) [libXML2]	xmlInitCharEncodingHandlers(LIBXML2_2.4.30) [libXML2]	xmlInitGlobals(LIBXML2_2.5.8) [libXML2]
xmlInitMemory(LIBXML2_2.4.30) [libXML2]	xmlInitNodeInfoSeq(LIBXML2_2.4.30) [libXML2]	xmlInitParser(LIBXML2_2.4.30) [libXML2]
xmlInitParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlInitThreads(LIBXML2_2.4.30) [libXML2]	xmlInitializeCatalog(LIBXML2_2.4.30) [libXML2]
xmlInitializeGlobalState(LIBXML2_2.4.30) [libXML2]	xmlIsBlankNode(LIBXML2_2.4.30) [libXML2]	xmlIsID(LIBXML2_2.4.30) [libXML2]
xmlIsLetter(LIBXML2_2.4.30) [libXML2]	xmlIsMainThread(LIBXML2_2.4.30) [libXML2]	xmlIsMixedElement(LIBXML2_2.4.30) [libXML2]
xmlIsRef(LIBXML2_2.4.30) [libXML2]	xmlIsXHTML(LIBXML2_2.4.30) [libXML2]	xmlKeepBlanksDefault(LIBXML2_2.4.30) [libXML2]
xmlLineNumbersDefault(LIBXML2_2.4.30) [libXML2]	xmlLinkGetData(LIBXML2_2.4.30) [libXML2]	xmlListAppend(LIBXML2_2.4.30) [libXML2]
xmlListClear(LIBXML2_2.4.30) [libXML2]	xmlListCopy(LIBXML2_2.4.30) [libXML2]	xmlListCreate(LIBXML2_2.4.30) [libXML2]
xmlListDelete(LIBXML2_2.4.30) [libXML2]	xmlListDup(LIBXML2_2.4.30) [libXML2]	xmlListEmpty(LIBXML2_2.4.30) [libXML2]
xmlListEnd(LIBXML2_2.4.30) [libXML2]	xmlListFront(LIBXML2_2.4.30) [libXML2]	xmlListInsert(LIBXML2_2.4.30) [libXML2]
xmlListMerge(LIBXML2_2.4.30) [libXML2]	xmlListPopBack(LIBXML2_2.4.30) [libXML2]	xmlListPopFront(LIBXML2_2.4.30) [libXML2]
xmlListPushBack(LIBXML2_2.4.30) [libXML2]	xmlListPushFront(LIBXML2_2.4.30) [libXML2]	xmlListRemoveAll(LIBXML2_2.4.30) [libXML2]
xmlListRemoveFirst(LIBXML2_2.4.30) [libXML2]	xmlListRemoveLast(LIBXML2_2.4.30) [libXML2]	xmlListReverse(LIBXML2_2.4.30) [libXML2]
xmlListReverseSearch(LIBXML2_2.4.30) [libXML2]	xmlListReverseWalk(LIBXML2_2.4.30) [libXML2]	xmlListSearch(LIBXML2_2.4.30) [libXML2]
xmlListSize(LIBXML2_2.4.30) [libXML2]	xmlListSort(LIBXML2_2.4.30) [libXML2]	xmlListWalk(LIBXML2_2.4.30) [libXML2]
xmlLoadACatalog(LIBXML2_2.4.30) [libXML2]	xmlLoadCatalog(LIBXML2_2.4.30) [libXML2]	xmlLoadCatalogs(LIBXML2_2.4.30) [libXML2]
xmlLoadExternalEntity(LIBXML2_2.4.30) [libXML2]	xmlLoadSGMLSuperCatalog(LIBXML2_2.4.30) [libXML2]	xmlLockLibrary(LIBXML2_2.4.30) [libXML2]

xmlLsCountNode(LIBXML2_2.4.30) [libXML2]	xmlLsOneNode(LIBXML2_2.4.30) [libXML2]	xmlMallocAtomicLoc(LIBXML2_2.5.9) [libXML2]
xmlMallocLoc(LIBXML2_2.4.30) [libXML2]	xmlMemBlocks(LIBXML2_2.6.16) [libXML2]	xmlMemDisplay(LIBXML2_2.4.30) [libXML2]
xmlMemFree(LIBXML2_2.4.30) [libXML2]	xmlMemGet(LIBXML2_2.4.30) [libXML2]	xmlMemMalloc(LIBXML2_2.4.30) [libXML2]
xmlMemRealloc(LIBXML2_2.4.30) [libXML2]	xmlMemSetup(LIBXML2_2.4.30) [libXML2]	xmlMemShow(LIBXML2_2.4.30) [libXML2]
xmlMemStrdupLoc(LIBXML2_2.4.30) [libXML2]	xmlMemUsed(LIBXML2_2.4.30) [libXML2]	xmlMemoryDump(LIBXML2_2.4.30) [libXML2]
xmlMemoryStrdup(LIBXML2_2.4.30) [libXML2]	xmlModuleClose(LIBXML2_2.6.17) [libXML2]	xmlModuleFree(LIBXML2_2.6.17) [libXML2]
xmlModuleOpen(LIBXML2_2.6.17) [libXML2]	xmlModuleSymbol(LIBXML2_2.6.17) [libXML2]	xmlMutexLock(LIBXML2_2.4.30) [libXML2]
xmlMutexUnlock(LIBXML2_2.4.30) [libXML2]	xmlNewAutomata(LIBXML2_2.4.30) [libXML2]	xmlNewCDataBlock(LIBXML2_2.4.30) [libXML2]
xmlNewCatalog(LIBXML2_2.4.30) [libXML2]	xmlNewCharEncodingHandler(LIBXML2_2.4.30) [libXML2]	xmlNewCharRef(LIBXML2_2.4.30) [libXML2]
xmlNewChild(LIBXML2_2.4.30) [libXML2]	xmlNewComment(LIBXML2_2.4.30) [libXML2]	xmlNewDoc(LIBXML2_2.4.30) [libXML2]
xmlNewDocComment(LIBXML2_2.4.30) [libXML2]	xmlNewDocElementContent(LIBXML2_2.6.18) [libXML2]	xmlNewDocFragment(LIBXML2_2.4.30) [libXML2]
xmlNewDocNode(LIBXML2_2.4.30) [libXML2]	xmlNewDocNodeEatName(LIBXML2_2.4.30) [libXML2]	xmlNewDocPI(LIBXML2_2.6.15) [libXML2]
xmlNewDocProp(LIBXML2_2.4.30) [libXML2]	xmlNewDocRawNode(LIBXML2_2.4.30) [libXML2]	xmlNewDocText(LIBXML2_2.4.30) [libXML2]
xmlNewDocTextLen(LIBXML2_2.4.30) [libXML2]	xmlNewDtd(LIBXML2_2.4.30) [libXML2]	xmlNewEntityInputStream(LIBXML2_2.4.30) [libXML2]
xmlNewIOInputStream(LIBXML2_2.4.30) [libXML2]	xmlNewInputFromFile(LIBXML2_2.4.30) [libXML2]	xmlNewInputStream(LIBXML2_2.4.30) [libXML2]
xmlNewMutex(LIBXML2_2.4.30) [libXML2]	xmlNewNode(LIBXML2_2.4.30) [libXML2]	xmlNewNodeEatName(LIBXML2_2.4.30) [libXML2]

xmlNewNs(LIBXML2_2.4.30) [libXML2]	xmlNewNsProp(LIBXML2_2.4.30) [libXML2]	xmlNewNsPropEatName(LIBXML2_2.4.30) [libXML2]
xmlNewPI(LIBXML2_2.4.30) [libXML2]	xmlNewParserCtxt(LIBXML2_2.4.30) [libXML2]	xmlNewProp(LIBXML2_2.4.30) [libXML2]
xmlNewRMutex(LIBXML2_2.4.30) [libXML2]	xmlNewReference(LIBXML2_2.4.30) [libXML2]	xmlNewStringInputStream(LIBXML2_2.4.30) [libXML2]
xmlNewText(LIBXML2_2.4.30) [libXML2]	xmlNewTextChild(LIBXML2_2.4.30) [libXML2]	xmlNewTextLen(LIBXML2_2.4.30) [libXML2]
xmlNewTextReader(LIBXML2_2.4.30) [libXML2]	xmlNewTextReaderFilename(LIBXML2_2.4.30) [libXML2]	xmlNewTextWriter(LIBXML2_2.6.0) [libXML2]
xmlNewTextWriterDoc(LIBXML2_2.6.3) [libXML2]	xmlNewTextWriterFilename(LIBXML2_2.6.0) [libXML2]	xmlNewTextWriterMemory(LIBXML2_2.6.0) [libXML2]
xmlNewTextWriterPushParser(LIBXML2_2.6.3) [libXML2]	xmlNewTextWriterTree(LIBXML2_2.6.3) [libXML2]	xmlNewValidCtxt(LIBXML2_2.5.8) [libXML2]
xmlNextChar(LIBXML2_2.4.30) [libXML2]	xmlNoNetExternalEntityLoader(LIBXML2_2.4.30) [libXML2]	xmlNodeAddContent(LIBXML2_2.4.30) [libXML2]
xmlNodeAddContentLen(LIBXML2_2.4.30) [libXML2]	xmlNodeBufGetContent(LIBXML2_2.6.0) [libXML2]	xmlNodeDump(LIBXML2_2.4.30) [libXML2]
xmlNodeDumpOutput(LIBXML2_2.4.30) [libXML2]	xmlNodeGetBase(LIBXML2_2.4.30) [libXML2]	xmlNodeGetContent(LIBXML2_2.4.30) [libXML2]
xmlNodeGetLang(LIBXML2_2.4.30) [libXML2]	xmlNodeGetSpacePreserve(LIBXML2_2.4.30) [libXML2]	xmlNodeIsText(LIBXML2_2.4.30) [libXML2]
xmlNodeListGetRawString(LIBXML2_2.4.30) [libXML2]	xmlNodeListGetString(LIBXML2_2.4.30) [libXML2]	xmlNodeSetBase(LIBXML2_2.4.30) [libXML2]
xmlNodeSetContent(LIBXML2_2.4.30) [libXML2]	xmlNodeSetContentLen(LIBXML2_2.4.30) [libXML2]	xmlNodeSetLang(LIBXML2_2.4.30) [libXML2]
xmlNodeSetName(LIBXML2_2.4.30) [libXML2]	xmlNodeSetSpacePreserve(LIBXML2_2.4.30) [libXML2]	xmlNormalizeURIPath(LIBXML2_2.4.30) [libXML2]
xmlNormalizeWindowsPath(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferClose(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferCreateFd(LIBXML2_2.4.30) [libXML2]

xmlOutputBufferCreateFile(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferCreateFilename(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferCreateFilenameDefault(LIBXML2_2.6.11) [libXML2]
xmlOutputBufferCreateIO(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferFlush(LIBXML2_2.4.30) [libXML2]	xmlOutputBufferWrite(LIBXML2_2.4.30) [libXML2]
xmlOutputBufferWriteEscape(LIBXML2_2.6.10) [libXML2]	xmlOutputBufferWriteString(LIBXML2_2.4.30) [libXML2]	xmlParseAttValue(LIBXML2_2.4.30) [libXML2]
xmlParseAttribute(LIBXML2_2.4.30) [libXML2]	xmlParseAttributeListDecl(LIBXML2_2.4.30) [libXML2]	xmlParseAttributeType(LIBXML2_2.4.30) [libXML2]
xmlParseBalancedChunkMemory(LIBXML2_2.4.30) [libXML2]	xmlParseBalancedChunkMemoryRecover(LIBXML2_2.4.30) [libXML2]	xmlParseCDSect(LIBXML2_2.4.30) [libXML2]
xmlParseCatalogFile(LIBXML2_2.4.30) [libXML2]	xmlParseCharData(LIBXML2_2.4.30) [libXML2]	xmlParseCharEncoding(LIBXML2_2.4.30) [libXML2]
xmlParseCharRef(LIBXML2_2.4.30) [libXML2]	xmlParseChunk(LIBXML2_2.4.30) [libXML2]	xmlParseComment(LIBXML2_2.4.30) [libXML2]
xmlParseContent(LIBXML2_2.4.30) [libXML2]	xmlParseCtxtExternalEntity(LIBXML2_2.4.30) [libXML2]	xmlParseDTD(LIBXML2_2.4.30) [libXML2]
xmlParseDefaultDecl(LIBXML2_2.4.30) [libXML2]	xmlParseDoc(LIBXML2_2.4.30) [libXML2]	xmlParseDocTypeDecl(LIBXML2_2.4.30) [libXML2]
xmlParseDocument(LIBXML2_2.4.30) [libXML2]	xmlParseElement(LIBXML2_2.4.30) [libXML2]	xmlParseElementChildrenContentDecl(LIBXML2_2.4.30) [libXML2]
xmlParseElementContentDecl(LIBXML2_2.4.30) [libXML2]	xmlParseElementDecl(LIBXML2_2.4.30) [libXML2]	xmlParseElementMixedContentDecl(LIBXML2_2.4.30) [libXML2]
xmlParseEncName(LIBXML2_2.4.30) [libXML2]	xmlParseEncodingDecl(LIBXML2_2.4.30) [libXML2]	xmlParseEndTag(LIBXML2_2.4.30) [libXML2]
xmlParseEntity(LIBXML2_2.4.30) [libXML2]	xmlParseEntityDecl(LIBXML2_2.4.30) [libXML2]	xmlParseEntityRef(LIBXML2_2.4.30) [libXML2]
xmlParseEntityValue(LIBXML2_2.4.30) [libXML2]	xmlParseEnumeratedType(LIBXML2_2.4.30) [libXML2]	xmlParseEnumerationType(LIBXML2_2.4.30) [libXML2]
xmlParseExtParsedEnt(LIBXML2_2.4.30) [libXML2]	xmlParseExternalEntity(LIBXML2_2.4.30) [libXML2]	xmlParseExternalID(LIBXML2_2.4.30) [libXML2]

xmlParseExternalSubset (LIBXML2_2.4.30) [libXML2]	xmlParseFile(LIBXML2_2.4.30) [libXML2]	xmlParseInNodeContext(LIBXML2_2.6.12) [libXML2]
xmlParseMarkupDecl(LIBXML2_2.4.30) [libXML2]	xmlParseMemory(LIBXML2_2.4.30) [libXML2]	xmlParseMisc(LIBXML2_2.4.30) [libXML2]
xmlParseName(LIBXML2_2.4.30) [libXML2]	xmlParseNmtoken(LIBXML2_2.4.30) [libXML2]	xmlParseNotationDecl(LIBXML2_2.4.30) [libXML2]
xmlParseNotationType(LIBXML2_2.4.30) [libXML2]	xmlParsePEReference(LIBXML2_2.4.30) [libXML2]	xmlParsePI(LIBXML2_2.4.30) [libXML2]
xmlParsePITarget(LIBXML2_2.4.30) [libXML2]	xmlParsePubidLiteral(LIBXML2_2.4.30) [libXML2]	xmlParseReference(LIBXML2_2.4.30) [libXML2]
xmlParseSDDecl(LIBXML2_2.4.30) [libXML2]	xmlParseStartTag(LIBXML2_2.4.30) [libXML2]	xmlParseSystemLiteral(LIBXML2_2.4.30) [libXML2]
xmlParseTextDecl(LIBXML2_2.4.30) [libXML2]	xmlParseURI(LIBXML2_2.4.30) [libXML2]	xmlParseURIRaw(LIBXML2_2.6.21) [libXML2]
xmlParseURIReference(LIBXML2_2.4.30) [libXML2]	xmlParseVersionInfo(LIBXML2_2.4.30) [libXML2]	xmlParseVersionNum(LIBXML2_2.4.30) [libXML2]
xmlParseXMLDecl(LIBXML2_2.4.30) [libXML2]	xmlParserAddNodeInfo(LIBXML2_2.4.30) [libXML2]	xmlParserError(LIBXML2_2.4.30) [libXML2]
xmlParserFindNodeInfo(LIBXML2_2.4.30) [libXML2]	xmlParserFindNodeInfoIndex(LIBXML2_2.4.30) [libXML2]	xmlParserGetDirectory(LIBXML2_2.4.30) [libXML2]
xmlParserHandlePEReference(LIBXML2_2.4.30) [libXML2]	xmlParserInputBufferCreateFd(LIBXML2_2.4.30) [libXML2]	xmlParserInputBufferCreateFile(LIBXML2_2.4.30) [libXML2]
xmlParserInputBufferCreateFilename(LIBXML2_2.4.30) [libXML2]	xmlParserInputBufferCreateFilenameDefault(LIBXML2_2.6.11) [libXML2]	xmlParserInputBufferCreateIO(LIBXML2_2.4.30) [libXML2]
xmlParserInputBufferCreateMem(LIBXML2_2.4.30) [libXML2]	xmlParserInputBufferCreateStatic(LIBXML2_2.6.0) [libXML2]	xmlParserInputBufferGrow(LIBXML2_2.4.30) [libXML2]
xmlParserInputBufferPush(LIBXML2_2.4.30) [libXML2]	xmlParserInputBufferRead(LIBXML2_2.4.30) [libXML2]	xmlParserInputGrow(LIBXML2_2.4.30) [libXML2]
xmlParserInputRead(LIBXML2_2.4.30) [libXML2]	xmlParserInputShrink(LIBXML2_2.4.30) [libXML2]	xmlParserPrintFileContext(LIBXML2_2.4.30) [libXML2]

xmlParserPrintFileInfo(LIBXML2_2.4.30) [libXML2]	xmlParserValidityError(LIBXML2_2.4.30) [libXML2]	xmlParserValidityWarning(LIBXML2_2.4.30) [libXML2]
xmlParserWarning(LIBXML2_2.4.30) [libXML2]	xmlPatternFromRoot(LIBXML2_2.6.18) [libXML2]	xmlPatternGetStreamCtxt(LIBXML2_2.6.18) [libXML2]
xmlPatternMatch(LIBXML2_2.6.3) [libXML2]	xmlPatternMaxDepth(LIBXML2_2.6.18) [libXML2]	xmlPatternMinDepth(LIBXML2_2.6.21) [libXML2]
xmlPatternStreamable(LIBXML2_2.6.18) [libXML2]	xmlPatterncompile(LIBXML2_2.6.3) [libXML2]	xmlPedanticParserDefault(LIBXML2_2.4.30) [libXML2]
xmlPopInput(LIBXML2_2.4.30) [libXML2]	xmlPopInputCallbacks(LIBXML2_2.6.10) [libXML2]	xmlPrintURI(LIBXML2_2.4.30) [libXML2]
xmlPushInput(LIBXML2_2.4.30) [libXML2]	xmlRMutexLock(LIBXML2_2.4.30) [libXML2]	xmlRMutexUnlock(LIBXML2_2.4.30) [libXML2]
xmlReadDoc(LIBXML2_2.6.0) [libXML2]	xmlReadFd(LIBXML2_2.6.0) [libXML2]	xmlReadFile(LIBXML2_2.6.0) [libXML2]
xmlReadIO(LIBXML2_2.6.0) [libXML2]	xmlReadMemory(LIBXML2_2.6.0) [libXML2]	xmlReaderForDoc(LIBXML2_2.6.0) [libXML2]
xmlReaderForFd(LIBXML2_2.6.0) [libXML2]	xmlReaderForFile(LIBXML2_2.6.0) [libXML2]	xmlReaderForIO(LIBXML2_2.6.0) [libXML2]
xmlReaderForMemory(LIBXML2_2.6.0) [libXML2]	xmlReaderNewDoc(LIBXML2_2.6.0) [libXML2]	xmlReaderNewFd(LIBXML2_2.6.0) [libXML2]
xmlReaderNewFile(LIBXML2_2.6.0) [libXML2]	xmlReaderNewIO(LIBXML2_2.6.0) [libXML2]	xmlReaderNewMemory(LIBXML2_2.6.0) [libXML2]
xmlReaderNewWalker(LIBXML2_2.6.0) [libXML2]	xmlReaderWalker(LIBXML2_2.6.0) [libXML2]	xmlReallocLoc(LIBXML2_2.4.30) [libXML2]
xmlReconciliateNs(LIBXML2_2.4.30) [libXML2]	xmlRecoverDoc(LIBXML2_2.4.30) [libXML2]	xmlRecoverFile(LIBXML2_2.4.30) [libXML2]
xmlRecoverMemory(LIBXML2_2.4.30) [libXML2]	xmlRegExecErrInfo(LIBXML2_2.6.17) [libXML2]	xmlRegExecNextValues(LIBXML2_2.6.17) [libXML2]
xmlRegExecPushString(LIBXML2_2.4.30) [libXML2]	xmlRegExecPushString2(LIBXML2_2.5.7) [libXML2]	xmlRegFreeExecCtxt(LIBXML2_2.4.30) [libXML2]
xmlRegFreeRegexp(LIBXML2_2.4.30) [libXML2]	xmlRegNewExecCtxt(LIBXML2_2.4.30) [libXML2]	xmlRegexpCompile(LIBXML2_2.4.30) [libXML2]

xmlRegexpExec(LIBXML2_2.4.30) [libXML2]	xmlRegexpIsDeterminist(LIBXML2_2.4.30) [libXML2]	xmlRegexpPrint(LIBXML2_2.4.30) [libXML2]
xmlRegisterCharEncodingHandler(LIBXML2_2.4.30) [libXML2]	xmlRegisterDefaultInputCallbacks(LIBXML2_2.4.30) [libXML2]	xmlRegisterDefaultOutputCallbacks(LIBXML2_2.4.30) [libXML2]
xmlRegisterHTTPPostCallbacks(LIBXML2_2.4.30) [libXML2]	xmlRegisterInputCallbacks(LIBXML2_2.4.30) [libXML2]	xmlRegisterNodeDefault(LIBXML2_2.5.0) [libXML2]
xmlRegisterOutputCallbacks(LIBXML2_2.4.30) [libXML2]	xmlRelaxNGCleanupTypes(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGDump(LIBXML2_2.5.2) [libXML2]
xmlRelaxNGDumpTree(LIBXML2_2.5.4) [libXML2]	xmlRelaxNGFree(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGFreeParserCtxt(LIBXML2_2.5.2) [libXML2]
xmlRelaxNGFreeValidCtxt(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGGetParserErrors(LIBXML2_2.5.9) [libXML2]	xmlRelaxNGGetValidErrors(LIBXML2_2.5.9) [libXML2]
xmlRelaxNGInitTypes(LIBXML2_2.6.16) [libXML2]	xmlRelaxNGNewDocParserCtxt(LIBXML2_2.5.7) [libXML2]	xmlRelaxNGNewMemParserCtxt(LIBXML2_2.5.2) [libXML2]
xmlRelaxNGNewParserCtxt(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGNewValidCtxt(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGParse(LIBXML2_2.5.2) [libXML2]
xmlRelaxNGSetParserErrors(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGSetValidErrors(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGSetValidStructuredErrors(LIBXML2_2.6.21) [libXML2]
xmlRelaxNGValidateDoc(LIBXML2_2.5.2) [libXML2]	xmlRelaxNGValidateFullElement(LIBXML2_2.5.7) [libXML2]	xmlRelaxNGValidatePopElement(LIBXML2_2.5.7) [libXML2]
xmlRelaxNGValidatePushCData(LIBXML2_2.5.7) [libXML2]	xmlRelaxNGValidatePushElement(LIBXML2_2.5.7) [libXML2]	xmlRelaxParserSetFlag(LIBXML2_2.6.5) [libXML2]
xmlRemoveID(LIBXML2_2.4.30) [libXML2]	xmlRemoveProp(LIBXML2_2.4.30) [libXML2]	xmlRemoveRef(LIBXML2_2.4.30) [libXML2]
xmlReplaceNode(LIBXML2_2.4.30) [libXML2]	xmlResetError(LIBXML2_2.6.0) [libXML2]	xmlResetLastError(LIBXML2_2.6.0) [libXML2]
xmlSAX2AttributeDecl(LIBXML2_2.6.0) [libXML2]	xmlSAX2CDataBlock(LIBXML2_2.6.0) [libXML2]	xmlSAX2Characters(LIBXML2_2.6.0) [libXML2]
xmlSAX2Comment(LIBXML2_2.6.0) [libXML2]	xmlSAX2ElementDecl(LIBXML2_2.6.0) [libXML2]	xmlSAX2EndDocument(LIBXML2_2.6.0) [libXML2]

xmlSAX2EndElement(LIBXML2_2.6.0) [libXML2]	xmlSAX2EndElementNs(LIBXML2_2.6.0) [libXML2]	xmlSAX2EntityDecl(LIBXML2_2.6.0) [libXML2]
xmlSAX2ExternalSubset(LIBXML2_2.6.0) [libXML2]	xmlSAX2GetColumnNumber(LIBXML2_2.6.0) [libXML2]	xmlSAX2GetEntity(LIBXML2_2.6.0) [libXML2]
xmlSAX2GetLineNumber(LIBXML2_2.6.0) [libXML2]	xmlSAX2GetParameterEntity(LIBXML2_2.6.0) [libXML2]	xmlSAX2GetPublicId(LIBXML2_2.6.0) [libXML2]
xmlSAX2GetSystemId(LIBXML2_2.6.0) [libXML2]	xmlSAX2HasExternalSubset(LIBXML2_2.6.0) [libXML2]	xmlSAX2HasInternalSubset(LIBXML2_2.6.0) [libXML2]
xmlSAX2IgnorableWhitespace(LIBXML2_2.6.0) [libXML2]	xmlSAX2InitDefaultSAXHandler(LIBXML2_2.6.0) [libXML2]	xmlSAX2InitDocbDefaultSAXHandler(LIBXML2_2.6.0) [libXML2]
xmlSAX2InitHtmlDefaultSAXHandler(LIBXML2_2.6.0) [libXML2]	xmlSAX2InternalSubset(LIBXML2_2.6.0) [libXML2]	xmlSAX2IsStandalone(LIBXML2_2.6.0) [libXML2]
xmlSAX2NotationDecl(LIBXML2_2.6.0) [libXML2]	xmlSAX2ProcessingInstruction(LIBXML2_2.6.0) [libXML2]	xmlSAX2Reference(LIBXML2_2.6.0) [libXML2]
xmlSAX2ResolveEntity(LIBXML2_2.6.0) [libXML2]	xmlSAX2SetDocumentLocator(LIBXML2_2.6.0) [libXML2]	xmlSAX2StartDocument(LIBXML2_2.6.0) [libXML2]
xmlSAX2StartElement(LIBXML2_2.6.0) [libXML2]	xmlSAX2StartElementNs(LIBXML2_2.6.0) [libXML2]	xmlSAX2UnparsedEntityDecl(LIBXML2_2.6.0) [libXML2]
xmlSAXDefaultVersion(LIBXML2_2.6.0) [libXML2]	xmlSAXParseDTD(LIBXML2_2.4.30) [libXML2]	xmlSAXParseDoc(LIBXML2_2.4.30) [libXML2]
xmlSAXParseEntity(LIBXML2_2.4.30) [libXML2]	xmlSAXParseFile(LIBXML2_2.4.30) [libXML2]	xmlSAXParseFileWithData(LIBXML2_2.4.30) [libXML2]
xmlSAXParseMemory(LIBXML2_2.4.30) [libXML2]	xmlSAXParseMemoryWithData(LIBXML2_2.4.30) [libXML2]	xmlSAXUserParseFile(LIBXML2_2.4.30) [libXML2]
xmlSAXUserParseMemory(LIBXML2_2.4.30) [libXML2]	xmlSAXVersion(LIBXML2_2.6.0) [libXML2]	xmlSaveClose(LIBXML2_2.6.8) [libXML2]
xmlSaveDoc(LIBXML2_2.6.8) [libXML2]	xmlSaveFile(LIBXML2_2.4.30) [libXML2]	xmlSaveFileEnc(LIBXML2_2.4.30) [libXML2]
xmlSaveFileTo(LIBXML2_2.4.30) [libXML2]	xmlSaveFlush(LIBXML2_2.6.8) [libXML2]	xmlSaveFormatFile(LIBXML2_2.4.30) [libXML2]

xmlSaveFormatFileEnc(LIBXML2_2.4.30) [libXML2]	xmlSaveFormatFileTo(LIBXML2_2.4.30) [libXML2]	xmlSaveSetAttrEscape(LIBXML2_2.6.10) [libXML2]
xmlSaveSetEscape(LIBXML2_2.6.10) [libXML2]	xmlSaveToFd(LIBXML2_2.6.8) [libXML2]	xmlSaveToFilename(LIBXML2_2.6.8) [libXML2]
xmlSaveToIO(LIBXML2_2.6.8) [libXML2]	xmlSaveTree(LIBXML2_2.6.8) [libXML2]	xmlSaveUri(LIBXML2_2.4.30) [libXML2]
xmlSchemaCleanupTypes(LIBXML2_2.5.8) [libXML2]	xmlSchemaCollapseString(LIBXML2_2.6.11) [libXML2]	xmlSchemaCompareValues(LIBXML2_2.5.8) [libXML2]
xmlSchemaDump(LIBXML2_2.5.8) [libXML2]	xmlSchemaFree(LIBXML2_2.5.8) [libXML2]	xmlSchemaFreeParserCtxt(LIBXML2_2.5.8) [libXML2]
xmlSchemaFreeValidCtxt(LIBXML2_2.5.8) [libXML2]	xmlSchemaFreeValue(LIBXML2_2.5.8) [libXML2]	xmlSchemaGetBuiltInType(LIBXML2_2.6.11) [libXML2]
xmlSchemaGetCanonValue(LIBXML2_2.6.18) [libXML2]	xmlSchemaGetParserErrors(LIBXML2_2.6.12) [libXML2]	xmlSchemaGetValType(LIBXML2_2.6.19) [libXML2]
xmlSchemaGetValidErrors(LIBXML2_2.6.12) [libXML2]	xmlSchemaInitTypes(LIBXML2_2.5.8) [libXML2]	xmlSchemaIsValid(LIBXML2_2.6.20) [libXML2]
xmlSchemaNewDocParserCtxt(LIBXML2_2.6.2) [libXML2]	xmlSchemaNewMemParserCtxt(LIBXML2_2.5.8) [libXML2]	xmlSchemaNewParserCtxt(LIBXML2_2.5.8) [libXML2]
xmlSchemaNewValidCtxt(LIBXML2_2.5.8) [libXML2]	xmlSchemaParse(LIBXML2_2.5.8) [libXML2]	xmlSchemaSAXPlug(LIBXML2_2.6.20) [libXML2]
xmlSchemaSAXUnplug(LIBXML2_2.6.20) [libXML2]	xmlSchemaSetParserErrors(LIBXML2_2.5.8) [libXML2]	xmlSchemaSetValidErrors(LIBXML2_2.5.8) [libXML2]
xmlSchemaSetValidOptions(LIBXML2_2.6.14) [libXML2]	xmlSchemaSetValidStructuredErrors(LIBXML2_2.6.21) [libXML2]	xmlSchemaValPredefTypeNode(LIBXML2_2.5.8) [libXML2]
xmlSchemaValidCtxtGetOptions(LIBXML2_2.6.14) [libXML2]	xmlSchemaValidateDoc(LIBXML2_2.5.8) [libXML2]	xmlSchemaValidateFile(LIBXML2_2.6.20) [libXML2]
xmlSchemaValidateOneElement(LIBXML2_2.6.14) [libXML2]	xmlSchemaValidateStream(LIBXML2_2.5.8) [libXML2]	xmlSchematronFree(LIBXML2_2.6.21) [libXML2]
xmlSchematronFreeParserCtxt(LIBXML2_2.6.21) [libXML2]	xmlSchematronFreeValidCtxt(LIBXML2_2.6.21) [libXML2]	xmlSchematronNewDocParserCtxt(LIBXML2_2.6.21) [libXML2]

xmlSchematronNewMemParserCtxt(LIBXML2_2.6.21) [libXML2]	xmlSchematronNewParserCtxt(LIBXML2_2.6.21) [libXML2]	xmlSchematronNewValidCtxt(LIBXML2_2.6.21) [libXML2]
xmlSchematronParse(LIBXML2_2.6.21) [libXML2]	xmlSchematronValidateDoc(LIBXML2_2.6.21) [libXML2]	xmlSearchNs(LIBXML2_2.4.30) [libXML2]
xmlSearchNsByHref(LIBXML2_2.4.30) [libXML2]	xmlSetBufferAllocationScheme(LIBXML2_2.4.30) [libXML2]	xmlSetCompressMode(LIBXML2_2.4.30) [libXML2]
xmlSetDocCompressMode(LIBXML2_2.4.30) [libXML2]	xmlSetEntityReferenceFunc(LIBXML2_2.4.30) [libXML2]	xmlSetExternalEntityLoader(LIBXML2_2.4.30) [libXML2]
xmlSetGenericErrorFunc(LIBXML2_2.4.30) [libXML2]	xmlSetListDoc(LIBXML2_2.4.30) [libXML2]	xmlSetNs(LIBXML2_2.4.30) [libXML2]
xmlSetNsProp(LIBXML2_2.4.30) [libXML2]	xmlSetProp(LIBXML2_2.4.30) [libXML2]	xmlSetStructuredErrorFunc(LIBXML2_2.6.0) [libXML2]
xmlSetTreeDoc(LIBXML2_2.4.30) [libXML2]	xmlSetupParserForBuffer(LIBXML2_2.4.30) [libXML2]	xmlShell(LIBXML2_2.4.30) [libXML2]
xmlShellBase(LIBXML2_2.4.30) [libXML2]	xmlShellCat(LIBXML2_2.4.30) [libXML2]	xmlShellDir(LIBXML2_2.4.30) [libXML2]
xmlShellDu(LIBXML2_2.4.30) [libXML2]	xmlShellList(LIBXML2_2.4.30) [libXML2]	xmlShellLoad(LIBXML2_2.4.30) [libXML2]
xmlShellPrintNode(LIBXML2_2.4.30) [libXML2]	xmlShellPrintXPathError(LIBXML2_2.4.30) [libXML2]	xmlShellPrintXPathResult(LIBXML2_2.4.30) [libXML2]
xmlShellPwd(LIBXML2_2.4.30) [libXML2]	xmlShellSave(LIBXML2_2.4.30) [libXML2]	xmlShellValidate(LIBXML2_2.4.30) [libXML2]
xmlShellWrite(LIBXML2_2.4.30) [libXML2]	xmlSkipBlankChars(LIBXML2_2.4.30) [libXML2]	xmlSnprintfElementContent(LIBXML2_2.4.30) [libXML2]
xmlSplitQName(LIBXML2_2.4.30) [libXML2]	xmlSplitQName2(LIBXML2_2.4.30) [libXML2]	xmlSplitQName3(LIBXML2_2.5.9) [libXML2]
xmlStopParser(LIBXML2_2.4.30) [libXML2]	xmlStrEqual(LIBXML2_2.4.30) [libXML2]	xmlStrPrintf(LIBXML2_2.6.0) [libXML2]
xmlStrQEqual(LIBXML2_2.6.0) [libXML2]	xmlStrVPrintf(LIBXML2_2.6.2) [libXML2]	xmlStrcasecmp(LIBXML2_2.4.30) [libXML2]
xmlStrcasestr(LIBXML2_2.4.30) [libXML2]	xmlStrcat(LIBXML2_2.4.30) [libXML2]	xmlStrchr(LIBXML2_2.4.30) [libXML2]
xmlStrcmp(LIBXML2_2.4.30) [libXML2]	xmlStrdup(LIBXML2_2.4.30) [libXML2]	xmlStreamPop(LIBXML2_2.6.18) [libXML2]

xmlStreamPush(LIBXML2_2.6.18) [libXML2]	xmlStreamPushAttr(LIBXML2_2.6.18) [libXML2]	xmlStringCurrentChar(LIBXML2_2.4.30) [libXML2]
xmlStringDecodeEntities(LIBXML2_2.4.30) [libXML2]	xmlStringGetNodeList(LIBXML2_2.4.30) [libXML2]	xmlStringLenDecodeEntities(LIBXML2_2.6.0) [libXML2]
xmlStringLenGetNodeList(LIBXML2_2.4.30) [libXML2]	xmlStrlen(LIBXML2_2.4.30) [libXML2]	xmlStrncasecmp(LIBXML2_2.4.30) [libXML2]
xmlStrncat(LIBXML2_2.4.30) [libXML2]	xmlStrncatNew(LIBXML2_2.6.5) [libXML2]	xmlStrncmp(LIBXML2_2.4.30) [libXML2]
xmlStrndup(LIBXML2_2.4.30) [libXML2]	xmlStrstr(LIBXML2_2.4.30) [libXML2]	xmlStrsub(LIBXML2_2.4.30) [libXML2]
xmlSubstituteEntitiesDefault(LIBXML2_2.4.30) [libXML2]	xmlSwitchEncoding(LIBXML2_2.4.30) [libXML2]	xmlSwitchInputEncoding(LIBXML2_2.6.0) [libXML2]
xmlSwitchToEncoding(LIBXML2_2.4.30) [libXML2]	xmlTextConcat(LIBXML2_2.4.30) [libXML2]	xmlTextMerge(LIBXML2_2.4.30) [libXML2]
xmlTextReaderAttributeCount(LIBXML2_2.4.30) [libXML2]	xmlTextReaderBaseUri(LIBXML2_2.4.30) [libXML2]	xmlTextReaderByteConsumed(LIBXML2_2.6.18) [libXML2]
xmlTextReaderClose(LIBXML2_2.5.0) [libXML2]	xmlTextReaderConstBaseUri(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstEncoding(LIBXML2_2.6.15) [libXML2]
xmlTextReaderConstLocalName(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstName(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstNamespaceUri(LIBXML2_2.6.0) [libXML2]
xmlTextReaderConstPrefix(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstString(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstValue(LIBXML2_2.6.0) [libXML2]
xmlTextReaderConstXmlLang(LIBXML2_2.6.0) [libXML2]	xmlTextReaderConstXmlVersion(LIBXML2_2.6.15) [libXML2]	xmlTextReaderCurrentDoc(LIBXML2_2.5.0) [libXML2]
xmlTextReaderCurrentNode(LIBXML2_2.5.0) [libXML2]	xmlTextReaderDepth(LIBXML2_2.4.30) [libXML2]	xmlTextReaderExpand(LIBXML2_2.5.7) [libXML2]
xmlTextReaderGetAttribute(LIBXML2_2.5.0) [libXML2]	xmlTextReaderGetAttributeNo(LIBXML2_2.5.0) [libXML2]	xmlTextReaderGetAttributeNs(LIBXML2_2.5.0) [libXML2]
xmlTextReaderGetErrorHandler(LIBXML2_2.5.2) [libXML2]	xmlTextReaderGetParserColumnNumber(LIBXML2_2.6.17) [libXML2]	xmlTextReaderGetParserLineNumber(LIBXML2_2.6.17) [libXML2]

xmlTextReaderGetParserProp(LIBXML2_2.5.0) [libXML2]	xmlTextReaderGetRemainder(LIBXML2_2.5.0) [libXML2]	xmlTextReaderHasAttributes(LIBXML2_2.4.30) [libXML2]
xmlTextReaderHasValue(LIBXML2_2.4.30) [libXML2]	xmlTextReaderIsDefault(LIBXML2_2.4.30) [libXML2]	xmlTextReaderIsEmptyElement(LIBXML2_2.4.30) [libXML2]
xmlTextReaderIsNamespaceDecl(LIBXML2_2.6.15) [libXML2]	xmlTextReaderIsValid(LIBXML2_2.5.7) [libXML2]	xmlTextReaderLocalName(LIBXML2_2.4.30) [libXML2]
xmlTextReaderLocatorBaseURI(LIBXML2_2.5.2) [libXML2]	xmlTextReaderLocatorLineNumber(LIBXML2_2.5.2) [libXML2]	xmlTextReaderLookupNamespace(LIBXML2_2.5.0) [libXML2]
xmlTextReaderMoveToAttribute(LIBXML2_2.5.0) [libXML2]	xmlTextReaderMoveToAttributeNo(LIBXML2_2.5.0) [libXML2]	xmlTextReaderMoveToAttributeNs(LIBXML2_2.5.0) [libXML2]
xmlTextReaderMoveToElement(LIBXML2_2.5.0) [libXML2]	xmlTextReaderMoveToFirstAttribute(LIBXML2_2.5.0) [libXML2]	xmlTextReaderMoveToNextAttribute(LIBXML2_2.5.0) [libXML2]
xmlTextReaderName(LIBXML2_2.4.30) [libXML2]	xmlTextReaderNamespaceUri(LIBXML2_2.4.30) [libXML2]	xmlTextReaderNext(LIBXML2_2.5.7) [libXML2]
xmlTextReaderNextSibling(LIBXML2_2.6.0) [libXML2]	xmlTextReaderNodeType(LIBXML2_2.4.30) [libXML2]	xmlTextReaderNormalization(LIBXML2_2.5.0) [libXML2]
xmlTextReaderPrefix(LIBXML2_2.4.30) [libXML2]	xmlTextReaderPreserve(LIBXML2_2.6.0) [libXML2]	xmlTextReaderPreservePattern(LIBXML2_2.6.3) [libXML2]
xmlTextReaderQuoteChar(LIBXML2_2.4.30) [libXML2]	xmlTextReaderRead(LIBXML2_2.4.30) [libXML2]	xmlTextReaderReadAttributeValue(LIBXML2_2.5.0) [libXML2]
xmlTextReaderReadInnerXml(LIBXML2_2.5.0) [libXML2]	xmlTextReaderReadOuterXml(LIBXML2_2.5.0) [libXML2]	xmlTextReaderReadState(LIBXML2_2.5.0) [libXML2]
xmlTextReaderReadString(LIBXML2_2.5.0) [libXML2]	xmlTextReaderRelaxNGSetSchema(LIBXML2_2.5.7) [libXML2]	xmlTextReaderRelaxNGValidate(LIBXML2_2.5.7) [libXML2]
xmlTextReaderSchemaValidate(LIBXML2_2.6.20) [libXML2]	xmlTextReaderSetErrorHandler(LIBXML2_2.5.2) [libXML2]	xmlTextReaderSetParserProp(LIBXML2_2.5.0) [libXML2]
xmlTextReaderSetSchema(LIBXML2_2.6.20) [libXML2]	xmlTextReaderSetStructuredErrorHandler(LIBXML2_2.6.6) [libXML2]	xmlTextReaderStandalone(LIBXML2_2.6.15) [libXML2]
xmlTextReaderValue(LIBXML2_2.4.30) [libXML2]	xmlTextReaderXmlLang(LIBXML2_2.4.30) [libXML2]	xmlTextWriterEndAttribute(LIBXML2_2.6.0) [libXML2]

xmlTextWriterEndCDATA(LIBXML2_2.6.0) [libXML2]	xmlTextWriterEndComment(LIBXML2_2.6.7) [libXML2]	xmlTextWriterEndDTD(LIBXML2_2.6.0) [libXML2]
xmlTextWriterEndDTDAttlist(LIBXML2_2.6.8) [libXML2]	xmlTextWriterEndDTDElement(LIBXML2_2.6.8) [libXML2]	xmlTextWriterEndDTDEntity(LIBXML2_2.6.8) [libXML2]
xmlTextWriterEndDocument(LIBXML2_2.6.0) [libXML2]	xmlTextWriterEndElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterEndPI(LIBXML2_2.6.0) [libXML2]
xmlTextWriterFlush(LIBXML2_2.6.0) [libXML2]	xmlTextWriterFullEndElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterSetIndent(LIBXML2_2.6.5) [libXML2]
xmlTextWriterSetIndentString(LIBXML2_2.6.5) [libXML2]	xmlTextWriterStartAttribute(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartAttributeNS(LIBXML2_2.6.0) [libXML2]
xmlTextWriterStartCDATA(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartComment(LIBXML2_2.6.7) [libXML2]	xmlTextWriterStartDTD(LIBXML2_2.6.0) [libXML2]
xmlTextWriterStartDTDAttlist(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartDTDElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartDTDEntity(LIBXML2_2.6.0) [libXML2]
xmlTextWriterStartDocument(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterStartElementNS(LIBXML2_2.6.0) [libXML2]
xmlTextWriterStartPI(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteAttribute(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteAttributeNS(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteBase64(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteBinHex(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteCDATA(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteComment(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteDTD(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteDTDAttlist(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteDTDElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteDTDEntity(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteDTDExternalEntity(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteDTDExternalEntityContents(LIBXML2_2.6.8) [libXML2]	xmlTextWriterWriteDTDInternalEntity(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteDTDNotation(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteElementNS(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatAttribute(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteFormatAttributeNS(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatCDATA(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatComment(LIBXML2_2.6.0) [libXML2]

xmlTextWriterWriteFormatDTD(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatDTDAttlist(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatDTDElement(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteFormatDTDInternalEntity(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatElementNS(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteFormatPI(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatRaw(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteFormatString(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWritePI(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteRaw(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteRawLen(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteString(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatAttribute(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatAttributeNS(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteVFormatCDATA(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatComment(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatDTD(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteVFormatDTDAttlist(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatDTDElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatDTDInternalEntity(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteVFormatElement(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatElementNS(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatPI(LIBXML2_2.6.0) [libXML2]
xmlTextWriterWriteVFormatRaw(LIBXML2_2.6.0) [libXML2]	xmlTextWriterWriteVFormatString(LIBXML2_2.6.0) [libXML2]	xmlThrDefBufferAllocScheme(LIBXML2_2.5.8) [libXML2]
xmlThrDefDefaultBufferSize(LIBXML2_2.5.8) [libXML2]	xmlThrDefDeregisterNodeDefault(LIBXML2_2.5.8) [libXML2]	xmlThrDefDoValidityCheckingDefaultValue(LIBXML2_2.5.8) [libXML2]
xmlThrDefGetWarningsDefaultValue(LIBXML2_2.5.8) [libXML2]	xmlThrDefIndentTreeOutput(LIBXML2_2.5.8) [libXML2]	xmlThrDefKeepBlanksDefaultValue(LIBXML2_2.5.8) [libXML2]
xmlThrDefLineNumbersDefaultValue(LIBXML2_2.5.8) [libXML2]	xmlThrDefLoadExtDtdDefaultValue(LIBXML2_2.5.8) [libXML2]	xmlThrDefOutputBufferCreateFilenameDefault(LIBXML2_2.6.11) [libXML2]
xmlThrDefParserDebugEntities(LIBXML2_2.5.8) [libXML2]	xmlThrDefParserInputBufferCreateFilenameDefault(LIBXML2_2.6.11) [libXML2]	xmlThrDefPedanticParserDefaultValue(LIBXML2_2.5.8) [libXML2]

xmlThrDefRegisterNodeDefault(LIBXML2_2.5.8) [libXML2]	xmlThrDefSaveNoEmptyTags(LIBXML2_2.5.8) [libXML2]	xmlThrDefSetGenericErrorFunc(LIBXML2_2.5.8) [libXML2]
xmlThrDefSetStructuredErrorFunc(LIBXML2_2.6.0) [libXML2]	xmlThrDefSubstituteEntitiesDefaultValue(LIBXML2_2.5.8) [libXML2]	xmlThrDefTreeIndentString(LIBXML2_2.5.8) [libXML2]
xmlURIEscape(LIBXML2_2.4.30) [libXML2]	xmlURIEscapeStr(LIBXML2_2.4.30) [libXML2]	xmlURIUnescapeString(LIBXML2_2.4.30) [libXML2]
xmlUTF8Charcmp(LIBXML2_2.5.9) [libXML2]	xmlUTF8Size(LIBXML2_2.5.9) [libXML2]	xmlUTF8Strlen(LIBXML2_2.4.30) [libXML2]
xmlUTF8Strloc(LIBXML2_2.4.30) [libXML2]	xmlUTF8Strndup(LIBXML2_2.4.30) [libXML2]	xmlUTF8Strpos(LIBXML2_2.4.30) [libXML2]
xmlUTF8Strsize(LIBXML2_2.4.30) [libXML2]	xmlUTF8Strsub(LIBXML2_2.4.30) [libXML2]	xmlUnlinkNode(LIBXML2_2.4.30) [libXML2]
xmlUnlockLibrary(LIBXML2_2.4.30) [libXML2]	xmlUnsetNsProp(LIBXML2_2.4.30) [libXML2]	xmlUnsetProp(LIBXML2_2.4.30) [libXML2]
xmlValidBuildContentModel(LIBXML2_2.4.30) [libXML2]	xmlValidCtxtNormalizeAttributeValue(LIBXML2_2.4.30) [libXML2]	xmlValidGetPotentialChildren(LIBXML2_2.4.30) [libXML2]
xmlValidGetValidElements(LIBXML2_2.4.30) [libXML2]	xmlValidNormalizeAttributeValue(LIBXML2_2.4.30) [libXML2]	xmlValidateAttributeDecl(LIBXML2_2.4.30) [libXML2]
xmlValidateAttributeValue(LIBXML2_2.4.30) [libXML2]	xmlValidateDocument(LIBXML2_2.4.30) [libXML2]	xmlValidateDocumentFinal(LIBXML2_2.4.30) [libXML2]
xmlValidateDtd(LIBXML2_2.4.30) [libXML2]	xmlValidateDtdFinal(LIBXML2_2.4.30) [libXML2]	xmlValidateElement(LIBXML2_2.4.30) [libXML2]
xmlValidateElementDecl(LIBXML2_2.4.30) [libXML2]	xmlValidateNCName(LIBXML2_2.5.4) [libXML2]	xmlValidateNMToken(LIBXML2_2.5.4) [libXML2]
xmlValidateName(LIBXML2_2.5.4) [libXML2]	xmlValidateNameValue(LIBXML2_2.4.30) [libXML2]	xmlValidateNamesValue(LIBXML2_2.4.30) [libXML2]
xmlValidateNmtokenValue(LIBXML2_2.4.30) [libXML2]	xmlValidateNmtokensValue(LIBXML2_2.4.30) [libXML2]	xmlValidateNotationDecl(LIBXML2_2.4.30) [libXML2]
xmlValidateNotationUse(LIBXML2_2.4.30) [libXML2]	xmlValidateOneAttribute(LIBXML2_2.4.30) [libXML2]	xmlValidateOneElement(LIBXML2_2.4.30) [libXML2]
xmlValidateOneNamespace(LIBXML2_2.4.30) [libXML2]	xmlValidatePopElement(LIBXML2_2.5.0) [libXML2]	xmlValidatePushCData(LIBXML2_2.5.0) [libXML2]

xmlValidatePushElement(LIBXML2_2.5.0) [libXML2]	xmlValidateQName(LIBXML2_2.5.4) [libXML2]	xmlValidateRoot(LIBXML2_2.4.30) [libXML2]
xmlXIncludeFreeContext(LIBXML2_2.6.2) [libXML2]	xmlXIncludeNewContext(LIBXML2_2.6.2) [libXML2]	xmlXIncludeProcess(LIBXML2_2.4.30) [libXML2]
xmlXIncludeProcessFlags(LIBXML2_2.6.3) [libXML2]	xmlXIncludeProcessNode(LIBXML2_2.6.2) [libXML2]	xmlXIncludeProcessTree(LIBXML2_2.5.9) [libXML2]
xmlXIncludeProcessTreeFlags(LIBXML2_2.6.3) [libXML2]	xmlXIncludeSetFlags(LIBXML2_2.6.3) [libXML2]	xmlXPathAddValues(LIBXML2_2.4.30) [libXML2]
xmlXPathBooleanFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathCastBooleanToNumber(LIBXML2_2.4.30) [libXML2]	xmlXPathCastBooleanToString(LIBXML2_2.4.30) [libXML2]
xmlXPathCastNodeSetToBoolean(LIBXML2_2.4.30) [libXML2]	xmlXPathCastNodeSetToNumber(LIBXML2_2.4.30) [libXML2]	xmlXPathCastNodeSetToString(LIBXML2_2.4.30) [libXML2]
xmlXPathCastNodeToNumber(LIBXML2_2.4.30) [libXML2]	xmlXPathCastNodeToString(LIBXML2_2.4.30) [libXML2]	xmlXPathCastNumberToBoolean(LIBXML2_2.4.30) [libXML2]
xmlXPathCastNumberToString(LIBXML2_2.4.30) [libXML2]	xmlXPathCastStringToBoolean(LIBXML2_2.4.30) [libXML2]	xmlXPathCastStringToNumber(LIBXML2_2.4.30) [libXML2]
xmlXPathCastToBoolean(LIBXML2_2.4.30) [libXML2]	xmlXPathCastToNumber(LIBXML2_2.4.30) [libXML2]	xmlXPathCastToString(LIBXML2_2.4.30) [libXML2]
xmlXPathCeilingFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathCmpNodes(LIBXML2_2.4.30) [libXML2]	xmlXPathCompareValues(LIBXML2_2.4.30) [libXML2]
xmlXPathCompile(LIBXML2_2.4.30) [libXML2]	xmlXPathCompiledEval(LIBXML2_2.4.30) [libXML2]	xmlXPathConcatFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathContainsFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathConvertBoolean(LIBXML2_2.4.30) [libXML2]	xmlXPathConvertNumber(LIBXML2_2.4.30) [libXML2]
xmlXPathConvertString(LIBXML2_2.4.30) [libXML2]	xmlXPathCountFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathCtxtCompile(LIBXML2_2.6.5) [libXML2]
xmlXPathDebugDumpCompExpr(LIBXML2_2.4.30) [libXML2]	xmlXPathDebugDumpObject(LIBXML2_2.4.30) [libXML2]	xmlXPathDifference(LIBXML2_2.4.30) [libXML2]
xmlXPathDistinct(LIBXML2_2.4.30) [libXML2]	xmlXPathDistinctSorted(LIBXML2_2.4.30) [libXML2]	xmlXPathDivValues(LIBXML2_2.4.30) [libXML2]

xmlXPathEqualValues(LIBXML2_2.4.30) [libXML2]	xmlXPathErr(LIBXML2_2.6.0) [libXML2]	xmlXPathEval(LIBXML2_2.4.30) [libXML2]
xmlXPathEvalExpr(LIBXML2_2.4.30) [libXML2]	xmlXPathEvalExpression(LIBXML2_2.4.30) [libXML2]	xmlXPathEvalPredicate(LIBXML2_2.4.30) [libXML2]
xmlXPathEvaluatePredicateResult(LIBXML2_2.4.30) [libXML2]	xmlXPathFalseFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathFloorFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathFreeCompExpr(LIBXML2_2.4.30) [libXML2]	xmlXPathFreeContext(LIBXML2_2.4.30) [libXML2]	xmlXPathFreeNodeSet(LIBXML2_2.4.30) [libXML2]
xmlXPathFreeNodeSetList(LIBXML2_2.4.30) [libXML2]	xmlXPathFreeObject(LIBXML2_2.4.30) [libXML2]	xmlXPathFreeParserContext(LIBXML2_2.4.30) [libXML2]
xmlXPathFunctionLookup(LIBXML2_2.4.30) [libXML2]	xmlXPathFunctionLookupNS(LIBXML2_2.4.30) [libXML2]	xmlXPathHasSameNodes(LIBXML2_2.4.30) [libXML2]
xmlXPathIdFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathInit(LIBXML2_2.4.30) [libXML2]	xmlXPathIntersection(LIBXML2_2.4.30) [libXML2]
xmlXPathIsInf(LIBXML2_2.4.30) [libXML2]	xmlXPathIsNaN(LIBXML2_2.4.30) [libXML2]	xmlXPathIsNodeType(LIBXML2_2.4.30) [libXML2]
xmlXPathLangFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathLastFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathLeading(LIBXML2_2.4.30) [libXML2]
xmlXPathLeadingSorted(LIBXML2_2.4.30) [libXML2]	xmlXPathLocalNameFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathModValues(LIBXML2_2.4.30) [libXML2]
xmlXPathMultValues(LIBXML2_2.4.30) [libXML2]	xmlXPathNamespaceURIFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathNewBoolean(LIBXML2_2.4.30) [libXML2]
xmlXPathNewCString(LIBXML2_2.4.30) [libXML2]	xmlXPathNewContext(LIBXML2_2.4.30) [libXML2]	xmlXPathNewFloat(LIBXML2_2.4.30) [libXML2]
xmlXPathNewNodeSet(LIBXML2_2.4.30) [libXML2]	xmlXPathNewNodeSetList(LIBXML2_2.4.30) [libXML2]	xmlXPathNewParserContext(LIBXML2_2.4.30) [libXML2]
xmlXPathNewString(LIBXML2_2.4.30) [libXML2]	xmlXPathNewValueTree(LIBXML2_2.4.30) [libXML2]	xmlXPathNextAncestor(LIBXML2_2.4.30) [libXML2]
xmlXPathNextAncestorOrSelf(LIBXML2_2.4.30) [libXML2]	xmlXPathNextAttribute(LIBXML2_2.4.30) [libXML2]	xmlXPathNextChild(LIBXML2_2.4.30) [libXML2]

xmlXPathNextDescendant(LIBXML2_2.4.30) [libXML2]	xmlXPathNextDescendantOrSelf(LIBXML2_2.4.30) [libXML2]	xmlXPathNextFollowing(LIBXML2_2.4.30) [libXML2]
xmlXPathNextFollowingSibling(LIBXML2_2.4.30) [libXML2]	xmlXPathNextNamespace(LIBXML2_2.4.30) [libXML2]	xmlXPathNextParent(LIBXML2_2.4.30) [libXML2]
xmlXPathNextPreceding(LIBXML2_2.4.30) [libXML2]	xmlXPathNextPrecedingSibling(LIBXML2_2.4.30) [libXML2]	xmlXPathNextSelf(LIBXML2_2.4.30) [libXML2]
xmlXPathNodeLeading(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeLeadingSorted(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetAdd(LIBXML2_2.4.30) [libXML2]
xmlXPathNodeSetAddNs(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetAddUnique(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetContains(LIBXML2_2.4.30) [libXML2]
xmlXPathNodeSetCreate(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetDel(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetFreeNs(LIBXML2_2.4.30) [libXML2]
xmlXPathNodeSetMerge(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetRemove(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeSetSort(LIBXML2_2.4.30) [libXML2]
xmlXPathNodeTrailing(LIBXML2_2.4.30) [libXML2]	xmlXPathNodeTrailingSorted(LIBXML2_2.4.30) [libXML2]	xmlXPathNormalizeFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathNotEqualValues(LIBXML2_2.4.30) [libXML2]	xmlXPathNotFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathNsLookup(LIBXML2_2.4.30) [libXML2]
xmlXPathNumberFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathObjectCopy(LIBXML2_2.4.30) [libXML2]	xmlXPathOrderDocElems(LIBXML2_2.5.6) [libXML2]
xmlXPathParseNCName(LIBXML2_2.4.30) [libXML2]	xmlXPathParseName(LIBXML2_2.4.30) [libXML2]	xmlXPathPopBoolean(LIBXML2_2.4.30) [libXML2]
xmlXPathPopExternal(LIBXML2_2.4.30) [libXML2]	xmlXPathPopNodeSet(LIBXML2_2.4.30) [libXML2]	xmlXPathPopNumber(LIBXML2_2.4.30) [libXML2]
xmlXPathPopString(LIBXML2_2.4.30) [libXML2]	xmlXPathPositionFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisterAllFunctions(LIBXML2_2.4.30) [libXML2]
xmlXPathRegisterFunc(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisterFuncLookup(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisterFuncNS(LIBXML2_2.4.30) [libXML2]
xmlXPathRegisterNs(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisterVariable(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisterVariableLookup(LIBXML2_2.4.30) [libXML2]

xmlXPathRegisterVariableNS(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisteredFuncsCleanup(LIBXML2_2.4.30) [libXML2]	xmlXPathRegisteredNsCleanup(LIBXML2_2.4.30) [libXML2]
xmlXPathRegisteredVariablesCleanup(LIBXML2_2.4.30) [libXML2]	xmlXPathRoot(LIBXML2_2.4.30) [libXML2]	xmlXPathRoundFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathStartsWithFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathStringEvalNumber(LIBXML2_2.4.30) [libXML2]	xmlXPathStringFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathStringLengthFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathSubValues(LIBXML2_2.4.30) [libXML2]	xmlXPathSubstringAfterFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathSubstringBeforeFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathSubstringFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathSumFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathTrailing(LIBXML2_2.4.30) [libXML2]	xmlXPathTrailingSorted(LIBXML2_2.4.30) [libXML2]	xmlXPathTranslateFunction(LIBXML2_2.4.30) [libXML2]
xmlXPathTrueFunction(LIBXML2_2.4.30) [libXML2]	xmlXPathValueFlipSign(LIBXML2_2.4.30) [libXML2]	xmlXPathVariableLookup(LIBXML2_2.4.30) [libXML2]
xmlXPathVariableLookupNS(LIBXML2_2.4.30) [libXML2]	xmlXPathWrapCString(LIBXML2_2.4.30) [libXML2]	xmlXPathWrapExternal(LIBXML2_2.4.30) [libXML2]
xmlXPathWrapNodeSet(LIBXML2_2.4.30) [libXML2]	xmlXPathWrapString(LIBXML2_2.4.30) [libXML2]	xmlXPatherror(LIBXML2_2.4.30) [libXML2]
xmlXPtrBuildNodeList(LIBXML2_2.4.30) [libXML2]	xmlXPtrEval(LIBXML2_2.4.30) [libXML2]	xmlXPtrEvalRangePredicate(LIBXML2_2.4.30) [libXML2]
xmlXPtrFreeLocationSet(LIBXML2_2.4.30) [libXML2]	xmlXPtrLocationSetAdd(LIBXML2_2.4.30) [libXML2]	xmlXPtrLocationSetCreate(LIBXML2_2.4.30) [libXML2]
xmlXPtrLocationSetDel(LIBXML2_2.4.30) [libXML2]	xmlXPtrLocationSetMerge(LIBXML2_2.4.30) [libXML2]	xmlXPtrLocationSetRemove(LIBXML2_2.4.30) [libXML2]
xmlXPtrNewCollapsedRange(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewContext(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewLocationSetNodeSet(LIBXML2_2.4.30) [libXML2]
xmlXPtrNewLocationSetNodes(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewRange(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewRangeNodeObject(LIBXML2_2.4.30) [libXML2]
xmlXPtrNewRangeNodePoint(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewRangeNodes(LIBXML2_2.4.30) [libXML2]	xmlXPtrNewRangePointNode(LIBXML2_2.4.30) [libXML2]

xmlXPtrNewRangePoints(LIBXML2_2.4.30) [libXML2]	xmlXPtrRangeToFunction(LIBXML2_2.4.30) [libXML2]	xmlXPtrWrapLocationSet(LIBXML2_2.4.30) [libXML2]

An LSB conforming implementation shall provide the generic deprecated functions for The XML C parser and toolkit for XML processing specified in Table 8-3, with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 8-3 libxml2 - The XML C parser and toolkit for XML processing Deprecated Function Interfaces

xmlParserInputRead [libXML2]	xmlParserInputRead(LIBXML2_2.4.30) [libXML2]	

An LSB conforming implementation shall provide the generic data interfaces for The XML C parser and toolkit for XML processing specified in Table 8-4, with the full mandatory functionality as described in the referenced underlying specification.

Table 8-4 libxml2 - The XML C parser and toolkit for XML processing Data Interfaces

emptyExp(LIBXML2_2.6.21) [libXML2]	forbiddenExp(LIBXML2_2.6.21) [libXML2]	xmlFree(LIBXML2_2.4.30) [libXML2]
xmlMalloc(LIBXML2_2.4.30) [libXML2]	xmlMallocAtomic(LIBXML2_2.5.7) [libXML2]	xmlMemStrdup(LIBXML2_2.4.30) [libXML2]
xmlParserMaxDepth(LIBXML2_2.6.0) [libXML2]	xmlRealloc(LIBXML2_2.4.30) [libXML2]	xmlStringComment(LIBXML2_2.4.30) [libXML2]
xmlStringText(LIBXML2_2.4.30) [libXML2]	xmlStringTextNoenc(LIBXML2_2.4.30) [libXML2]	xmlXPathNAN(LIBXML2_2.4.30) [libXML2]
xmlXPathNINF(LIBXML2_2.4.30) [libXML2]	xmlXPathPINF(LIBXML2_2.4.30) [libXML2]	

8.2 Data Definitions for libxml2

This section defines global identifiers and their values that are associated with interfaces contained in libxml2. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

8.2.1 libxml2/libxml/HTMLparser.h

```
#define htmlElementAllowedHereDesc(parent,elt)   \
      htmlElementAllowedHere((parent), (elt)->name)
#define htmlRequiredAttrs(elt)  (elt)->attrs_req
#define htmlDefaultSubelement(elt)     elt->defaultsubelt

typedef enum {
    HTML_NA = 0,
    HTML_INVALID = 1,
    HTML_DEPRECATED = 2,
    HTML_VALID = 4,
    HTML_REQUIRED = 12
} htmlStatus;
typedef struct _htmlElemDesc {
    const char *name;
    char startTag;
    char endTag;
    char saveEndTag;
    char empty;
    char depr;
    char dtd;
    char isinline;
    const char *desc;
    const char **subelts;
    const char *defaultsubelt;
    const char **attrs_opt;
    const char **attrs_depr;
    const char **attrs_req;
} htmlElemDesc;
typedef xmlDocPtr htmlDocPtr;
typedef xmlSAXHandlerPtr htmlSAXHandlerPtr;
typedef xmlParserCtxtPtr htmlParserCtxtPtr;
typedef struct _htmlEntityDesc {
    unsigned int value;
    const char *name;
    const char *desc;
} htmlEntityDesc;
typedef xmlNodePtr htmlNodePtr;
typedef enum {
    HTML_PARSE_RECOVER = 1 << 0,
    HTML_PARSE_NOERROR = 1 << 5,
    HTML_PARSE_NOWARNING = 1 << 6,
    HTML_PARSE_PEDANTIC = 1 << 7,
    HTML_PARSE_NOBLANKS = 1 << 8,
    HTML_PARSE_NONET = 1 << 11,
    HTML_PARSE_COMPACT = 1 << 16
} htmlParserOption;
typedef xmlParserInputPtr htmlParserInputPtr;
typedef htmlElemDesc *htmlElemDescPtr;
typedef htmlEntityDesc *htmlEntityDescPtr;
typedef xmlParserInput htmlParserInput;
typedef xmlSAXHandler htmlSAXHandler;
extern int UTF8ToHtml(unsigned char *out, int *outlen,
                      const unsigned char *in, int *inlen);
extern  htmlStatus  htmlAttrAllowed(const  htmlElemDesc  *,  const
xmlChar *,
                                int);
extern int htmlAutoCloseTag(htmlDocPtr doc, const xmlChar * name,
```



```
                        htmlNodePtr elem);
extern  htmlParserCtxtPtr  htmlCreateMemoryParserCtxt(const  char
*buffer,
                                                int size);
extern                                               htmlParserCtxtPtr
htmlCreatePushParserCtxt(htmlSAXHandlerPtr sax,
                                               void *user_data,
                                               const char *chunk,
                                               int size,
                                               const char *filename,
                                               xmlCharEncoding enc);
extern htmlDocPtr htmlCtxtReadDoc(htmlParserCtxtPtr ctxt,
                             const xmlChar * cur, const char *URL,
                             const char *encoding, int options);
extern htmlDocPtr htmlCtxtReadFd(htmlParserCtxtPtr ctxt, int fd,
                            const char *URL, const char *encoding,
                            int options);
extern htmlDocPtr htmlCtxtReadFile(htmlParserCtxtPtr ctxt,
                              const char *filename,
                              const char *encoding, int options);
extern htmlDocPtr htmlCtxtReadIO(htmlParserCtxtPtr ctxt,
                            xmlInputReadCallback ioread,
                            xmlInputCloseCallback ioclose,
                            void *ioctx, const char *URL,
                            const char *encoding, int options);
extern htmlDocPtr htmlCtxtReadMemory(htmlParserCtxtPtr ctxt,
                                const char *buffer, int size,
                                const   char   *URL,   const   char
*encoding,
                                int options);
extern void htmlCtxtReset(htmlParserCtxtPtr ctxt);
extern int htmlCtxtUseOptions(htmlParserCtxtPtr ctxt, int options);
extern  int  htmlElementAllowedHere(const  htmlElemDesc  *,  const
xmlChar *);
extern htmlStatus htmlElementStatusHere(const htmlElemDesc *,
                                   const htmlElemDesc *);
extern int htmlEncodeEntities(unsigned char *out, int *outlen,
                         const unsigned char *in, int *inlen,
                         int quoteChar);
extern const htmlEntityDesc *htmlEntityLookup(const xmlChar * name);
extern  const  htmlEntityDesc  *htmlEntityValueLookup(unsigned  int
value);
extern void htmlFreeParserCtxt(htmlParserCtxtPtr ctxt);
extern int htmlHandleOmittedElem(int val);
extern int htmlIsAutoClosed(htmlDocPtr doc, htmlNodePtr elem);
extern int htmlIsScriptAttribute(const xmlChar * name);
extern int htmlParseCharRef(htmlParserCtxtPtr ctxt);
extern int htmlParseChunk(htmlParserCtxtPtr ctxt, const char *chunk,
                     int size, int terminate);
extern htmlDocPtr htmlParseDoc(xmlChar * cur, const char *encoding);
extern int htmlParseDocument(htmlParserCtxtPtr ctxt);
extern void htmlParseElement(htmlParserCtxtPtr ctxt);
extern  const  htmlEntityDesc  *htmlParseEntityRef(htmlParserCtxtPtr
ctxt,
                                              const xmlChar * *str);
extern htmlDocPtr htmlParseFile(const char *filename,
                           const char *encoding);
extern htmlDocPtr htmlReadDoc(const xmlChar * cur, const char *URL,
                         const char *encoding, int options);
extern htmlDocPtr htmlReadFd(int fd, const char *URL, const char
*encoding,
                        int options);
extern  htmlDocPtr  htmlReadFile(const  char  *URL,  const  char
*encoding,
                          int options);
extern htmlDocPtr htmlReadIO(xmlInputReadCallback ioread,
```

```
                          xmlInputCloseCallback ioclose, void *ioctx,
                          const char *URL, const char *encoding,
                          int options);
extern htmlDocPtr htmlReadMemory(const char *buffer, int size,
                            const char *URL, const char *encoding,
                            int options);
extern  htmlDocPtr  htmlSAXParseDoc(xmlChar  *  cur,  const  char
*encoding,
                             htmlSAXHandlerPtr sax, void *userData);
extern htmlDocPtr htmlSAXParseFile(const char *filename,
                              const char *encoding,
                              htmlSAXHandlerPtr      sax,      void
*userData);
extern const htmlElemDesc *htmlTagLookup(const xmlChar * tag);
```

8.2.2 libxml2/libxml/HTMLtree.h

```
#define HTML_PRESERVE_NODE     XML_CDATA_SECTION_NODE
#define HTML_COMMENT_NODE      XML_COMMENT_NODE
#define HTML_ENTITY_REF_NODE    XML_ENTITY_REF_NODE
#define HTML_PI_NODE    XML_PI_NODE
#define HTML_TEXT_NODE  XML_TEXT_NODE

extern void htmlDocContentDumpFormatOutput(xmlOutputBufferPtr buf,
                                    xmlDocPtr cur,
                                    const char *encoding,
                                    int format);
extern   void   htmlDocContentDumpOutput(xmlOutputBufferPtr   buf,
xmlDocPtr cur,
                                const char *encoding);
extern int htmlDocDump(FILE * f, xmlDocPtr cur);
extern void htmlDocDumpMemory(xmlDocPtr cur, xmlChar * *mem, int
*size);
extern const xmlChar *htmlGetMetaEncoding(htmlDocPtr doc);
extern int htmlIsBooleanAttr(const xmlChar * name);
extern htmlDocPtr htmlNewDoc(const xmlChar * URI,
                          const xmlChar * ExternalID);
extern htmlDocPtr htmlNewDocNoDtD(const xmlChar * URI,
                             const xmlChar * ExternalID);
extern int htmlNodeDump(xmlBufferPtr buf, xmlDocPtr doc, xmlNodePtr
cur);
extern void htmlNodeDumpFile(FILE * out, xmlDocPtr doc, xmlNodePtr
cur);
extern int htmlNodeDumpFileFormat(FILE * out, xmlDocPtr doc,
                             xmlNodePtr cur, const char *encoding,
                             int format);
extern   void   htmlNodeDumpFormatOutput(xmlOutputBufferPtr   buf,
xmlDocPtr doc,
                                xmlNodePtr    cur,    const    char
*encoding,
                                int format);
extern  void  htmlNodeDumpOutput(xmlOutputBufferPtr  buf,  xmlDocPtr
doc,
                            xmlNodePtr cur, const char *encoding);
extern int htmlSaveFile(const char *filename, xmlDocPtr cur);
extern int htmlSaveFileEnc(const char *filename, xmlDocPtr cur,
                        const char *encoding);
extern int htmlSaveFileFormat(const char *filename, xmlDocPtr cur,
                           const char *encoding, int format);
extern  int  htmlSetMetaEncoding(htmlDocPtr  doc,  const  xmlChar  *
encoding);
```


8.2.3 libxml2/libxml/SAX2.h

```
typedef void (*internalSubsetSAXFunc) (void *, const xmlChar *,
                                       const xmlChar *, const xmlChar *);
typedef int (*isStandaloneSAXFunc) (void *);
typedef int (*hasInternalSubsetSAXFunc) (void *);
typedef int (*hasExternalSubsetSAXFunc) (void *);
typedef  xmlParserInputPtr(*resolveEntitySAXFunc)  (void  *,  const
xmlChar *,
                                                    const xmlChar *);
typedef xmlEntityPtr(*getEntitySAXFunc) (void *, const xmlChar *);
typedef void (*entityDeclSAXFunc) (void *, const xmlChar *, int,
                                   const xmlChar *, const xmlChar *,
                                   xmlChar *);
typedef void (*notationDeclSAXFunc) (void *, const xmlChar *,
                                     const xmlChar *, const xmlChar *);
typedef struct _xmlEnumeration {
    struct _xmlEnumeration *next;
    const xmlChar *name;
} xmlEnumeration;
typedef xmlEnumeration *xmlEnumerationPtr;
typedef void (*attributeDeclSAXFunc) (void *, const xmlChar *,
                                      const xmlChar *, int, int,
                                      const           xmlChar          *,
xmlEnumerationPtr);
typedef enum {
    XML_ELEMENT_CONTENT_PCDATA = 1,
    XML_ELEMENT_CONTENT_ELEMENT = 2,
    XML_ELEMENT_CONTENT_SEQ = 3,
    XML_ELEMENT_CONTENT_OR = 4
} xmlElementContentType;
typedef enum {
    XML_ELEMENT_CONTENT_ONCE = 1,
    XML_ELEMENT_CONTENT_OPT = 2,
    XML_ELEMENT_CONTENT_MULT = 3,
    XML_ELEMENT_CONTENT_PLUS = 4
} xmlElementContentOccur;
typedef struct _xmlElementContent {
    xmlElementContentType type;
    xmlElementContentOccur ocur;
    const xmlChar *name;
    struct _xmlElementContent *c1;
    struct _xmlElementContent *c2;
    struct _xmlElementContent *parent;
    const xmlChar *prefix;
} xmlElementContent;
typedef xmlElementContent *xmlElementContentPtr;
typedef void (*elementDeclSAXFunc) (void *, const xmlChar *, int,
                                    xmlElementContentPtr);
typedef void (*unparsedEntityDeclSAXFunc) (void *, const xmlChar *,
                                           const xmlChar *,
                                           const xmlChar *,
                                           const xmlChar *);
typedef struct _xmlSAXLocator {
    const xmlChar *(*getPublicId) (void *);
    const xmlChar *(*getSystemId) (void *);
    int (*getLineNumber) (void *);
    int (*getColumnNumber) (void *);
} xmlSAXLocator;
typedef xmlSAXLocator *xmlSAXLocatorPtr;
typedef     void     (*setDocumentLocatorSAXFunc)     (void     *,
xmlSAXLocatorPtr);
typedef void (*startDocumentSAXFunc) (void *);
typedef void (*endDocumentSAXFunc) (void *);
typedef void (*startElementSAXFunc) (void *, const xmlChar *,
```

```
                                    const xmlChar * *);
typedef void (*endElementSAXFunc) (void *, const xmlChar *);
typedef void (*referenceSAXFunc) (void *, const xmlChar *);
typedef void (*charactersSAXFunc) (void *, const xmlChar *, int);
typedef void (*ignorableWhitespaceSAXFunc) (void *, const xmlChar
*, int);
typedef void (*processingInstructionSAXFunc) (void *, const xmlChar
*,
                                              const xmlChar *);
typedef void (*commentSAXFunc) (void *, const xmlChar *);
typedef void (*warningSAXFunc) (void *, const char *, ...);
typedef void (*errorSAXFunc) (void *, const char *, ...);
typedef void (*fatalErrorSAXFunc) (void *, const char *, ...);
typedef  xmlEntityPtr(*getParameterEntitySAXFunc)  (void  *,  const
xmlChar *);
typedef void (*cdataBlockSAXFunc) (void *, const xmlChar *, int);
typedef void (*externalSubsetSAXFunc) (void *, const xmlChar *,
                                       const xmlChar *, const xmlChar *);
typedef void (*startElementNsSAX2Func) (void *, const xmlChar *,
                                        const xmlChar *, const xmlChar *,
                                        int, const xmlChar * *, int, int,
                                        const xmlChar * *);
typedef void (*endElementNsSAX2Func) (void *, const xmlChar *,
                                      const xmlChar *, const xmlChar *);
typedef struct _xmlSAXHandler {
    internalSubsetSAXFunc internalSubset;
    isStandaloneSAXFunc isStandalone;
    hasInternalSubsetSAXFunc hasInternalSubset;
    hasExternalSubsetSAXFunc hasExternalSubset;
    resolveEntitySAXFunc resolveEntity;
    getEntitySAXFunc getEntity;
    entityDeclSAXFunc entityDecl;
    notationDeclSAXFunc notationDecl;
    attributeDeclSAXFunc attributeDecl;
    elementDeclSAXFunc elementDecl;
    unparsedEntityDeclSAXFunc unparsedEntityDecl;
    setDocumentLocatorSAXFunc setDocumentLocator;
    startDocumentSAXFunc startDocument;
    endDocumentSAXFunc endDocument;
    startElementSAXFunc startElement;
    endElementSAXFunc endElement;
    referenceSAXFunc reference;
    charactersSAXFunc characters;
    ignorableWhitespaceSAXFunc ignorableWhitespace;
    processingInstructionSAXFunc processingInstruction;
    commentSAXFunc comment;
    warningSAXFunc warning;
    errorSAXFunc error;
    fatalErrorSAXFunc fatalError;
    getParameterEntitySAXFunc getParameterEntity;
    cdataBlockSAXFunc cdataBlock;
    externalSubsetSAXFunc externalSubset;
    unsigned int initialized;
    void *_private;
    startElementNsSAX2Func startElementNs;
    endElementNsSAX2Func endElementNs;
    xmlStructuredErrorFunc serror;
} xmlSAXHandler;
extern void docbDefaultSAXHandlerInit(void);
extern void htmlDefaultSAXHandlerInit(void);
extern void xmlDefaultSAXHandlerInit(void);
extern void xmlSAX2AttributeDecl(void *ctx, const xmlChar * elem,
                                 const xmlChar * fullname, int type,
                                 int def, const xmlChar * defaultValue,
                                 xmlEnumerationPtr tree);
```



```
extern void xmlSAX2CDataBlock(void *ctx, const xmlChar * value, int
len);
extern void xmlSAX2Characters(void *ctx, const xmlChar * ch, int
len);
extern void xmlSAX2Comment(void *ctx, const xmlChar * value);
extern void xmlSAX2ElementDecl(void *ctx, const xmlChar * name, int
type,
                               xmlElementContentPtr content);
extern void xmlSAX2EndDocument(void *ctx);
extern void xmlSAX2EndElement(void *ctx, const xmlChar * name);
extern  void  xmlSAX2EndElementNs(void  *ctx,  const  xmlChar  *
localname,
                                  const xmlChar * prefix,
                                  const xmlChar * URI);
extern void xmlSAX2EntityDecl(void *ctx, const xmlChar * name, int
type,
                              const xmlChar * publicId,
                              const  xmlChar  *  systemId,  xmlChar  *
content);
extern void xmlSAX2ExternalSubset(void *ctx, const xmlChar * name,
                                  const xmlChar * ExternalID,
                                  const xmlChar * SystemID);
extern int xmlSAX2GetColumnNumber(void *ctx);
extern  xmlEntityPtr  xmlSAX2GetEntity(void  *ctx,  const  xmlChar  *
name);
extern int xmlSAX2GetLineNumber(void *ctx);
extern xmlEntityPtr xmlSAX2GetParameterEntity(void *ctx,
                                              const xmlChar * name);
extern const xmlChar *xmlSAX2GetPublicId(void *ctx);
extern const xmlChar *xmlSAX2GetSystemId(void *ctx);
extern int xmlSAX2HasExternalSubset(void *ctx);
extern int xmlSAX2HasInternalSubset(void *ctx);
extern void xmlSAX2IgnorableWhitespace(void *ctx, const xmlChar *
ch,
                                       int len);
extern void xmlSAX2InitDefaultSAXHandler(xmlSAXHandler * hdlr,
                                         int warning);
extern void xmlSAX2InitDocbDefaultSAXHandler(xmlSAXHandler * hdlr);
extern void xmlSAX2InitHtmlDefaultSAXHandler(xmlSAXHandler * hdlr);
extern void xmlSAX2InternalSubset(void *ctx, const xmlChar * name,
                                  const xmlChar * ExternalID,
                                  const xmlChar * SystemID);
extern int xmlSAX2IsStandalone(void *ctx);
extern void xmlSAX2NotationDecl(void *ctx, const xmlChar * name,
                                const xmlChar * publicId,
                                const xmlChar * systemId);
extern void xmlSAX2ProcessingInstruction(void *ctx, const xmlChar
* target,
                                         const xmlChar * data);
extern void xmlSAX2Reference(void *ctx, const xmlChar * name);
extern xmlParserInputPtr xmlSAX2ResolveEntity(void *ctx,
                                              const xmlChar * publicId,
                                              const xmlChar * systemId);
extern void xmlSAX2SetDocumentLocator(void *ctx, xmlSAXLocatorPtr
loc);
extern void xmlSAX2StartDocument(void *ctx);
extern void xmlSAX2StartElement(void *ctx, const xmlChar * fullname,
                                const xmlChar * *atts);
extern  void  xmlSAX2StartElementNs(void  *ctx,  const  xmlChar  *
localname,
                                    const xmlChar * prefix,
                                    const xmlChar * URI, int nb_namespaces,
                                    const xmlChar * *namespaces,
                                    int nb_attributes, int nb_defaulted,
                                    const xmlChar * *attributes);
```



```
extern void xmlSAX2UnparsedEntityDecl(void *ctx, const xmlChar *
name,
                                const xmlChar * publicId,
                                const xmlChar * systemId,
                                const xmlChar * notationName);
extern int xmlSAXDefaultVersion(int version);
extern int xmlSAXVersion(xmlSAXHandler * hdlr, int version);
```

8.2.4 libxml2/libxml/c14n.h

```
typedef int (*xmlC14NIsVisibleCallback) (void *, xmlNodePtr,
xmlNodePtr);
extern int xmlC14NDocDumpMemory(xmlDocPtr, xmlNodeSetPtr, int,
xmlChar * *,
                            int, xmlChar * *);
extern int xmlC14NDocSave(xmlDocPtr, xmlNodeSetPtr, int, xmlChar *
*, int,
                        const char *, int);
extern int xmlC14NDocSaveTo(xmlDocPtr, xmlNodeSetPtr, int, xmlChar
* *,
                         int, xmlOutputBufferPtr);
extern int xmlC14NExecute(xmlDocPtr, xmlC14NIsVisibleCallback,
void *, int,
                        xmlChar * *, int, xmlOutputBufferPtr);
```

8.2.5 libxml2/libxml/catalog.h

```
#define XML_CATALOGS_NAMESPACE   \
       (const                           xmlChar                      *)
"urn:oasis:names:tc:entity:xmlns:xml:catalog"
#define XML_CATALOG_PI  (const xmlChar *) "oasis-xml-catalog"

typedef enum {
    XML_CATA_ALLOW_NONE = 0,
    XML_CATA_ALLOW_GLOBAL = 1,
    XML_CATA_ALLOW_DOCUMENT = 2,
    XML_CATA_ALLOW_ALL = 3
} xmlCatalogAllow;
typedef struct _xmlCatalog xmlCatalog;
typedef xmlCatalog *xmlCatalogPtr;
typedef enum {
    XML_CATA_PREFER_NONE = 0,
    XML_CATA_PREFER_PUBLIC = 1,
    XML_CATA_PREFER_SYSTEM = 2
} xmlCatalogPrefer;
extern int xmlACatalogAdd(xmlCatalogPtr catal, const xmlChar * type,
                        const xmlChar * orig, const xmlChar *
replace);
extern void xmlACatalogDump(xmlCatalogPtr catal, FILE * out);
extern int xmlACatalogRemove(xmlCatalogPtr catal, const xmlChar *
value);
extern xmlChar *xmlACatalogResolve(xmlCatalogPtr catal,
                                const xmlChar * pubID,
                                const xmlChar * sysID);
extern xmlChar *xmlACatalogResolvePublic(xmlCatalogPtr catal,
                                    const xmlChar * pubID);
extern xmlChar *xmlACatalogResolveSystem(xmlCatalogPtr catal,
                                    const xmlChar * sysID);
extern xmlChar *xmlACatalogResolveURI(xmlCatalogPtr catal,
                                   const xmlChar * URI);
extern int xmlCatalogAdd(const xmlChar * type, const xmlChar * orig,
                     const xmlChar * replace);
```



```
extern void *xmlCatalogAddLocal(void *catalogs, const xmlChar *
URL);
extern void xmlCatalogCleanup(void);
extern int xmlCatalogConvert(void);
extern void xmlCatalogDump(FILE * out);
extern void xmlCatalogFreeLocal(void *catalogs);
extern xmlCatalogAllow xmlCatalogGetDefaults(void);
extern int xmlCatalogIsEmpty(xmlCatalogPtr catal);
extern xmlChar *xmlCatalogLocalResolve(void *catalogs,
                                       const xmlChar * pubID,
                                       const xmlChar * sysID);
extern xmlChar *xmlCatalogLocalResolveURI(void *catalogs,
                                          const xmlChar * URI);
extern int xmlCatalogRemove(const xmlChar * value);
extern xmlChar *xmlCatalogResolve(const xmlChar * pubID,
                                  const xmlChar * sysID);
extern xmlChar *xmlCatalogResolvePublic(const xmlChar * pubID);
extern xmlChar *xmlCatalogResolveSystem(const xmlChar * sysID);
extern xmlChar *xmlCatalogResolveURI(const xmlChar * URI);
extern int xmlCatalogSetDebug(int level);
extern                                              xmlCatalogPrefer
xmlCatalogSetDefaultPrefer(xmlCatalogPrefer
                                                 prefer);
extern void xmlCatalogSetDefaults(xmlCatalogAllow allow);
extern int xmlConvertSGMLCatalog(xmlCatalogPtr catal);
extern void xmlFreeCatalog(xmlCatalogPtr catal);
extern void xmlInitializeCatalog(void);
extern xmlCatalogPtr xmlLoadACatalog(const char *filename);
extern int xmlLoadCatalog(const char *filename);
extern void xmlLoadCatalogs(const char *paths);
extern xmlCatalogPtr xmlLoadSGMLSuperCatalog(const char *filename);
extern xmlCatalogPtr xmlNewCatalog(int sgml);
extern xmlDocPtr xmlParseCatalogFile(const char *filename);
```

8.2.6 libxml2/libxml/debugXML.h

```
typedef char *(*xmlShellReadlineFunc) (char *);
typedef struct _xmlShellCtxt {
    char *filename;
    xmlDocPtr doc;
    xmlNodePtr node;
    xmlXPathContextPtr pctxt;
    int loaded;
    FILE *output;
    xmlShellReadlineFunc input;
} xmlShellCtxt;
typedef xmlShellCtxt *xmlShellCtxtPtr;
typedef int (*xmlShellCmd) (xmlShellCtxtPtr, char *, xmlNodePtr,
                            xmlNodePtr);
extern const char *xmlBoolToText(int boolval);
extern int xmlDebugCheckDocument(FILE * output, xmlDocPtr doc);
extern void xmlDebugDumpAttr(FILE * output, xmlAttrPtr attr, int
depth);
extern void xmlDebugDumpAttrList(FILE * output, xmlAttrPtr attr,
                                 int depth);
extern void xmlDebugDumpDTD(FILE * output, xmlDtdPtr dtd);
extern void xmlDebugDumpDocument(FILE * output, xmlDocPtr doc);
extern void xmlDebugDumpDocumentHead(FILE * output, xmlDocPtr doc);
extern void xmlDebugDumpEntities(FILE * output, xmlDocPtr doc);
extern void xmlDebugDumpNode(FILE * output, xmlNodePtr node, int
depth);
extern void xmlDebugDumpNodeList(FILE * output, xmlNodePtr node,
                                 int depth);
extern void xmlDebugDumpOneNode(FILE * output, xmlNodePtr node, int
depth);
```

```
extern void xmlDebugDumpString(FILE * output, const xmlChar * str);
extern int xmlLsCountNode(xmlNodePtr node);
extern void xmlLsOneNode(FILE * output, xmlNodePtr node);
extern void xmlShell(xmlDocPtr doc, char *filename,
                     xmlShellReadlineFunc input, FILE * output);
extern int xmlShellBase(xmlShellCtxtPtr ctxt, char *arg, xmlNodePtr
node,
                        xmlNodePtr node2);
extern int xmlShellCat(xmlShellCtxtPtr ctxt, char *arg, xmlNodePtr
node,
                       xmlNodePtr node2);
extern int xmlShellDir(xmlShellCtxtPtr ctxt, char *arg, xmlNodePtr
node,
                       xmlNodePtr node2);
extern int xmlShellDu(xmlShellCtxtPtr ctxt, char *arg, xmlNodePtr
tree,
                      xmlNodePtr node2);
extern int xmlShellList(xmlShellCtxtPtr ctxt, char *arg, xmlNodePtr
node,
                        xmlNodePtr node2);
extern int xmlShellLoad(xmlShellCtxtPtr ctxt, char *filename,
                        xmlNodePtr node, xmlNodePtr node2);
extern void xmlShellPrintNode(xmlNodePtr node);
extern void xmlShellPrintXPathError(int errorType, const char *arg);
extern void xmlShellPrintXPathResult(xmlXPathObjectPtr list);
extern   int   xmlShellPwd(xmlShellCtxtPtr   ctxt,   char   *buffer,
xmlNodePtr node,
                       xmlNodePtr node2);
extern int xmlShellSave(xmlShellCtxtPtr ctxt, char *filename,
                        xmlNodePtr node, xmlNodePtr node2);
extern int xmlShellValidate(xmlShellCtxtPtr ctxt, char *dtd,
                            xmlNodePtr node, xmlNodePtr node2);
extern int xmlShellWrite(xmlShellCtxtPtr ctxt, char *filename,
                         xmlNodePtr node, xmlNodePtr node2);
```

8.2.7 libxml2/libxml/dict.h

```
typedef struct _xmlDict xmlDict;
typedef xmlDict *xmlDictPtr;
extern void xmlDictCleanup(void);
extern xmlDictPtr xmlDictCreate(void);
extern xmlDictPtr xmlDictCreateSub(xmlDictPtr);
extern const xmlChar *xmlDictExists(xmlDictPtr, const xmlChar *,
int);
extern void xmlDictFree(xmlDictPtr);
extern const xmlChar *xmlDictLookup(xmlDictPtr, const xmlChar *,
int);
extern int xmlDictOwns(xmlDictPtr, const xmlChar *);
extern const xmlChar *xmlDictQLookup(xmlDictPtr, const xmlChar *,
                                     const xmlChar *);
extern int xmlDictReference(xmlDictPtr);
extern int xmlDictSize(xmlDictPtr);
```

8.2.8 libxml2/libxml/encoding.h

```
typedef enum {
    XML_CHAR_ENCODING_ERROR = -1,
    XML_CHAR_ENCODING_NONE = 0,
    XML_CHAR_ENCODING_UTF8 = 1,
    XML_CHAR_ENCODING_UTF16LE = 2,
    XML_CHAR_ENCODING_UTF16BE = 3,
    XML_CHAR_ENCODING_UCS4LE = 4,
    XML_CHAR_ENCODING_UCS4BE = 5,
```



```
    XML_CHAR_ENCODING_EBCDIC = 6,
    XML_CHAR_ENCODING_UCS4_2143 = 7,
    XML_CHAR_ENCODING_UCS4_3412 = 8,
    XML_CHAR_ENCODING_UCS2 = 9,
    XML_CHAR_ENCODING_8859_1 = 10,
    XML_CHAR_ENCODING_8859_2 = 11,
    XML_CHAR_ENCODING_8859_3 = 12,
    XML_CHAR_ENCODING_8859_4 = 13,
    XML_CHAR_ENCODING_8859_5 = 14,
    XML_CHAR_ENCODING_8859_6 = 15,
    XML_CHAR_ENCODING_8859_7 = 16,
    XML_CHAR_ENCODING_8859_8 = 17,
    XML_CHAR_ENCODING_8859_9 = 18,
    XML_CHAR_ENCODING_2022_JP = 19,
    XML_CHAR_ENCODING_SHIFT_JIS = 20,
    XML_CHAR_ENCODING_EUC_JP = 21,
    XML_CHAR_ENCODING_ASCII = 22
} xmlCharEncoding;
extern int UTF8Toisolat1(unsigned char *out, int *outlen,
                         const unsigned char *in, int *inlen);
extern int isolat1ToUTF8(unsigned char *out, int *outlen,
                         const unsigned char *in, int *inlen);
extern int xmlAddEncodingAlias(const char *name, const char *alias);
extern int xmlCharEncCloseFunc(xmlCharEncodingHandler * handler);
extern int xmlCharEncFirstLine(xmlCharEncodingHandler * handler,
                               xmlBufferPtr out, xmlBufferPtr in);
extern int xmlCharEncInFunc(xmlCharEncodingHandler * handler,
                            xmlBufferPtr out, xmlBufferPtr in);
extern int xmlCharEncOutFunc(xmlCharEncodingHandler * handler,
                             xmlBufferPtr out, xmlBufferPtr in);
extern void xmlCleanupCharEncodingHandlers(void);
extern void xmlCleanupEncodingAliases(void);
extern int xmlDelEncodingAlias(const char *alias);
extern xmlCharEncoding xmlDetectCharEncoding(const unsigned char
*in,
                                             int len);
extern xmlCharEncodingHandlerPtr xmlFindCharEncodingHandler(const
char
                                                            *name);
extern                                          xmlCharEncodingHandlerPtr
xmlGetCharEncodingHandler(xmlCharEncoding
                                                            enc);
extern const char *xmlGetCharEncodingName(xmlCharEncoding enc);
extern const char *xmlGetEncodingAlias(const char *alias);
extern void xmlInitCharEncodingHandlers(void);
extern xmlCharEncodingHandlerPtr xmlNewCharEncodingHandler(const
char
                                                            *name,

xmlCharEncodingInputFunc
                                                            input,

xmlCharEncodingOutputFunc
                                                            output);
extern xmlCharEncoding xmlParseCharEncoding(const char *name);
extern                                                                  void
xmlRegisterCharEncodingHandler(xmlCharEncodingHandlerPtr
                                                handler);
```

8.2.9 libxml2/libxml/entities.h

```
typedef enum {
    XML_INTERNAL_GENERAL_ENTITY = 1,
    XML_EXTERNAL_GENERAL_PARSED_ENTITY = 2,
    XML_EXTERNAL_GENERAL_UNPARSED_ENTITY = 3,
```

```
    XML_INTERNAL_PARAMETER_ENTITY = 4,
    XML_EXTERNAL_PARAMETER_ENTITY = 5,
    XML_INTERNAL_PREDEFINED_ENTITY = 6
} xmlEntityType;
typedef struct _xmlEntity {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlDtd *parent;
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    xmlChar *orig;
    xmlChar *content;
    int length;
    xmlEntityType etype;
    const xmlChar *ExternalID;
    const xmlChar *SystemID;
    struct _xmlEntity *nexte;
    const xmlChar *URI;
    int owner;
} xmlEntity;
typedef xmlEntity *xmlEntityPtr;
typedef struct _xmlHashTable xmlEntitiesTable;
typedef xmlEntitiesTable *xmlEntitiesTablePtr;
extern xmlEntityPtr xmlAddDocEntity(xmlDocPtr doc, const xmlChar *
name,
                                    int type, const xmlChar * ExternalID,
                                    const xmlChar * SystemID,
                                    const xmlChar * content);
extern xmlEntityPtr xmlAddDtdEntity(xmlDocPtr doc, const xmlChar *
name,
                                    int type, const xmlChar * ExternalID,
                                    const xmlChar * SystemID,
                                    const xmlChar * content);
extern                                          xmlEntitiesTablePtr
xmlCopyEntitiesTable(xmlEntitiesTablePtr table);
extern void xmlDumpEntitiesTable(xmlBufferPtr buf,
                                 xmlEntitiesTablePtr table);
extern void xmlDumpEntityDecl(xmlBufferPtr buf, xmlEntityPtr ent);
extern xmlChar *xmlEncodeEntitiesReentrant(xmlDocPtr doc,
                                           const xmlChar * input);
extern xmlChar *xmlEncodeSpecialChars(xmlDocPtr doc,
                                      const xmlChar * input);
extern void xmlFreeEntitiesTable(xmlEntitiesTablePtr table);
extern xmlEntityPtr xmlGetDocEntity(xmlDocPtr doc, const xmlChar *
name);
extern xmlEntityPtr xmlGetDtdEntity(xmlDocPtr doc, const xmlChar *
name);
extern xmlEntityPtr xmlGetParameterEntity(xmlDocPtr doc,
                                          const xmlChar * name);
extern xmlEntityPtr xmlGetPredefinedEntity(const xmlChar * name);
```

8.2.10 libxml2/libxml/globals.h

```
#define xmlDeregisterNodeDefaultValue    \
       (*(__xmlDeregisterNodeDefaultValue()))
#define xmlDoValidityCheckingDefaultValue       \
       (*(__xmlDoValidityCheckingDefaultValue()))
#define xmlOutputBufferCreateFilenameValue      \
       (*(__xmlOutputBufferCreateFilenameValue()))
#define xmlParserInputBufferCreateFilenameValue  \
       (*(__xmlParserInputBufferCreateFilenameValue()))
```



```
#define xmlPedanticParserDefaultValue    \
       (*(__xmlPedanticParserDefaultValue()))
#define xmlSubstituteEntitiesDefaultValue       \
       (*(__xmlSubstituteEntitiesDefaultValue()))
#define docbDefaultSAXHandler   (*(__docbDefaultSAXHandler()))
#define htmlDefaultSAXHandler   (*(__htmlDefaultSAXHandler()))
#define oldXMLWDcompatibility   (*(__oldXMLWDcompatibility()))
#define xmlBufferAllocScheme    (*(__xmlBufferAllocScheme()))
#define xmlDefaultBufferSize    (*(__xmlDefaultBufferSize()))
#define xmlDefaultSAXHandler    (*(__xmlDefaultSAXHandler()))
#define xmlDefaultSAXLocator    (*(__xmlDefaultSAXLocator()))
#define xmlGenericError (*(__xmlGenericError()))
#define xmlGenericErrorContext  (*(__xmlGenericErrorContext()))
#define                                     xmlGetWarningsDefaultValue
(*(__xmlGetWarningsDefaultValue()))
#define xmlIndentTreeOutput     (*(__xmlIndentTreeOutput()))
#define                                      xmlKeepBlanksDefaultValue
(*(__xmlKeepBlanksDefaultValue()))
#define xmlLastError    (*(__xmlLastError()))
#define                                     xmlLineNumbersDefaultValue
(*(__xmlLineNumbersDefaultValue()))
#define                                      xmlLoadExtDtdDefaultValue
(*(__xmlLoadExtDtdDefaultValue()))
#define xmlParserDebugEntities  (*(__xmlParserDebugEntities()))
#define xmlParserVersion       (*(__xmlParserVersion()))
#define                                    xmlRegisterNodeDefaultValue
(*(__xmlRegisterNodeDefaultValue()))
#define xmlSaveNoEmptyTags      (*(__xmlSaveNoEmptyTags()))
#define xmlStructuredError      (*(__xmlStructuredError()))
#define xmlTreeIndentString     (*(__xmlTreeIndentString()))

typedef     xmlOutputBufferPtr(*xmlOutputBufferCreateFilenameFunc)
(const char
                                                        *,

xmlCharEncodingHandlerPtr,
                                                        int);
typedef void (*xmlRegisterNodeFunc) (xmlNodePtr);
typedef
xmlParserInputBufferPtr(*xmlParserInputBufferCreateFilenameFunc)
(const
                                                          char *,

xmlCharEncoding);
typedef struct _xmlSAXHandlerV1 {
    internalSubsetSAXFunc internalSubset;
    isStandaloneSAXFunc isStandalone;
    hasInternalSubsetSAXFunc hasInternalSubset;
    hasExternalSubsetSAXFunc hasExternalSubset;
    resolveEntitySAXFunc resolveEntity;
    getEntitySAXFunc getEntity;
    entityDeclSAXFunc entityDecl;
    notationDeclSAXFunc notationDecl;
    attributeDeclSAXFunc attributeDecl;
    elementDeclSAXFunc elementDecl;
    unparsedEntityDeclSAXFunc unparsedEntityDecl;
    setDocumentLocatorSAXFunc setDocumentLocator;
    startDocumentSAXFunc startDocument;
    endDocumentSAXFunc endDocument;
    startElementSAXFunc startElement;
    endElementSAXFunc endElement;
    referenceSAXFunc reference;
    charactersSAXFunc characters;
    ignorableWhitespaceSAXFunc ignorableWhitespace;
    processingInstructionSAXFunc processingInstruction;
    commentSAXFunc comment;
```



```
    warningSAXFunc warning;
    errorSAXFunc error;
    fatalErrorSAXFunc fatalError;
    getParameterEntitySAXFunc getParameterEntity;
    cdataBlockSAXFunc cdataBlock;
    externalSubsetSAXFunc externalSubset;
    unsigned int initialized;
} xmlSAXHandlerV1;
typedef void (*xmlDeregisterNodeFunc) (xmlNodePtr);
typedef struct _xmlGlobalState xmlGlobalState;
typedef xmlGlobalState *xmlGlobalStatePtr;
extern xmlSAXHandlerV1 *__docbDefaultSAXHandler(void);
extern xmlSAXHandlerV1 *__htmlDefaultSAXHandler(void);
extern int *__oldXMLWDcompatibility(void);
extern xmlBufferAllocationScheme *__xmlBufferAllocScheme(void);
extern int *__xmlDefaultBufferSize(void);
extern xmlSAXHandlerV1 *__xmlDefaultSAXHandler(void);
extern xmlSAXLocator *__xmlDefaultSAXLocator(void);
extern                                          xmlDeregisterNodeFunc
*__xmlDeregisterNodeDefaultValue(void);
extern int *__xmlDoValidityCheckingDefaultValue(void);
extern xmlGenericErrorFunc *__xmlGenericError(void);
extern void **__xmlGenericErrorContext(void);
extern int *__xmlGetWarningsDefaultValue(void);
extern int *__xmlIndentTreeOutput(void);
extern int *__xmlKeepBlanksDefaultValue(void);
extern xmlError *__xmlLastError(void);
extern int *__xmlLineNumbersDefaultValue(void);
extern int *__xmlLoadExtDtdDefaultValue(void);
extern xmlOutputBufferCreateFilenameFunc
   *__xmlOutputBufferCreateFilenameValue(void);
extern int *__xmlParserDebugEntities(void);
extern xmlParserInputBufferCreateFilenameFunc
   *__xmlParserInputBufferCreateFilenameValue(void);
extern const char **__xmlParserVersion(void);
extern int *__xmlPedanticParserDefaultValue(void);
extern xmlRegisterNodeFunc *__xmlRegisterNodeDefaultValue(void);
extern int *__xmlSaveNoEmptyTags(void);
extern xmlStructuredErrorFunc *__xmlStructuredError(void);
extern int *__xmlSubstituteEntitiesDefaultValue(void);
extern const char **__xmlTreeIndentString(void);
extern void xmlCleanupGlobals(void);
extern                                          xmlDeregisterNodeFunc
xmlDeregisterNodeDefault(xmlDeregisterNodeFunc
                                                 func);
extern xmlFreeFunc xmlFree;
extern void xmlInitGlobals(void);
extern void xmlInitializeGlobalState(xmlGlobalStatePtr gs);
extern xmlMallocFunc xmlMalloc;
extern xmlMallocFunc xmlMallocAtomic;
extern xmlStrdupFunc xmlMemStrdup;
extern xmlOutputBufferCreateFilenameFunc
xmlOutputBufferCreateFilenameDefault(xmlOutputBufferCreateFilenam
eFunc
                                     func);
extern                                      xmlParserInputBufferPtr
xmlParserInputBufferCreateFilename(const
                                                        char
                                                        *URI,

xmlCharEncoding
                                                        enc);
extern xmlParserInputBufferCreateFilenameFunc
xmlParserInputBufferCreateFilenameDefault
(xmlParserInputBufferCreateFilenameFunc func);
extern xmlReallocFunc xmlRealloc;
```



```
extern                                          xmlRegisterNodeFunc
xmlRegisterNodeDefault(xmlRegisterNodeFunc
                                               func);
extern xmlBufferAllocationScheme
xmlThrDefBufferAllocScheme(xmlBufferAllocationScheme v);
extern int xmlThrDefDefaultBufferSize(int v);
extern xmlDeregisterNodeFunc
xmlThrDefDeregisterNodeDefault(xmlDeregisterNodeFunc func);
extern int xmlThrDefDoValidityCheckingDefaultValue(int v);
extern int xmlThrDefGetWarningsDefaultValue(int v);
extern int xmlThrDefIndentTreeOutput(int v);
extern int xmlThrDefKeepBlanksDefaultValue(int v);
extern int xmlThrDefLineNumbersDefaultValue(int v);
extern int xmlThrDefLoadExtDtdDefaultValue(int v);
extern xmlOutputBufferCreateFilenameFunc
xmlThrDefOutputBufferCreateFilenameDefault
(xmlOutputBufferCreateFilenameFunc func);
extern int xmlThrDefParserDebugEntities(int v);
extern xmlParserInputBufferCreateFilenameFunc
xmlThrDefParserInputBufferCreateFilenameDefault
(xmlParserInputBufferCreateFilenameFunc func);
extern int xmlThrDefPedanticParserDefaultValue(int v);
extern                                          xmlRegisterNodeFunc
xmlThrDefRegisterNodeDefault(xmlRegisterNodeFunc
                                               func);
extern int xmlThrDefSaveNoEmptyTags(int v);
extern void xmlThrDefSetGenericErrorFunc(void *ctx,
                                         xmlGenericErrorFunc handler);
extern void xmlThrDefSetStructuredErrorFunc(void *ctx,
                                            xmlStructuredErrorFunc
                                            handler);
extern int xmlThrDefSubstituteEntitiesDefaultValue(int v);
extern const char *xmlThrDefTreeIndentString(const char *v);
```

8.2.11 libxml2/libxml/hash.h

```
#define XML_CAST_FPTR(fptr)    fptr

typedef struct _xmlHashTable xmlHashTable;
typedef xmlHashTable *xmlHashTablePtr;
typedef void (xmlHashDeallocator) (void *, xmlChar *);
typedef void (*xmlHashScannerFull) (void *, void *, const xmlChar
*,
                                    const xmlChar *, const xmlChar *);
typedef void *(*xmlHashCopier) (void *, xmlChar *);
typedef void (*xmlHashScanner) (void *, void *, xmlChar *);
extern int xmlHashAddEntry(xmlHashTablePtr table, const xmlChar *
name,
                           void *userdata);
extern int xmlHashAddEntry2(xmlHashTablePtr table, const xmlChar *
name,
                            const xmlChar * name2, void *userdata);
extern int xmlHashAddEntry3(xmlHashTablePtr table, const xmlChar *
name,
                            const xmlChar * name2, const xmlChar * name3,
                            void *userdata);
extern    xmlHashTablePtr    xmlHashCopy(xmlHashTablePtr    table,
xmlHashCopier f);
extern xmlHashTablePtr xmlHashCreate(int size);
extern xmlHashTablePtr xmlHashCreateDict(int size, xmlDictPtr dict);
extern void xmlHashFree(xmlHashTablePtr table, xmlHashDeallocator
f);
extern void *xmlHashLookup(xmlHashTablePtr table, const xmlChar *
name);
```

```
extern void *xmlHashLookup2(xmlHashTablePtr table, const xmlChar *
name,
                            const xmlChar * name2);
extern void *xmlHashLookup3(xmlHashTablePtr table, const xmlChar *
name,
                            const  xmlChar  *  name2,  const  xmlChar  *
name3);
extern void *xmlHashQLookup(xmlHashTablePtr table, const xmlChar *
name,
                            const xmlChar * prefix);
extern void *xmlHashQLookup2(xmlHashTablePtr table, const xmlChar
* name,
                            const  xmlChar  *  prefix,  const  xmlChar  *
name2,
                            const xmlChar * prefix2);
extern void *xmlHashQLookup3(xmlHashTablePtr table, const xmlChar
* name,
                            const  xmlChar  *  prefix,  const  xmlChar  *
name2,
                            const xmlChar * prefix2,
                            const xmlChar * name3,
                            const xmlChar * prefix3);
extern int xmlHashRemoveEntry(xmlHashTablePtr table, const xmlChar
* name,
                             xmlHashDeallocator f);
extern int xmlHashRemoveEntry2(xmlHashTablePtr table, const xmlChar
* name,
                              const xmlChar * name2,
                              xmlHashDeallocator f);
extern int xmlHashRemoveEntry3(xmlHashTablePtr table, const xmlChar
* name,
                              const xmlChar * name2,
                              const xmlChar * name3,
                              xmlHashDeallocator f);
extern void xmlHashScan(xmlHashTablePtr table, xmlHashScanner f,
                        void *data);
extern  void  xmlHashScan3(xmlHashTablePtr  table,  const  xmlChar  *
name,
                         const xmlChar * name2, const xmlChar * name3,
                         xmlHashScanner f, void *data);
extern       void       xmlHashScanFull(xmlHashTablePtr       table,
xmlHashScannerFull f,
                            void *data);
extern void xmlHashScanFull3(xmlHashTablePtr table, const xmlChar
* name,
                            const  xmlChar  *  name2,  const  xmlChar  *
name3,
                            xmlHashScannerFull f, void *data);
extern int xmlHashSize(xmlHashTablePtr table);
extern int xmlHashUpdateEntry(xmlHashTablePtr table, const xmlChar
* name,
                             void *userdata, xmlHashDeallocator f);
extern int xmlHashUpdateEntry2(xmlHashTablePtr table, const xmlChar
* name,
                              const xmlChar * name2, void *userdata,
                              xmlHashDeallocator f);
extern int xmlHashUpdateEntry3(xmlHashTablePtr table, const xmlChar
* name,
                              const xmlChar * name2,
                              const xmlChar * name3, void *userdata,
                              xmlHashDeallocator f);
```

8.2.12 libxml2/libxml/list.h

```
typedef struct _xmlList xmlList;
```



```
typedef xmlList *xmlListPtr;
typedef struct _xmlLink xmlLink;
typedef xmlLink *xmlLinkPtr;
typedef int (*xmlListWalker) (const void *, const void *);
typedef void (*xmlListDeallocator) (xmlLinkPtr);
typedef int (*xmlListDataCompare) (const void *, const void *);
extern void *xmlLinkGetData(xmlLinkPtr lk);
extern int xmlListAppend(xmlListPtr l, void *data);
extern void xmlListClear(xmlListPtr l);
extern int xmlListCopy(xmlListPtr cur, const xmlListPtr old);
extern xmlListPtr xmlListCreate(xmlListDeallocator deallocator,
                                xmlListDataCompare compare);
extern void xmlListDelete(xmlListPtr l);
extern xmlListPtr xmlListDup(const xmlListPtr old);
extern int xmlListEmpty(xmlListPtr l);
extern xmlLinkPtr xmlListEnd(xmlListPtr l);
extern xmlLinkPtr xmlListFront(xmlListPtr l);
extern int xmlListInsert(xmlListPtr l, void *data);
extern void xmlListMerge(xmlListPtr l1, xmlListPtr l2);
extern void xmlListPopBack(xmlListPtr l);
extern void xmlListPopFront(xmlListPtr l);
extern int xmlListPushBack(xmlListPtr l, void *data);
extern int xmlListPushFront(xmlListPtr l, void *data);
extern int xmlListRemoveAll(xmlListPtr l, void *data);
extern int xmlListRemoveFirst(xmlListPtr l, void *data);
extern int xmlListRemoveLast(xmlListPtr l, void *data);
extern void xmlListReverse(xmlListPtr l);
extern void *xmlListReverseSearch(xmlListPtr l, void *data);
extern void xmlListReverseWalk(xmlListPtr l, xmlListWalker walker,
                               const void *user);
extern void *xmlListSearch(xmlListPtr l, void *data);
extern int xmlListSize(xmlListPtr l);
extern void xmlListSort(xmlListPtr l);
extern void xmlListWalk(xmlListPtr l, xmlListWalker walker,
                        const void *user);
```

8.2.13 libxml2/libxml/parser.h

```
#define XML_DEFAULT_VERSION    "1.0"
#define XML_SAX2_MAGIC  0xDEEDBEAF
#define XML_DETECT_IDS  2
#define XML_COMPLETE_ATTRS     4
#define XML_SKIP_IDS    8

typedef xmlSAXHandler *xmlSAXHandlerPtr;
typedef xmlParserNodeInfoSeq *xmlParserNodeInfoSeqPtr;
typedef xmlParserInputPtr(*xmlExternalEntityLoader) (const char *,
                                                     const char *,
                                                     xmlParserCtxtPtr);
typedef xmlParserNodeInfo *xmlParserNodeInfoPtr;
typedef enum {
    XML_WITH_THREAD = 1,
    XML_WITH_TREE = 2,
    XML_WITH_OUTPUT = 3,
    XML_WITH_PUSH = 4,
    XML_WITH_READER = 5,
    XML_WITH_PATTERN = 6,
    XML_WITH_WRITER = 7,
    XML_WITH_SAX1 = 8,
    XML_WITH_FTP = 9,
    XML_WITH_HTTP = 10,
    XML_WITH_VALID = 11,
    XML_WITH_HTML = 12,
    XML_WITH_LEGACY = 13,
    XML_WITH_C14N = 14,
```

```
    XML_WITH_CATALOG = 15,
    XML_WITH_XPATH = 16,
    XML_WITH_XPTR = 17,
    XML_WITH_XINCLUDE = 18,
    XML_WITH_ICONV = 19,
    XML_WITH_ISO8859X = 20,
    XML_WITH_UNICODE = 21,
    XML_WITH_REGEXP = 22,
    XML_WITH_AUTOMATA = 23,
    XML_WITH_EXPR = 24,
    XML_WITH_SCHEMAS = 25,
    XML_WITH_SCHEMATRON = 26,
    XML_WITH_MODULES = 27,
    XML_WITH_DEBUG = 28,
    XML_WITH_DEBUG_MEM = 29,
    XML_WITH_DEBUG_RUN = 30,
    XML_WITH_NONE = 99999
} xmlFeature;
typedef enum {
    XML_PARSE_RECOVER = 1 << 0,
    XML_PARSE_NOENT = 1 << 1,
    XML_PARSE_DTDLOAD = 1 << 2,
    XML_PARSE_DTDATTR = 1 << 3,
    XML_PARSE_DTDVALID = 1 << 4,
    XML_PARSE_NOERROR = 1 << 5,
    XML_PARSE_NOWARNING = 1 << 6,
    XML_PARSE_PEDANTIC = 1 << 7,
    XML_PARSE_NOBLANKS = 1 << 8,
    XML_PARSE_SAX1 = 1 << 9,
    XML_PARSE_XINCLUDE = 1 << 10,
    XML_PARSE_NONET = 1 << 11,
    XML_PARSE_NODICT = 1 << 12,
    XML_PARSE_NSCLEAN = 1 << 13,
    XML_PARSE_NOCDATA = 1 << 14,
    XML_PARSE_NOXINCNODE = 1 << 15,
    XML_PARSE_COMPACT = 1 << 16
} xmlParserOption;
extern long int xmlByteConsumed(xmlParserCtxtPtr ctxt);
extern void xmlCleanupParser(void);
extern void xmlClearNodeInfoSeq(xmlParserNodeInfoSeqPtr seq);
extern void xmlClearParserCtxt(xmlParserCtxtPtr ctxt);
extern xmlParserCtxtPtr xmlCreateDocParserCtxt(const xmlChar * cur);
extern xmlParserCtxtPtr xmlCreateIOParserCtxt(xmlSAXHandlerPtr sax,
                                              void *user_data,
                                              xmlInputReadCallback  ioread,
                                              xmlInputCloseCallback
                                              ioclose, void *ioctx,
                                              xmlCharEncoding enc);
extern  xmlParserCtxtPtr  xmlCreatePushParserCtxt(xmlSAXHandlerPtr
sax,
                                                  void *user_data,
                                                  const char *chunk,
                                                  int size,
                                                  const char *filename);
extern   xmlDocPtr   xmlCtxtReadDoc(xmlParserCtxtPtr   ctxt,   const
xmlChar * cur,
                                    const char *URL, const char *encoding,
                                    int options);
extern xmlDocPtr xmlCtxtReadFd(xmlParserCtxtPtr ctxt, int fd,
                               const char *URL, const char *encoding,
                               int options);
extern xmlDocPtr xmlCtxtReadFile(xmlParserCtxtPtr ctxt,
                                 const char *filename,
                                 const char *encoding, int options);
extern xmlDocPtr xmlCtxtReadIO(xmlParserCtxtPtr ctxt,
                               xmlInputReadCallback ioread,
```



```
                            xmlInputCloseCallback   ioclose,   void
*ioctx,
                            const char *URL, const char *encoding,
                            int options);
extern xmlDocPtr xmlCtxtReadMemory(xmlParserCtxtPtr ctxt,
                               const char *buffer, int size,
                               const char *URL, const char *encoding,
                               int options);
extern void xmlCtxtReset(xmlParserCtxtPtr ctxt);
extern  int  xmlCtxtResetPush(xmlParserCtxtPtr  ctxt,  const  char
*chunk,
                          int size, const char *filename,
                          const char *encoding);
extern int xmlCtxtUseOptions(xmlParserCtxtPtr ctxt, int options);
extern void xmlFreeParserCtxt(xmlParserCtxtPtr ctxt);
extern xmlExternalEntityLoader xmlGetExternalEntityLoader(void);
extern int xmlHasFeature(xmlFeature feature);
extern xmlDtdPtr xmlIOParseDTD(xmlSAXHandlerPtr sax,
                           xmlParserInputBufferPtr input,
                           xmlCharEncoding enc);
extern void xmlInitNodeInfoSeq(xmlParserNodeInfoSeqPtr seq);
extern void xmlInitParser(void);
extern int xmlInitParserCtxt(xmlParserCtxtPtr ctxt);
extern int xmlKeepBlanksDefault(int val);
extern int xmlLineNumbersDefault(int val);
extern xmlParserInputPtr xmlLoadExternalEntity(const char *URL,
                                       const char *ID,
                                       xmlParserCtxtPtr ctxt);
extern xmlParserInputPtr xmlNewIOInputStream(xmlParserCtxtPtr ctxt,
                                       xmlParserInputBufferPtr
input,
                                       xmlCharEncoding enc);
extern xmlParserCtxtPtr xmlNewParserCtxt(void);
extern     int     xmlParseBalancedChunkMemory(xmlDocPtr     doc,
xmlSAXHandlerPtr sax,
                               void *user_data, int depth,
                               const xmlChar * string,
                               xmlNodePtr * lst);
extern int xmlParseBalancedChunkMemoryRecover(xmlDocPtr doc,
                                       xmlSAXHandlerPtr sax,
                                       void *user_data, int depth,
                                       const xmlChar * string,
                                       xmlNodePtr * lst,
                                       int recover);
extern int xmlParseChunk(xmlParserCtxtPtr ctxt, const char *chunk,
                     int size, int terminate);
extern int xmlParseCtxtExternalEntity(xmlParserCtxtPtr ctx,
                                const xmlChar * URL,
                                const xmlChar * ID,
                                xmlNodePtr * lst);
extern xmlDtdPtr xmlParseDTD(const xmlChar * ExternalID,
                         const xmlChar * SystemID);
extern xmlDocPtr xmlParseDoc(const xmlChar * cur);
extern int xmlParseDocument(xmlParserCtxtPtr ctxt);
extern xmlDocPtr xmlParseEntity(const char *filename);
extern int xmlParseExtParsedEnt(xmlParserCtxtPtr ctxt);
extern int xmlParseExternalEntity(xmlDocPtr doc, xmlSAXHandlerPtr
sax,
                            void *user_data, int depth,
                            const xmlChar * URL, const xmlChar *
ID,
                            xmlNodePtr * lst);
extern xmlDocPtr xmlParseFile(const char *filename);
extern xmlParserErrors xmlParseInNodeContext(xmlNodePtr node,
                                       const char *data, int datalen,
                                       int options,
```



```
                                        xmlNodePtr * lst);
extern xmlDocPtr xmlParseMemory(const char *buffer, int size);
extern void xmlParserAddNodeInfo(xmlParserCtxtPtr ctxt,
                                 const xmlParserNodeInfoPtr info);
extern const xmlParserNodeInfo *xmlParserFindNodeInfo(const
                                                      xmlParserCtxtPtr
                                                      ctxt,
                                                      const xmlNodePtr
                                                      node);
extern long unsigned int xmlParserFindNodeInfoIndex(const

xmlParserNodeInfoSeqPtr
                                                    seq,
                                                    const xmlNodePtr node);
extern int xmlParserInputGrow(xmlParserInputPtr in, int len);
extern int xmlParserInputRead(xmlParserInputPtr in, int len);
extern int xmlPedanticParserDefault(int val);
extern xmlDocPtr xmlReadDoc(const xmlChar * cur, const char *URL,
                            const char *encoding, int options);
extern xmlDocPtr xmlReadFd(int fd, const char *URL, const char
*encoding,
                           int options);
extern xmlDocPtr xmlReadFile(const char *URL, const char *encoding,
                             int options);
extern xmlDocPtr xmlReadIO(xmlInputReadCallback ioread,
                           xmlInputCloseCallback ioclose, void *ioctx,
                           const char *URL, const char *encoding,
                           int options);
extern xmlDocPtr xmlReadMemory(const char *buffer, int size,
                               const char *URL, const char *encoding,
                               int options);
extern xmlDocPtr xmlRecoverDoc(const xmlChar * cur);
extern xmlDocPtr xmlRecoverFile(const char *filename);
extern xmlDocPtr xmlRecoverMemory(const char *buffer, int size);
extern xmlDtdPtr xmlSAXParseDTD(xmlSAXHandlerPtr sax,
                                const xmlChar * ExternalID,
                                const xmlChar * SystemID);
extern xmlDocPtr xmlSAXParseDoc(xmlSAXHandlerPtr sax, const xmlChar
* cur,
                                int recovery);
extern xmlDocPtr xmlSAXParseEntity(xmlSAXHandlerPtr sax,
                                   const char *filename);
extern xmlDocPtr xmlSAXParseFile(xmlSAXHandlerPtr sax,
                                 const char *filename, int recovery);
extern xmlDocPtr xmlSAXParseFileWithData(xmlSAXHandlerPtr sax,
                                         const char *filename,
                                         int recovery, void *data);
extern xmlDocPtr xmlSAXParseMemory(xmlSAXHandlerPtr sax,
                                   const char *buffer, int size,
                                   int recovery);
extern xmlDocPtr xmlSAXParseMemoryWithData(xmlSAXHandlerPtr sax,
                                           const char *buffer, int size,
                                           int recovery, void *data);
extern   int   xmlSAXUserParseFile(xmlSAXHandlerPtr   sax,   void
*user_data,
                                   const char *filename);
extern   int   xmlSAXUserParseMemory(xmlSAXHandlerPtr   sax,   void
*user_data,
                                     const char *buffer, int size);
extern void xmlSetExternalEntityLoader(xmlExternalEntityLoader f);
extern void xmlSetupParserForBuffer(xmlParserCtxtPtr ctxt,
                                    const xmlChar * buffer,
                                    const char *filename);
extern void xmlStopParser(xmlParserCtxtPtr ctxt);
extern int xmlSubstituteEntitiesDefault(int val);
```


8.2.14 libxml2/libxml/parserInternals.h

```
#define IS_ASCII_LETTER(c)      \
      (((0x41 <= (c)) && ((c) <= 0x5a)) || ((0x61 <= (c)) && ((c)
<= \
      0x7a)))
#define SKIP_EOL(p)     \
      if (*(p) == 0x13) { p++ ; if (*(p) == 0x10) p++; } if (*(p)
== 0x10) \
      { p++ ; if (*(p) == 0x13) p++; }
#define IS_ASCII_DIGIT(c)      ((0x30 <= (c)) && ((c) <= 0x39))
#define IS_LETTER(c)    (IS_BASECHAR(c) || IS_IDEOGRAPHIC(c))
#define IS_COMBINING_CH(c)     0
#define XML_SUBSTITUTE_NONE    0
#define XML_SUBSTITUTE_REF     1
#define XML_MAX_NAMELEN 100
#define XML_SUBSTITUTE_PEREF    2
#define INPUT_CHUNK    250
#define XML_SUBSTITUTE_BOTH    3
#define MOVETO_STARTTAG(p)     while ((*p) && (*(p) != '<')) (p)++
#define MOVETO_ENDTAG(p)       while ((*p) && (*(p) != '>')) (p)++
#define IS_BASECHAR(c)  xmlIsBaseCharQ(c)
#define IS_LETTER_CH(c) xmlIsBaseChar_ch(c)
#define IS_BLANK(c)     xmlIsBlankQ(c)
#define IS_BLANK_CH(c)  xmlIsBlank_ch(c)
#define IS_CHAR(c)      xmlIsCharQ(c)
#define IS_BYTE_CHAR(c) xmlIsChar_ch(c)
#define IS_CHAR_CH(c)   xmlIsChar_ch(c)
#define IS_COMBINING(c) xmlIsCombiningQ(c)
#define IS_DIGIT(c)     xmlIsDigitQ(c)
#define IS_DIGIT_CH(c)  xmlIsDigit_ch(c)
#define IS_EXTENDER(c)  xmlIsExtenderQ(c)
#define IS_EXTENDER_CH(c)      xmlIsExtender_ch(c)
#define IS_IDEOGRAPHIC(c)      xmlIsIdeographicQ(c)
#define IS_PUBIDCHAR(c) xmlIsPubidCharQ(c)
#define IS_PUBIDCHAR_CH(c)     xmlIsPubidChar_ch(c)

typedef void (*xmlEntityReferenceFunc) (xmlEntityPtr, xmlNodePtr,
                                        xmlNodePtr);
extern   htmlParserCtxtPtr   htmlCreateFileParserCtxt(const   char
*filename,
                                                      const char *encoding);
extern void htmlInitAutoClose(void);
extern xmlParserInputPtr inputPop(xmlParserCtxtPtr ctxt);
extern int  inputPush(xmlParserCtxtPtr  ctxt,  xmlParserInputPtr
value);
extern const xmlChar *namePop(xmlParserCtxtPtr ctxt);
extern int namePush(xmlParserCtxtPtr ctxt, const xmlChar * value);
extern xmlNodePtr nodePop(xmlParserCtxtPtr ctxt);
extern int nodePush(xmlParserCtxtPtr ctxt, xmlNodePtr value);
extern int xmlCopyChar(int len, xmlChar * out, int val);
extern int xmlCopyCharMultiByte(xmlChar * out, int val);
extern xmlParserCtxtPtr xmlCreateEntityParserCtxt(const xmlChar *
URL,
                                                  const xmlChar * ID,
                                                  const xmlChar * base);
extern    xmlParserCtxtPtr    xmlCreateFileParserCtxt(const    char
*filename);
extern   xmlParserCtxtPtr   xmlCreateMemoryParserCtxt(const   char
*buffer,
                                                      int size);
extern    xmlParserCtxtPtr    xmlCreateURLParserCtxt(const    char
*filename,
                                                 int options);
extern int xmlCurrentChar(xmlParserCtxtPtr ctxt, int *len);
```



```
extern void xmlFreeInputStream(xmlParserInputPtr input);
extern int xmlIsLetter(int c);
extern  xmlParserInputPtr  xmlNewEntityInputStream(xmlParserCtxtPtr
ctxt,
                                                   xmlEntityPtr entity);
extern xmlParserInputPtr xmlNewInputFromFile(xmlParserCtxtPtr ctxt,
                                             const char *filename);
extern xmlParserInputPtr xmlNewInputStream(xmlParserCtxtPtr ctxt);
extern  xmlParserInputPtr  xmlNewStringInputStream(xmlParserCtxtPtr
ctxt,
                                                   const xmlChar * buffer);
extern void xmlNextChar(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseAttValue(xmlParserCtxtPtr ctxt);
extern const xmlChar *xmlParseAttribute(xmlParserCtxtPtr ctxt,
                                        xmlChar * *value);
extern void xmlParseAttributeListDecl(xmlParserCtxtPtr ctxt);
extern int xmlParseAttributeType(xmlParserCtxtPtr ctxt,
                                 xmlEnumerationPtr * tree);
extern void xmlParseCDSect(xmlParserCtxtPtr ctxt);
extern void xmlParseCharData(xmlParserCtxtPtr ctxt, int cdata);
extern int xmlParseCharRef(xmlParserCtxtPtr ctxt);
extern void xmlParseComment(xmlParserCtxtPtr ctxt);
extern void xmlParseContent(xmlParserCtxtPtr ctxt);
extern  int  xmlParseDefaultDecl(xmlParserCtxtPtr  ctxt,  xmlChar  *
*value);
extern void xmlParseDocTypeDecl(xmlParserCtxtPtr ctxt);
extern void xmlParseElement(xmlParserCtxtPtr ctxt);
extern xmlElementContentPtr
xmlParseElementChildrenContentDecl(xmlParserCtxtPtr    ctxt,    int
inputchk);
extern int xmlParseElementContentDecl(xmlParserCtxtPtr ctxt,
                                      const xmlChar * name,
                                      xmlElementContentPtr * result);
extern int xmlParseElementDecl(xmlParserCtxtPtr ctxt);
extern xmlElementContentPtr
xmlParseElementMixedContentDecl(xmlParserCtxtPtr     ctxt,     int
inputchk);
extern xmlChar *xmlParseEncName(xmlParserCtxtPtr ctxt);
extern const xmlChar *xmlParseEncodingDecl(xmlParserCtxtPtr ctxt);
extern void xmlParseEndTag(xmlParserCtxtPtr ctxt);
extern void xmlParseEntityDecl(xmlParserCtxtPtr ctxt);
extern xmlEntityPtr xmlParseEntityRef(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseEntityValue(xmlParserCtxtPtr ctxt,
                                    xmlChar * *orig);
extern int xmlParseEnumeratedType(xmlParserCtxtPtr ctxt,
                                  xmlEnumerationPtr * tree);
extern xmlEnumerationPtr  xmlParseEnumerationType(xmlParserCtxtPtr
ctxt);
extern xmlChar *xmlParseExternalID(xmlParserCtxtPtr ctxt,
                                   xmlChar * *publicID, int strict);
extern void xmlParseExternalSubset(xmlParserCtxtPtr ctxt,
                                   const xmlChar * ExternalID,
                                   const xmlChar * SystemID);
extern void xmlParseMarkupDecl(xmlParserCtxtPtr ctxt);
extern void xmlParseMisc(xmlParserCtxtPtr ctxt);
extern const xmlChar *xmlParseName(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseNmtoken(xmlParserCtxtPtr ctxt);
extern void xmlParseNotationDecl(xmlParserCtxtPtr ctxt);
extern   xmlEnumerationPtr   xmlParseNotationType(xmlParserCtxtPtr
ctxt);
extern void xmlParsePEReference(xmlParserCtxtPtr ctxt);
extern void xmlParsePI(xmlParserCtxtPtr ctxt);
extern const xmlChar *xmlParsePITarget(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParsePubidLiteral(xmlParserCtxtPtr ctxt);
extern void xmlParseReference(xmlParserCtxtPtr ctxt);
extern int xmlParseSDDecl(xmlParserCtxtPtr ctxt);
```



```
extern const xmlChar *xmlParseStartTag(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseSystemLiteral(xmlParserCtxtPtr ctxt);
extern void xmlParseTextDecl(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseVersionInfo(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlParseVersionNum(xmlParserCtxtPtr ctxt);
extern void xmlParseXMLDecl(xmlParserCtxtPtr ctxt);
extern void xmlParserHandlePEReference(xmlParserCtxtPtr ctxt);
extern void xmlParserInputShrink(xmlParserInputPtr in);
extern unsigned int xmlParserMaxDepth;
extern xmlChar xmlPopInput(xmlParserCtxtPtr ctxt);
extern  int  xmlPushInput(xmlParserCtxtPtr  ctxt,  xmlParserInputPtr
input);
extern void xmlSetEntityReferenceFunc(xmlEntityReferenceFunc func);
extern int xmlSkipBlankChars(xmlParserCtxtPtr ctxt);
extern xmlChar *xmlSplitQName(xmlParserCtxtPtr ctxt, const xmlChar
* name,
                              xmlChar * *prefix);
extern const xmlChar const xmlStringComment[];
extern   int   xmlStringCurrentChar(xmlParserCtxtPtr   ctxt,   const
xmlChar * cur,
                                    int *len);
extern xmlChar *xmlStringDecodeEntities(xmlParserCtxtPtr ctxt,
                                        const xmlChar * str, int what,
                                        xmlChar end, xmlChar end2,
                                        xmlChar end3);
extern xmlChar *xmlStringLenDecodeEntities(xmlParserCtxtPtr ctxt,
                                           const xmlChar * str, int len,
                                           int what, xmlChar end,
                                           xmlChar end2, xmlChar end3);
extern const xmlChar const xmlStringText[];
extern const xmlChar const xmlStringTextNoenc[];
extern      int      xmlSwitchEncoding(xmlParserCtxtPtr      ctxt,
xmlCharEncoding enc);
extern int xmlSwitchInputEncoding(xmlParserCtxtPtr ctxt,
                                  xmlParserInputPtr input,
                                  xmlCharEncodingHandlerPtr handler);
extern int xmlSwitchToEncoding(xmlParserCtxtPtr ctxt,
                               xmlCharEncodingHandlerPtr handler);
```

8.2.15 libxml2/libxml/pattern.h

```
typedef struct _xmlStreamCtxt xmlStreamCtxt;
typedef xmlStreamCtxt *xmlStreamCtxtPtr;
typedef struct _xmlPattern xmlPattern;
typedef xmlPattern *xmlPatternPtr;
typedef enum {
    XML_PATTERN_DEFAULT = 0,
    XML_PATTERN_XPATH = 1 << 0,
    XML_PATTERN_XSSEL = 1 << 1,
    XML_PATTERN_XSFIELD = 1 << 2
} xmlPatternFlags;
extern void xmlFreePattern(xmlPatternPtr comp);
extern void xmlFreePatternList(xmlPatternPtr comp);
extern void xmlFreeStreamCtxt(xmlStreamCtxtPtr stream);
extern int xmlPatternFromRoot(xmlPatternPtr comp);
extern    xmlStreamCtxtPtr    xmlPatternGetStreamCtxt(xmlPatternPtr
comp);
extern int xmlPatternMatch(xmlPatternPtr comp, xmlNodePtr node);
extern int xmlPatternMaxDepth(xmlPatternPtr comp);
extern int xmlPatternMinDepth(xmlPatternPtr comp);
extern int xmlPatternStreamable(xmlPatternPtr comp);
extern xmlPatternPtr xmlPatterncompile(const xmlChar * pattern,
                                       xmlDict * dict, int flags,
                                       const xmlChar * *namespaces);
extern int xmlStreamPop(xmlStreamCtxtPtr stream);
```

```
extern int xmlStreamPush(xmlStreamCtxtPtr stream, const xmlChar *
name,
                         const xmlChar * ns);
extern int xmlStreamPushAttr(xmlStreamCtxtPtr stream, const xmlChar
* name,
                             const xmlChar * ns);
```

8.2.16 libxml2/libxml/relaxng.h

```
typedef struct _xmlRelaxNGParserCtxt xmlRelaxNGParserCtxt;
typedef xmlRelaxNGParserCtxt *xmlRelaxNGParserCtxtPtr;
typedef struct _xmlRelaxNGValidCtxt xmlRelaxNGValidCtxt;
typedef xmlRelaxNGValidCtxt *xmlRelaxNGValidCtxtPtr;
typedef struct _xmlRelaxNG xmlRelaxNG;
typedef xmlRelaxNG *xmlRelaxNGPtr;
typedef  void  (*xmlRelaxNGValidityErrorFunc)  (void  *,  const  char
*, ...);
typedef void (*xmlRelaxNGValidityWarningFunc) (void *, const char
*, ...);
typedef enum {
    XML_RELAXNG_OK = 0,
    XML_RELAXNG_ERR_MEMORY,
    XML_RELAXNG_ERR_TYPE,
    XML_RELAXNG_ERR_TYPEVAL,
    XML_RELAXNG_ERR_DUPID,
    XML_RELAXNG_ERR_TYPECMP,
    XML_RELAXNG_ERR_NOSTATE,
    XML_RELAXNG_ERR_NODEFINE,
    XML_RELAXNG_ERR_LISTEXTRA,
    XML_RELAXNG_ERR_LISTEMPTY,
    XML_RELAXNG_ERR_INTERNODATA,
    XML_RELAXNG_ERR_INTERSEQ,
    XML_RELAXNG_ERR_INTEREXTRA,
    XML_RELAXNG_ERR_ELEMNAME,
    XML_RELAXNG_ERR_ATTRNAME,
    XML_RELAXNG_ERR_ELEMNONS,
    XML_RELAXNG_ERR_ATTRNONS,
    XML_RELAXNG_ERR_ELEMWRONGNS,
    XML_RELAXNG_ERR_ATTRWRONGNS,
    XML_RELAXNG_ERR_ELEMEXTRANS,
    XML_RELAXNG_ERR_ATTREXTRANS,
    XML_RELAXNG_ERR_ELEMNOTEMPTY,
    XML_RELAXNG_ERR_NOELEM,
    XML_RELAXNG_ERR_NOTELEM,
    XML_RELAXNG_ERR_ATTRVALID,
    XML_RELAXNG_ERR_CONTENTVALID,
    XML_RELAXNG_ERR_EXTRACONTENT,
    XML_RELAXNG_ERR_INVALIDATTR,
    XML_RELAXNG_ERR_DATAELEM,
    XML_RELAXNG_ERR_VALELEM,
    XML_RELAXNG_ERR_LISTELEM,
    XML_RELAXNG_ERR_DATATYPE,
    XML_RELAXNG_ERR_VALUE,
    XML_RELAXNG_ERR_LIST,
    XML_RELAXNG_ERR_NOGRAMMAR,
    XML_RELAXNG_ERR_EXTRADATA,
    XML_RELAXNG_ERR_LACKDATA,
    XML_RELAXNG_ERR_INTERNAL,
    XML_RELAXNG_ERR_ELEMWRONG,
    XML_RELAXNG_ERR_TEXTWRONG
} xmlRelaxNGValidErr;
typedef enum {
    XML_RELAXNGP_NONE = 0,
    XML_RELAXNGP_FREE_DOC = 1,
    XML_RELAXNGP_CRNG = 2
```



```
} xmlRelaxNGParserFlag;
extern void xmlRelaxNGCleanupTypes(void);
extern void xmlRelaxNGDump(FILE * output, xmlRelaxNGPtr schema);
extern void xmlRelaxNGDumpTree(FILE * output, xmlRelaxNGPtr schema);
extern void xmlRelaxNGFree(xmlRelaxNGPtr schema);
extern void xmlRelaxNGFreeParserCtxt(xmlRelaxNGParserCtxtPtr ctxt);
extern void xmlRelaxNGFreeValidCtxt(xmlRelaxNGValidCtxtPtr ctxt);
extern int xmlRelaxNGGetParserErrors(xmlRelaxNGParserCtxtPtr ctxt,
                                     xmlRelaxNGValidityErrorFunc * err,
                                     xmlRelaxNGValidityWarningFunc     *
warn,
                                     void **ctx);
extern int xmlRelaxNGGetValidErrors(xmlRelaxNGValidCtxtPtr ctxt,
                                    xmlRelaxNGValidityErrorFunc * err,
                                    xmlRelaxNGValidityWarningFunc * warn,
                                    void **ctx);
extern int xmlRelaxNGInitTypes(void);
extern                                          xmlRelaxNGParserCtxtPtr
xmlRelaxNGNewDocParserCtxt(xmlDocPtr doc);
extern   xmlRelaxNGParserCtxtPtr   xmlRelaxNGNewMemParserCtxt(const
char
                                                  *buffer,
                                                  int size);
extern xmlRelaxNGParserCtxtPtr xmlRelaxNGNewParserCtxt(const char
*URL);
extern xmlRelaxNGValidCtxtPtr xmlRelaxNGNewValidCtxt(xmlRelaxNGPtr
schema);
extern xmlRelaxNGPtr xmlRelaxNGParse(xmlRelaxNGParserCtxtPtr ctxt);
extern void xmlRelaxNGSetParserErrors(xmlRelaxNGParserCtxtPtr ctxt,
                                      xmlRelaxNGValidityErrorFunc err,
                                      xmlRelaxNGValidityWarningFunc warn,
                                      void *ctx);
extern void xmlRelaxNGSetValidErrors(xmlRelaxNGValidCtxtPtr ctxt,
                                     xmlRelaxNGValidityErrorFunc err,
                                     xmlRelaxNGValidityWarningFunc warn,
                                     void *ctx);
extern                                                      void
xmlRelaxNGSetValidStructuredErrors(xmlRelaxNGValidCtxtPtr ctxt,
                                           xmlStructuredErrorFunc
                                           serror, void *ctx);
extern int xmlRelaxNGValidateDoc(xmlRelaxNGValidCtxtPtr ctxt,
                                 xmlDocPtr doc);
extern   int   xmlRelaxNGValidateFullElement(xmlRelaxNGValidCtxtPtr
ctxt,
                                      xmlDocPtr doc, xmlNodePtr elem);
extern   int    xmlRelaxNGValidatePopElement(xmlRelaxNGValidCtxtPtr
ctxt,
                                     xmlDocPtr doc, xmlNodePtr elem);
extern int xmlRelaxNGValidatePushCData(xmlRelaxNGValidCtxtPtr ctxt,
                                    const xmlChar * data, int len);
extern   int   xmlRelaxNGValidatePushElement(xmlRelaxNGValidCtxtPtr
ctxt,
                                      xmlDocPtr doc, xmlNodePtr elem);
extern int xmlRelaxParserSetFlag(xmlRelaxNGParserCtxtPtr ctxt, int
flag);
```

8.2.17 libxml2/libxml/schematron.h

```
typedef struct _xmlSchematronValidCtxt xmlSchematronValidCtxt;
typedef xmlSchematronValidCtxt *xmlSchematronValidCtxtPtr;
typedef struct _xmlSchematron xmlSchematron;
typedef xmlSchematron *xmlSchematronPtr;
typedef struct _xmlSchematronParserCtxt xmlSchematronParserCtxt;
typedef xmlSchematronParserCtxt *xmlSchematronParserCtxtPtr;
typedef enum {
```

```
    XML_SCHEMATRON_OUT_QUIET = 1 << 0,
    XML_SCHEMATRON_OUT_TEXT = 1 << 1,
    XML_SCHEMATRON_OUT_XML = 1 << 2,
    XML_SCHEMATRON_OUT_FILE = 1 << 8,
    XML_SCHEMATRON_OUT_BUFFER = 1 << 9,
    XML_SCHEMATRON_OUT_IO = 1 << 10
} xmlSchematronValidOptions;
extern void xmlSchematronFree(xmlSchematronPtr schema);
extern void xmlSchematronFreeParserCtxt(xmlSchematronParserCtxtPtr
ctxt);
extern  void  xmlSchematronFreeValidCtxt(xmlSchematronValidCtxtPtr
ctxt);
extern                                      xmlSchematronParserCtxtPtr
xmlSchematronNewDocParserCtxt(xmlDocPtr
                                                            doc);
extern                                      xmlSchematronParserCtxtPtr
xmlSchematronNewMemParserCtxt(const char
                                                            *buffer,
                                                            int size);
extern xmlSchematronParserCtxtPtr xmlSchematronNewParserCtxt(const
char
                                                          *URL);
extern                                       xmlSchematronValidCtxtPtr
xmlSchematronNewValidCtxt(xmlSchematronPtr
                                                            schema,
                                                           int options);
extern                                                      xmlSchematronPtr
xmlSchematronParse(xmlSchematronParserCtxtPtr
                                          ctxt);
extern int xmlSchematronValidateDoc(xmlSchematronValidCtxtPtr ctxt,
                                    xmlDocPtr instance);
```

8.2.18 libxml2/libxml/threads.h

```
typedef struct _xmlRMutex xmlRMutex;
typedef xmlRMutex *xmlRMutexPtr;
typedef struct _xmlMutex xmlMutex;
typedef xmlMutex *xmlMutexPtr;
extern void xmlCleanupThreads(void);
extern void xmlFreeMutex(xmlMutexPtr tok);
extern void xmlFreeRMutex(xmlRMutexPtr tok);
extern xmlGlobalStatePtr xmlGetGlobalState(void);
extern int xmlGetThreadId(void);
extern void xmlInitThreads(void);
extern int xmlIsMainThread(void);
extern void xmlLockLibrary(void);
extern void xmlMutexLock(xmlMutexPtr tok);
extern void xmlMutexUnlock(xmlMutexPtr tok);
extern xmlMutexPtr xmlNewMutex(void);
extern xmlRMutexPtr xmlNewRMutex(void);
extern void xmlRMutexLock(xmlRMutexPtr tok);
extern void xmlRMutexUnlock(xmlRMutexPtr tok);
extern void xmlUnlockLibrary(void);
```

8.2.19 libxml2/libxml/tree.h

```
#define XML_GET_CONTENT(n)      \
       ((n)->type == XML_ELEMENT_NODE ? NULL : (n)->content)
#define XML_XML_NAMESPACE       \
       (const xmlChar *) "http://www.w3.org/XML/1998/namespace"
#define XML_XML_ID     (const xmlChar *) "xml:id"
#define XML_GET_LINE(n) (xmlGetLineNo(n))
#define BASE_BUFFER_SIZE       4096
```



```
#define xmlChildrenNode children
#define xmlRootNode    children
#define XML_LOCAL_NAMESPACE    XML_NAMESPACE_DECL

typedef enum {
    XML_BUFFER_ALLOC_DOUBLEIT = 0,
    XML_BUFFER_ALLOC_EXACT = 1,
    XML_BUFFER_ALLOC_IMMUTABLE = 2
} xmlBufferAllocationScheme;
typedef struct _xmlBuffer {
    xmlChar *content;
    unsigned int use;
    unsigned int size;
    xmlBufferAllocationScheme alloc;
} xmlBuffer;
typedef xmlBuffer *xmlBufferPtr;
typedef enum {
    XML_ELEMENT_NODE = 1,
    XML_ATTRIBUTE_NODE = 2,
    XML_TEXT_NODE = 3,
    XML_CDATA_SECTION_NODE = 4,
    XML_ENTITY_REF_NODE = 5,
    XML_ENTITY_NODE = 6,
    XML_PI_NODE = 7,
    XML_COMMENT_NODE = 8,
    XML_DOCUMENT_NODE = 9,
    XML_DOCUMENT_TYPE_NODE = 10,
    XML_DOCUMENT_FRAG_NODE = 11,
    XML_NOTATION_NODE = 12,
    XML_HTML_DOCUMENT_NODE = 13,
    XML_DTD_NODE = 14,
    XML_ELEMENT_DECL = 15,
    XML_ATTRIBUTE_DECL = 16,
    XML_ENTITY_DECL = 17,
    XML_NAMESPACE_DECL = 18,
    XML_XINCLUDE_START = 19,
    XML_XINCLUDE_END = 20,
    XML_DOCB_DOCUMENT_NODE = 21
} xmlElementType;
typedef xmlElementType xmlNsType;
typedef struct _xmlNs {
    struct _xmlNs *next;
    xmlNsType type;
    const xmlChar *href;
    const xmlChar *prefix;
    void *_private;
} xmlNs;
typedef enum {
    XML_ATTRIBUTE_CDATA = 1,
    XML_ATTRIBUTE_ID = 2,
    XML_ATTRIBUTE_IDREF = 3,
    XML_ATTRIBUTE_IDREFS = 4,
    XML_ATTRIBUTE_ENTITY = 5,
    XML_ATTRIBUTE_ENTITIES = 6,
    XML_ATTRIBUTE_NMTOKEN = 7,
    XML_ATTRIBUTE_NMTOKENS = 8,
    XML_ATTRIBUTE_ENUMERATION = 9,
    XML_ATTRIBUTE_NOTATION = 10
} xmlAttributeType;
typedef struct _xmlNode {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlNode *parent;
```

```
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    xmlNs *ns;
    xmlChar *content;
    struct _xmlAttr *properties;
    xmlNs *nsDef;
    void *psvi;
    unsigned short line;
    unsigned short extra;
} xmlNode;
typedef xmlNode *xmlNodePtr;
typedef struct _xmlDoc {
    void *_private;
    xmlElementType type;
    char *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlNode *parent;
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    int compression;
    int standalone;
    struct _xmlDtd *intSubset;
    struct _xmlDtd *extSubset;
    struct _xmlNs *oldNs;
    const xmlChar *version;
    const xmlChar *encoding;
    void *ids;
    void *refs;
    const xmlChar *URL;
    int charset;
    struct _xmlDict *dict;
    void *psvi;
} xmlDoc;
typedef xmlDoc *xmlDocPtr;
typedef xmlNs *xmlNsPtr;
typedef struct _xmlDtd {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlDoc *parent;
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    void *notations;
    void *elements;
    void *attributes;
    void *entities;
    const xmlChar *ExternalID;
    const xmlChar *SystemID;
    void *pentities;
} xmlDtd;
typedef xmlDtd *xmlDtdPtr;
typedef struct _xmlDOMWrapCtxt {
    void *_private;
} xmlDOMWrapCtxt;
typedef xmlDOMWrapCtxt *xmlDOMWrapCtxtPtr;
typedef struct _xmlAttr {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
```



```
    struct _xmlNode *last;
    struct _xmlNode *parent;
    struct _xmlAttr *next;
    struct _xmlAttr *prev;
    struct _xmlDoc *doc;
    xmlNs *ns;
    xmlAttributeType atype;
    void *psvi;
} xmlAttr;
typedef xmlAttr *xmlAttrPtr;
typedef int (*xmlOutputWriteCallback) (void *, const char *, int);
typedef int (*xmlOutputCloseCallback) (void *);
typedef int (*xmlCharEncodingInputFunc) (unsigned char *, int *,
                                         const unsigned char *, int *);
typedef int (*xmlCharEncodingOutputFunc) (unsigned char *, int *,
                                          const unsigned char *, int *);
typedef struct _xmlCharEncodingHandler {
    char *name;
    xmlCharEncodingInputFunc input;
    xmlCharEncodingOutputFunc output;
    iconv_t iconv_in;
    iconv_t iconv_out;
} xmlCharEncodingHandler;
typedef xmlCharEncodingHandler *xmlCharEncodingHandlerPtr;
typedef struct _xmlOutputBuffer {
    void *context;
    xmlOutputWriteCallback writecallback;
    xmlOutputCloseCallback closecallback;
    xmlCharEncodingHandlerPtr encoder;
    xmlBufferPtr buffer;
    xmlBufferPtr conv;
    int written;
    int error;
} xmlOutputBuffer;
typedef xmlOutputBuffer *xmlOutputBufferPtr;
extern xmlNodePtr xmlAddChild(xmlNodePtr parent, xmlNodePtr cur);
extern  xmlNodePtr  xmlAddChildList(xmlNodePtr  parent,  xmlNodePtr
cur);
extern  xmlNodePtr  xmlAddNextSibling(xmlNodePtr  cur,  xmlNodePtr
elem);
extern  xmlNodePtr  xmlAddPrevSibling(xmlNodePtr  cur,  xmlNodePtr
elem);
extern xmlNodePtr xmlAddSibling(xmlNodePtr cur, xmlNodePtr elem);
extern void xmlAttrSerializeTxtContent(xmlBufferPtr buf, xmlDocPtr
doc,
                                       xmlAttrPtr attr,
                                       const xmlChar * string);
extern int xmlBufferAdd(xmlBufferPtr buf, const xmlChar * str, int
len);
extern int xmlBufferAddHead(xmlBufferPtr buf, const xmlChar * str,
                            int len);
extern int xmlBufferCCat(xmlBufferPtr buf, const char *str);
extern int xmlBufferCat(xmlBufferPtr buf, const xmlChar * str);
extern const xmlChar *xmlBufferContent(const xmlBufferPtr buf);
extern xmlBufferPtr xmlBufferCreate(void);
extern xmlBufferPtr xmlBufferCreateSize(size_t size);
extern xmlBufferPtr xmlBufferCreateStatic(void *mem, size_t size);
extern int xmlBufferDump(FILE * file, xmlBufferPtr buf);
extern void xmlBufferEmpty(xmlBufferPtr buf);
extern void xmlBufferFree(xmlBufferPtr buf);
extern int xmlBufferGrow(xmlBufferPtr buf, unsigned int len);
extern int xmlBufferLength(const xmlBufferPtr buf);
extern int xmlBufferResize(xmlBufferPtr buf, unsigned int size);
extern void xmlBufferSetAllocationScheme(xmlBufferPtr buf,
                                         xmlBufferAllocationScheme
scheme);
```

```
extern int xmlBufferShrink(xmlBufferPtr buf, unsigned int len);
extern void xmlBufferWriteCHAR(xmlBufferPtr buf, const xmlChar *
string);
extern void xmlBufferWriteChar(xmlBufferPtr buf, const char
*string);
extern void xmlBufferWriteQuotedString(xmlBufferPtr buf,
                                       const xmlChar * string);
extern xmlChar *xmlBuildQName(const xmlChar * ncname,
                              const xmlChar * prefix, xmlChar * memory,
                              int len);
extern xmlDocPtr xmlCopyDoc(xmlDocPtr doc, int recursive);
extern xmlDtdPtr xmlCopyDtd(xmlDtdPtr dtd);
extern xmlNsPtr xmlCopyNamespace(xmlNsPtr cur);
extern xmlNsPtr xmlCopyNamespaceList(xmlNsPtr cur);
extern xmlNodePtr xmlCopyNode(const xmlNodePtr node, int recursive);
extern xmlNodePtr xmlCopyNodeList(const xmlNodePtr node);
extern xmlAttrPtr xmlCopyProp(xmlNodePtr target, xmlAttrPtr cur);
extern xmlAttrPtr xmlCopyPropList(xmlNodePtr target, xmlAttrPtr
cur);
extern xmlDtdPtr xmlCreateIntSubset(xmlDocPtr doc, const xmlChar *
name,
                                   const xmlChar * ExternalID,
                                   const xmlChar * SystemID);
extern void xmlDOMWrapFreeCtxt(xmlDOMWrapCtxtPtr ctxt);
extern xmlDOMWrapCtxtPtr xmlDOMWrapNewCtxt(void);
extern xmlNodePtr xmlDocCopyNode(const xmlNodePtr node, xmlDocPtr
doc,
                                 int recursive);
extern xmlNodePtr xmlDocCopyNodeList(xmlDocPtr doc, const
xmlNodePtr node);
extern int xmlDocDump(FILE * f, xmlDocPtr cur);
extern void xmlDocDumpFormatMemory(xmlDocPtr cur, xmlChar * *mem,
                                   int *size, int format);
extern void xmlDocDumpFormatMemoryEnc(xmlDocPtr out_doc,
                                      xmlChar * *doc_txt_ptr,
                                      int *doc_txt_len,
                                      const char *txt_encoding,
                                      int format);
extern void xmlDocDumpMemory(xmlDocPtr cur, xmlChar * *mem, int
*size);
extern void xmlDocDumpMemoryEnc(xmlDocPtr out_doc, xmlChar *
*doc_txt_ptr,
                                int *doc_txt_len,
                                const char *txt_encoding);
extern int xmlDocFormatDump(FILE * f, xmlDocPtr cur, int format);
extern xmlNodePtr xmlDocGetRootElement(xmlDocPtr doc);
extern xmlNodePtr xmlDocSetRootElement(xmlDocPtr doc, xmlNodePtr
root);
extern void xmlElemDump(FILE * f, xmlDocPtr doc, xmlNodePtr cur);
extern void xmlFreeDoc(xmlDocPtr cur);
extern void xmlFreeDtd(xmlDtdPtr cur);
extern void xmlFreeNode(xmlNodePtr cur);
extern void xmlFreeNodeList(xmlNodePtr cur);
extern void xmlFreeNs(xmlNsPtr cur);
extern void xmlFreeNsList(xmlNsPtr cur);
extern void xmlFreeProp(xmlAttrPtr cur);
extern void xmlFreePropList(xmlAttrPtr cur);
extern                                    xmlBufferAllocationScheme
xmlGetBufferAllocationScheme(void);
extern int xmlGetCompressMode(void);
extern int xmlGetDocCompressMode(xmlDocPtr doc);
extern xmlDtdPtr xmlGetIntSubset(xmlDocPtr doc);
extern xmlNodePtr xmlGetLastChild(xmlNodePtr parent);
extern long int xmlGetLineNo(xmlNodePtr node);
extern xmlChar *xmlGetNoNsProp(xmlNodePtr node, const xmlChar *
name);
```



```
extern xmlChar *xmlGetNodePath(xmlNodePtr node);
extern xmlNsPtr *xmlGetNsList(xmlDocPtr doc, xmlNodePtr node);
extern xmlChar *xmlGetNsProp(xmlNodePtr node, const xmlChar * name,
                             const xmlChar * nameSpace);
extern xmlChar *xmlGetProp(xmlNodePtr node, const xmlChar * name);
extern xmlAttrPtr xmlHasNsProp(xmlNodePtr node, const xmlChar *
name,
                               const xmlChar * nameSpace);
extern xmlAttrPtr xmlHasProp(xmlNodePtr node, const xmlChar * name);
extern int xmlIsBlankNode(xmlNodePtr node);
extern int xmlIsXHTML(const xmlChar * systemID, const xmlChar *
publicID);
extern xmlNodePtr xmlNewCDataBlock(xmlDocPtr doc, const xmlChar *
content,
                                   int len);
extern xmlNodePtr xmlNewCharRef(xmlDocPtr doc, const xmlChar *
name);
extern xmlNodePtr xmlNewChild(xmlNodePtr parent, xmlNsPtr ns,
                              const xmlChar * name,
                              const xmlChar * content);
extern xmlNodePtr xmlNewComment(const xmlChar * content);
extern xmlDocPtr xmlNewDoc(const xmlChar * version);
extern xmlNodePtr xmlNewDocComment(xmlDocPtr doc, const xmlChar *
content);
extern xmlNodePtr xmlNewDocFragment(xmlDocPtr doc);
extern xmlNodePtr xmlNewDocNode(xmlDocPtr doc, xmlNsPtr ns,
                                const xmlChar * name,
                                const xmlChar * content);
extern xmlNodePtr xmlNewDocNodeEatName(xmlDocPtr doc, xmlNsPtr ns,
                                       xmlChar * name,
                                       const xmlChar * content);
extern xmlNodePtr xmlNewDocPI(xmlDocPtr doc, const xmlChar * name,
                              const xmlChar * content);
extern xmlAttrPtr xmlNewDocProp(xmlDocPtr doc, const xmlChar * name,
                                const xmlChar * value);
extern xmlNodePtr xmlNewDocRawNode(xmlDocPtr doc, xmlNsPtr ns,
                                   const xmlChar * name,
                                   const xmlChar * content);
extern xmlNodePtr xmlNewDocText(xmlDocPtr doc, const xmlChar *
content);
extern xmlNodePtr xmlNewDocTextLen(xmlDocPtr doc, const xmlChar *
content,
                                   int len);
extern xmlDtdPtr xmlNewDtd(xmlDocPtr doc, const xmlChar * name,
                           const xmlChar * ExternalID,
                           const xmlChar * SystemID);
extern xmlNodePtr xmlNewNode(xmlNsPtr ns, const xmlChar * name);
extern xmlNodePtr xmlNewNodeEatName(xmlNsPtr ns, xmlChar * name);
extern xmlNsPtr xmlNewNs(xmlNodePtr node, const xmlChar * href,
                         const xmlChar * prefix);
extern xmlAttrPtr xmlNewNsProp(xmlNodePtr node, xmlNsPtr ns,
                               const xmlChar * name,
                               const xmlChar * value);
extern xmlAttrPtr xmlNewNsPropEatName(xmlNodePtr node, xmlNsPtr ns,
                                      xmlChar * name,
                                      const xmlChar * value);
extern xmlNodePtr xmlNewPI(const xmlChar * name, const xmlChar *
content);
extern xmlAttrPtr xmlNewProp(xmlNodePtr node, const xmlChar * name,
                             const xmlChar * value);
extern xmlNodePtr xmlNewReference(xmlDocPtr doc, const xmlChar *
name);
extern xmlNodePtr xmlNewText(const xmlChar * content);
extern xmlNodePtr xmlNewTextChild(xmlNodePtr parent, xmlNsPtr ns,
                                  const xmlChar * name,
                                  const xmlChar * content);
```



```
extern xmlNodePtr xmlNewTextLen(const xmlChar * content, int len);
extern void xmlNodeAddContent(xmlNodePtr cur, const xmlChar *
content);
extern void xmlNodeAddContentLen(xmlNodePtr cur, const xmlChar *
content,
                                 int len);
extern int xmlNodeBufGetContent(xmlBufferPtr buffer, xmlNodePtr
cur);
extern int xmlNodeDump(xmlBufferPtr buf, xmlDocPtr doc, xmlNodePtr
cur,
                       int level, int format);
extern void xmlNodeDumpOutput(xmlOutputBufferPtr buf, xmlDocPtr doc,
                              xmlNodePtr cur, int level, int format,
                              const char *encoding);
extern xmlChar *xmlNodeGetBase(xmlDocPtr doc, xmlNodePtr cur);
extern xmlChar *xmlNodeGetContent(xmlNodePtr cur);
extern xmlChar *xmlNodeGetLang(xmlNodePtr cur);
extern int xmlNodeGetSpacePreserve(xmlNodePtr cur);
extern int xmlNodeIsText(xmlNodePtr node);
extern xmlChar *xmlNodeListGetRawString(xmlDocPtr doc, xmlNodePtr
list,
                                        int inLine);
extern xmlChar *xmlNodeListGetString(xmlDocPtr doc, xmlNodePtr list,
                                     int inLine);
extern void xmlNodeSetBase(xmlNodePtr cur, const xmlChar * uri);
extern void xmlNodeSetContent(xmlNodePtr cur, const xmlChar *
content);
extern void xmlNodeSetContentLen(xmlNodePtr cur, const xmlChar *
content,
                                 int len);
extern void xmlNodeSetLang(xmlNodePtr cur, const xmlChar * lang);
extern void xmlNodeSetName(xmlNodePtr cur, const xmlChar * name);
extern void xmlNodeSetSpacePreserve(xmlNodePtr cur, int val);
extern int xmlReconciliateNs(xmlDocPtr doc, xmlNodePtr tree);
extern int xmlRemoveProp(xmlAttrPtr cur);
extern xmlNodePtr xmlReplaceNode(xmlNodePtr old, xmlNodePtr cur);
extern int xmlSaveFile(const char *filename, xmlDocPtr cur);
extern int xmlSaveFileEnc(const char *filename, xmlDocPtr cur,
                          const char *encoding);
extern int xmlSaveFileTo(xmlOutputBufferPtr buf, xmlDocPtr cur,
                         const char *encoding);
extern int xmlSaveFormatFile(const char *filename, xmlDocPtr cur,
                             int format);
extern int xmlSaveFormatFileEnc(const char *filename, xmlDocPtr cur,
                                const char *encoding, int format);
extern int xmlSaveFormatFileTo(xmlOutputBufferPtr buf, xmlDocPtr
cur,
                               const char *encoding, int format);
extern xmlNsPtr xmlSearchNs(xmlDocPtr doc, xmlNodePtr node,
                            const xmlChar * nameSpace);
extern xmlNsPtr xmlSearchNsByHref(xmlDocPtr doc, xmlNodePtr node,
                                  const xmlChar * href);
extern void xmlSetBufferAllocationScheme(xmlBufferAllocationScheme
scheme);
extern void xmlSetCompressMode(int mode);
extern void xmlSetDocCompressMode(xmlDocPtr doc, int mode);
extern void xmlSetListDoc(xmlNodePtr list, xmlDocPtr doc);
extern void xmlSetNs(xmlNodePtr node, xmlNsPtr ns);
extern xmlAttrPtr xmlSetNsProp(xmlNodePtr node, xmlNsPtr ns,
                               const xmlChar * name,
                               const xmlChar * value);
extern xmlAttrPtr xmlSetProp(xmlNodePtr node, const xmlChar * name,
                             const xmlChar * value);
extern void xmlSetTreeDoc(xmlNodePtr tree, xmlDocPtr doc);
extern xmlChar *xmlSplitQName2(const xmlChar * name, xmlChar *
*prefix);
```



```
extern const xmlChar *xmlSplitQName3(const xmlChar * name, int
*len);
extern xmlNodePtr xmlStringGetNodeList(xmlDocPtr doc,
                                       const xmlChar * value);
extern xmlNodePtr xmlStringLenGetNodeList(xmlDocPtr doc,
                                          const xmlChar * value, int len);
extern int xmlTextConcat(xmlNodePtr node, const xmlChar * content,
                         int len);
extern xmlNodePtr xmlTextMerge(xmlNodePtr first, xmlNodePtr second);
extern void xmlUnlinkNode(xmlNodePtr cur);
extern int xmlUnsetNsProp(xmlNodePtr node, xmlNsPtr ns,
                          const xmlChar * name);
extern int xmlUnsetProp(xmlNodePtr node, const xmlChar * name);
extern int xmlValidateNCName(const xmlChar * value, int space);
extern int xmlValidateNMToken(const xmlChar * value, int space);
extern int xmlValidateName(const xmlChar * value, int space);
extern int xmlValidateQName(const xmlChar * value, int space);
```

8.2.20 libxml2/libxml/uri.h

```
typedef struct _xmlURI {
    char *scheme;
    char *opaque;
    char *authority;
    char *server;
    char *user;
    int port;
    char *path;
    char *query;
    char *fragment;
    int cleanup;
} xmlURI;
typedef xmlURI *xmlURIPtr;
extern xmlChar *xmlBuildRelativeURI(const xmlChar * URI,
                                    const xmlChar * base);
extern xmlChar *xmlBuildURI(const xmlChar * URI, const xmlChar *
base);
extern xmlChar *xmlCanonicPath(const xmlChar * path);
extern xmlURIPtr xmlCreateURI(void);
extern void xmlFreeURI(xmlURIPtr uri);
extern int xmlNormalizeURIPath(char *path);
extern xmlURIPtr xmlParseURI(const char *str);
extern xmlURIPtr xmlParseURIRaw(const char *str, int raw);
extern int xmlParseURIReference(xmlURIPtr uri, const char *str);
extern void xmlPrintURI(FILE * stream, xmlURIPtr uri);
extern xmlChar *xmlSaveUri(xmlURIPtr uri);
extern xmlChar *xmlURIEscape(const xmlChar * str);
extern xmlChar *xmlURIEscapeStr(const xmlChar * str, const xmlChar
* list);
extern char *xmlURIUnescapeString(const char *str, int len, char
*target);
```

8.2.21 libxml2/libxml/valid.h

```
typedef xmlValidCtxt *xmlValidCtxtPtr;
typedef struct _xmlHashTable xmlAttributeTable;
typedef xmlAttributeTable *xmlAttributeTablePtr;
typedef enum {
    XML_ELEMENT_TYPE_UNDEFINED = 0,
    XML_ELEMENT_TYPE_EMPTY = 1,
    XML_ELEMENT_TYPE_ANY = 2,
    XML_ELEMENT_TYPE_MIXED = 3,
    XML_ELEMENT_TYPE_ELEMENT = 4
```

```
} xmlElementTypeVal;
typedef enum {
    XML_ATTRIBUTE_NONE = 1,
    XML_ATTRIBUTE_REQUIRED = 2,
    XML_ATTRIBUTE_IMPLIED = 3,
    XML_ATTRIBUTE_FIXED = 4
} xmlAttributeDefault;
typedef struct _xmlAttribute {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlDtd *parent;
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    struct _xmlAttribute *nexth;
    xmlAttributeType atype;
    xmlAttributeDefault def;
    const xmlChar *defaultValue;
    xmlEnumerationPtr tree;
    const xmlChar *prefix;
    const xmlChar *elem;
} xmlAttribute;
typedef xmlAttribute *xmlAttributePtr;
typedef struct _xmlElement {
    void *_private;
    xmlElementType type;
    const xmlChar *name;
    struct _xmlNode *children;
    struct _xmlNode *last;
    struct _xmlDtd *parent;
    struct _xmlNode *next;
    struct _xmlNode *prev;
    struct _xmlDoc *doc;
    xmlElementTypeVal etype;
    xmlElementContentPtr content;
    xmlAttributePtr attributes;
    const xmlChar *prefix;
    xmlRegexpPtr contModel;
} xmlElement;
typedef xmlElement *xmlElementPtr;
typedef struct _xmlHashTable xmlNotationTable;
typedef xmlNotationTable *xmlNotationTablePtr;
typedef struct _xmlNotation {
    const xmlChar *name;
    const xmlChar *PublicID;
    const xmlChar *SystemID;
} xmlNotation;
typedef xmlNotation *xmlNotationPtr;
typedef struct _xmlID {
    struct _xmlID *next;
    const xmlChar *value;
    xmlAttrPtr attr;
    const xmlChar *name;
    int lineno;
    struct _xmlDoc *doc;
} xmlID;
typedef xmlID *xmlIDPtr;
typedef struct _xmlRef {
    struct _xmlRef *next;
    const xmlChar *value;
    xmlAttrPtr attr;
    const xmlChar *name;
    int lineno;
```



```
} xmlRef;
typedef xmlRef *xmlRefPtr;
typedef struct _xmlHashTable xmlElementTable;
typedef xmlElementTable *xmlElementTablePtr;
typedef struct _xmlHashTable xmlIDTable;
typedef xmlIDTable *xmlIDTablePtr;
typedef struct _xmlHashTable xmlRefTable;
typedef xmlRefTable *xmlRefTablePtr;
extern     xmlAttributePtr     xmlAddAttributeDecl(xmlValidCtxtPtr,
xmlDtdPtr,
                                       const xmlChar *,
                                       const xmlChar *,
                                       const xmlChar *,
                                       xmlAttributeType,
                                       xmlAttributeDefault,
                                       const xmlChar *,
                                       xmlEnumerationPtr);
extern xmlElementPtr xmlAddElementDecl(xmlValidCtxtPtr, xmlDtdPtr,
                                  const xmlChar *, xmlElementTypeVal,
                                  xmlElementContentPtr);
extern xmlIDPtr xmlAddID(xmlValidCtxtPtr, xmlDocPtr, const xmlChar
*,
                     xmlAttrPtr);
extern      xmlNotationPtr      xmlAddNotationDecl(xmlValidCtxtPtr,
xmlDtdPtr,
                                    const xmlChar *, const xmlChar *,
                                    const xmlChar *);
extern   xmlRefPtr   xmlAddRef(xmlValidCtxtPtr,   xmlDocPtr,   const
xmlChar *,
                       xmlAttrPtr);
extern                                          xmlAttributeTablePtr
xmlCopyAttributeTable(xmlAttributeTablePtr);
extern xmlElementContentPtr xmlCopyDocElementContent(xmlDocPtr,
                                                xmlElementContentPtr);
extern xmlElementTablePtr xmlCopyElementTable(xmlElementTablePtr);
extern xmlEnumerationPtr xmlCopyEnumeration(xmlEnumerationPtr);
extern                                           xmlNotationTablePtr
xmlCopyNotationTable(xmlNotationTablePtr);
extern xmlEnumerationPtr xmlCreateEnumeration(const xmlChar *);
extern void xmlDumpAttributeDecl(xmlBufferPtr, xmlAttributePtr);
extern          void          xmlDumpAttributeTable(xmlBufferPtr,
xmlAttributeTablePtr);
extern void xmlDumpElementDecl(xmlBufferPtr, xmlElementPtr);
extern void xmlDumpElementTable(xmlBufferPtr, xmlElementTablePtr);
extern void xmlDumpNotationDecl(xmlBufferPtr, xmlNotationPtr);
extern          void           xmlDumpNotationTable(xmlBufferPtr,
xmlNotationTablePtr);
extern void xmlFreeAttributeTable(xmlAttributeTablePtr);
extern          void          xmlFreeDocElementContent(xmlDocPtr,
xmlElementContentPtr);
extern void xmlFreeElementTable(xmlElementTablePtr);
extern void xmlFreeEnumeration(xmlEnumerationPtr);
extern void xmlFreeIDTable(xmlIDTablePtr);
extern void xmlFreeNotationTable(xmlNotationTablePtr);
extern void xmlFreeRefTable(xmlRefTablePtr);
extern void xmlFreeValidCtxt(xmlValidCtxtPtr);
extern xmlAttributePtr xmlGetDtdAttrDesc(xmlDtdPtr, const xmlChar
*,
                                    const xmlChar *);
extern xmlElementPtr xmlGetDtdElementDesc(xmlDtdPtr, const xmlChar
*);
extern   xmlNotationPtr   xmlGetDtdNotationDesc(xmlDtdPtr,   const
xmlChar *);
extern xmlAttributePtr xmlGetDtdQAttrDesc(xmlDtdPtr, const xmlChar
*,
                                     const xmlChar *,
```

```
                                        const xmlChar *);
extern   xmlElementPtr   xmlGetDtdQElementDesc(xmlDtdPtr,   const
xmlChar *,
                                        const xmlChar *);
extern xmlAttrPtr xmlGetID(xmlDocPtr, const xmlChar *);
extern xmlListPtr xmlGetRefs(xmlDocPtr, const xmlChar *);
extern int xmlIsID(xmlDocPtr, xmlNodePtr, xmlAttrPtr);
extern int xmlIsMixedElement(xmlDocPtr, const xmlChar *);
extern int xmlIsRef(xmlDocPtr, xmlNodePtr, xmlAttrPtr);
extern xmlElementContentPtr xmlNewDocElementContent(xmlDocPtr,
                                                    const xmlChar *,
                                                    xmlElementContentType);
extern xmlValidCtxtPtr xmlNewValidCtxt(void);
extern int xmlRemoveID(xmlDocPtr, xmlAttrPtr);
extern int xmlRemoveRef(xmlDocPtr, xmlAttrPtr);
extern     void     xmlSnprintfElementContent(char     *,     int,
xmlElementContentPtr,
                                   int);
extern       int       xmlValidBuildContentModel(xmlValidCtxtPtr,
xmlElementPtr);
extern                                                        xmlChar
*xmlValidCtxtNormalizeAttributeValue(xmlValidCtxtPtr,
                                                    xmlDocPtr, xmlNodePtr,
                                                    const xmlChar *,
                                                    const xmlChar *);
extern int xmlValidGetPotentialChildren(xmlElementContent *,
                                        const xmlChar * *, int *, int);
extern int xmlValidGetValidElements(xmlNode *, xmlNode *,
                                    const xmlChar * *, int);
extern     xmlChar     *xmlValidNormalizeAttributeValue(xmlDocPtr,
xmlNodePtr,
                                                    const xmlChar *,
                                                    const xmlChar *);
extern int xmlValidateAttributeDecl(xmlValidCtxtPtr, xmlDocPtr,
                                    xmlAttributePtr);
extern   int   xmlValidateAttributeValue(xmlAttributeType,   const
xmlChar *);
extern int xmlValidateDocument(xmlValidCtxtPtr, xmlDocPtr);
extern int xmlValidateDocumentFinal(xmlValidCtxtPtr, xmlDocPtr);
extern int xmlValidateDtd(xmlValidCtxtPtr, xmlDocPtr, xmlDtdPtr);
extern int xmlValidateDtdFinal(xmlValidCtxtPtr, xmlDocPtr);
extern    int    xmlValidateElement(xmlValidCtxtPtr,    xmlDocPtr,
xmlNodePtr);
extern int xmlValidateElementDecl(xmlValidCtxtPtr, xmlDocPtr,
                                  xmlElementPtr);
extern int xmlValidateNameValue(const xmlChar *);
extern int xmlValidateNamesValue(const xmlChar *);
extern int xmlValidateNmtokenValue(const xmlChar *);
extern int xmlValidateNmtokensValue(const xmlChar *);
extern int xmlValidateNotationDecl(xmlValidCtxtPtr, xmlDocPtr,
                                   xmlNotationPtr);
extern int xmlValidateNotationUse(xmlValidCtxtPtr, xmlDocPtr,
                                  const xmlChar *);
extern  int  xmlValidateOneAttribute(xmlValidCtxtPtr,  xmlDocPtr,
xmlNodePtr,
                                   xmlAttrPtr, const xmlChar *);
extern   int   xmlValidateOneElement(xmlValidCtxtPtr,   xmlDocPtr,
xmlNodePtr);
extern  int  xmlValidateOneNamespace(xmlValidCtxtPtr,  xmlDocPtr,
xmlNodePtr,
                                   const xmlChar *, xmlNsPtr,
                                   const xmlChar *);
extern   int   xmlValidatePopElement(xmlValidCtxtPtr,   xmlDocPtr,
xmlNodePtr,
                                 const xmlChar *);
```



```
extern int xmlValidatePushCData(xmlValidCtxtPtr, const xmlChar *,
int);
extern   int   xmlValidatePushElement(xmlValidCtxtPtr,   xmlDocPtr,
xmlNodePtr,
                                const xmlChar *);
extern int xmlValidateRoot(xmlValidCtxtPtr, xmlDocPtr);
```

8.2.22 libxml2/libxml/xinclude.h

```
#define XINCLUDE_PARSE_ENCODING (const xmlChar *) "encoding"
#define XINCLUDE_FALLBACK      (const xmlChar *) "fallback"
#define XINCLUDE_HREF   (const xmlChar *) "href"
#define       XINCLUDE_OLD_NS       (const       xmlChar       *)
"http://www.w3.org/2001/XInclude"
#define    XINCLUDE_NS                      (const    xmlChar    *)
"http://www.w3.org/2003/XInclude"
#define XINCLUDE_NODE   (const xmlChar *) "include"
#define XINCLUDE_PARSE  (const xmlChar *) "parse"
#define XINCLUDE_PARSE_TEXT    (const xmlChar *) "text"
#define XINCLUDE_PARSE_XML     (const xmlChar *) "xml"
#define XINCLUDE_PARSE_XPOINTER (const xmlChar *) "xpointer"

typedef struct _xmlXIncludeCtxt xmlXIncludeCtxt;
typedef xmlXIncludeCtxt *xmlXIncludeCtxtPtr;
extern void xmlXIncludeFreeContext(xmlXIncludeCtxtPtr ctxt);
extern xmlXIncludeCtxtPtr xmlXIncludeNewContext(xmlDocPtr doc);
extern int xmlXIncludeProcess(xmlDocPtr doc);
extern int xmlXIncludeProcessFlags(xmlDocPtr doc, int flags);
extern int xmlXIncludeProcessNode(xmlXIncludeCtxtPtr ctxt,
                                xmlNodePtr tree);
extern int xmlXIncludeProcessTree(xmlNodePtr tree);
extern int xmlXIncludeProcessTreeFlags(xmlNodePtr tree, int flags);
extern int xmlXIncludeSetFlags(xmlXIncludeCtxtPtr ctxt, int flags);
```

8.2.23 libxml2/libxml/xmlIO.h

```
typedef int (*xmlOutputMatchCallback) (const char *);
typedef void *(*xmlOutputOpenCallback) (const char *);
typedef struct _xmlParserNodeInfo {
    const struct _xmlNode *node;
    long unsigned int begin_pos;
    long unsigned int begin_line;
    long unsigned int end_pos;
    long unsigned int end_line;
} xmlParserNodeInfo;
typedef struct _xmlParserNodeInfoSeq {
    long unsigned int maximum;
    long unsigned int length;
    xmlParserNodeInfo *buffer;
} xmlParserNodeInfoSeq;
typedef void (*xmlValidityErrorFunc) (void *, const char *, ...);
typedef void (*xmlValidityWarningFunc) (void *, const char *, ...);
typedef struct _xmlValidState xmlValidState;
typedef struct _xmlValidCtxt {
    void *userData;
    xmlValidityErrorFunc error;
    xmlValidityWarningFunc warning;
    xmlNodePtr node;
    int nodeNr;
    int nodeMax;
    xmlNodePtr *nodeTab;
    unsigned int finishDtd;
    xmlDocPtr doc;
```

```
    int valid;
    xmlValidState *vstate;
    int vstateNr;
    int vstateMax;
    xmlValidState *vstateTab;
    xmlAutomataPtr am;
    xmlAutomataStatePtr state;
} xmlValidCtxt;
typedef enum {
    XML_PARSER_EOF = -1,
    XML_PARSER_START = 0,
    XML_PARSER_MISC = 1,
    XML_PARSER_PI = 2,
    XML_PARSER_DTD = 3,
    XML_PARSER_PROLOG = 4,
    XML_PARSER_COMMENT = 5,
    XML_PARSER_START_TAG = 6,
    XML_PARSER_CONTENT = 7,
    XML_PARSER_CDATA_SECTION = 8,
    XML_PARSER_END_TAG = 9,
    XML_PARSER_ENTITY_DECL = 10,
    XML_PARSER_ENTITY_VALUE = 11,
    XML_PARSER_ATTRIBUTE_VALUE = 12,
    XML_PARSER_SYSTEM_LITERAL = 13,
    XML_PARSER_EPILOG = 14,
    XML_PARSER_IGNORE = 15,
    XML_PARSER_PUBLIC_LITERAL = 16
} xmlParserInputState;
typedef enum {
    XML_PARSE_UNKNOWN = 0,
    XML_PARSE_DOM = 1,
    XML_PARSE_SAX = 2,
    XML_PARSE_PUSH_DOM = 3,
    XML_PARSE_PUSH_SAX = 4,
    XML_PARSE_READER = 5
} xmlParserMode;
typedef struct _xmlParserCtxt {
    struct _xmlSAXHandler *sax;
    void *userData;
    xmlDocPtr myDoc;
    int wellFormed;
    int replaceEntities;
    const xmlChar *version;
    const xmlChar *encoding;
    int standalone;
    int html;
    xmlParserInputPtr input;
    int inputNr;
    int inputMax;
    xmlParserInputPtr *inputTab;
    xmlNodePtr node;
    int nodeNr;
    int nodeMax;
    xmlNodePtr *nodeTab;
    int record_info;
    xmlParserNodeInfoSeq node_seq;
    int errNo;
    int hasExternalSubset;
    int hasPErefs;
    int external;
    int valid;
    int validate;
    xmlValidCtxt vctxt;
    xmlParserInputState instate;
    int token;
    char *directory;
```



```
    const xmlChar *name;
    int nameNr;
    int nameMax;
    const xmlChar **nameTab;
    long int nbChars;
    long int checkIndex;
    int keepBlanks;
    int disableSAX;
    int inSubset;
    const xmlChar *intSubName;
    xmlChar *extSubURI;
    xmlChar *extSubSystem;
    int *space;
    int spaceNr;
    int spaceMax;
    int *spaceTab;
    int depth;
    xmlParserInputPtr entity;
    int charset;
    int nodelen;
    int nodemem;
    int pedantic;
    void *_private;
    int loadsubset;
    int linenumbers;
    void *catalogs;
    int recovery;
    int progressive;
    xmlDictPtr dict;
    const xmlChar **atts;
    int maxatts;
    int docdict;
    const xmlChar *str_xml;
    const xmlChar *str_xmlns;
    const xmlChar *str_xml_ns;
    int sax2;
    int nsNr;
    int nsMax;
    const xmlChar **nsTab;
    int *attallocs;
    void **pushTab;
    xmlHashTablePtr attsDefault;
    xmlHashTablePtr attsSpecial;
    int nsWellFormed;
    int options;
    int dictNames;
    int freeElemsNr;
    xmlNodePtr freeElems;
    int freeAttrsNr;
    xmlAttrPtr freeAttrs;
    xmlError lastError;
    xmlParserMode parseMode;
} xmlParserCtxt;
typedef xmlParserCtxt *xmlParserCtxtPtr;
typedef int (*xmlInputMatchCallback) (const char *);
typedef void *(*xmlInputOpenCallback) (const char *);
extern                                          xmlOutputBufferPtr
xmlAllocOutputBuffer(xmlCharEncodingHandlerPtr
                                        encoder);
extern                                     xmlParserInputBufferPtr
xmlAllocParserInputBuffer(xmlCharEncoding
                                                enc);
extern int xmlCheckFilename(const char *path);
extern xmlParserInputPtr xmlCheckHTTPInput(xmlParserCtxtPtr ctxt,
                                    xmlParserInputPtr ret);
extern void xmlCleanupInputCallbacks(void);
```

```
extern void xmlCleanupOutputCallbacks(void);
extern int xmlFileClose(void *context);
extern int xmlFileMatch(const char *filename);
extern void *xmlFileOpen(const char *filename);
extern int xmlFileRead(void *context, char *buffer, int len);
extern void xmlFreeParserInputBuffer(xmlParserInputBufferPtr in);
extern int xmlIOFTPClose(void *context);
extern int xmlIOFTPMatch(const char *filename);
extern void *xmlIOFTPOpen(const char *filename);
extern int xmlIOFTPRead(void *context, char *buffer, int len);
extern int xmlIOHTTPClose(void *context);
extern int xmlIOHTTPMatch(const char *filename);
extern void *xmlIOHTTPOpen(const char *filename);
extern void *xmlIOHTTPOpenW(const char *post_uri, int compression);
extern int xmlIOHTTPRead(void *context, char *buffer, int len);
extern  xmlParserInputPtr  xmlNoNetExternalEntityLoader(const  char
*URL,
                                            const char *ID,
                                            xmlParserCtxtPtr
                                            ctxt);
extern xmlChar *xmlNormalizeWindowsPath(const xmlChar * path);
extern int xmlOutputBufferClose(xmlOutputBufferPtr out);
extern xmlOutputBufferPtr xmlOutputBufferCreateFd(int fd,

xmlCharEncodingHandlerPtr
                                           encoder);
extern xmlOutputBufferPtr xmlOutputBufferCreateFile(FILE * file,

xmlCharEncodingHandlerPtr
                                            encoder);
extern xmlOutputBufferPtr xmlOutputBufferCreateFilename(const char
*URI,

xmlCharEncodingHandlerPtr
                                              encoder,
                                              int compression);
extern                                         xmlOutputBufferPtr
xmlOutputBufferCreateIO(xmlOutputWriteCallback
                                           iowrite,
                                           xmlOutputCloseCallback
                                           ioclose, void *ioctx,

xmlCharEncodingHandlerPtr
                                           encoder);
extern int xmlOutputBufferFlush(xmlOutputBufferPtr out);
extern int xmlOutputBufferWrite(xmlOutputBufferPtr out, int len,
                                const char *buf);
extern int xmlOutputBufferWriteEscape(xmlOutputBufferPtr out,
                                      const xmlChar * str,
                                      xmlCharEncodingOutputFunc
escaping);
extern int xmlOutputBufferWriteString(xmlOutputBufferPtr out,
                                      const char *str);
extern char *xmlParserGetDirectory(const char *filename);
extern xmlParserInputBufferPtr xmlParserInputBufferCreateFd(int fd,
                                                    xmlCharEncoding
                                                    enc);
extern xmlParserInputBufferPtr xmlParserInputBufferCreateFile(FILE
* file,

xmlCharEncoding
                                                      enc);
extern xmlParserInputBufferPtr
xmlParserInputBufferCreateIO(xmlInputReadCallback ioread,
                             xmlInputCloseCallback ioclose, void *ioctx,
                             xmlCharEncoding enc);
```



```
extern xmlParserInputBufferPtr xmlParserInputBufferCreateMem(const
char
                                                    *mem,
                                                    int size,

xmlCharEncoding
                                                    enc);
extern                                          xmlParserInputBufferPtr
xmlParserInputBufferCreateStatic(const char
                                                      *mem,
                                                      int size,

xmlCharEncoding
                                                      enc);
extern  int  xmlParserInputBufferGrow(xmlParserInputBufferPtr  in,
int len);
extern  int  xmlParserInputBufferPush(xmlParserInputBufferPtr  in,
int len,
                                const char *buf);
extern  int  xmlParserInputBufferRead(xmlParserInputBufferPtr  in,
int len);
extern int xmlPopInputCallbacks(void);
extern void xmlRegisterDefaultInputCallbacks(void);
extern void xmlRegisterDefaultOutputCallbacks(void);
extern void xmlRegisterHTTPPostCallbacks(void);
extern     int     xmlRegisterInputCallbacks(xmlInputMatchCallback
matchFunc,
                                xmlInputOpenCallback openFunc,
                                xmlInputReadCallback readFunc,
                                xmlInputCloseCallback closeFunc);
extern    int    xmlRegisterOutputCallbacks(xmlOutputMatchCallback
matchFunc,
                                xmlOutputOpenCallback openFunc,
                                xmlOutputWriteCallback writeFunc,
                                xmlOutputCloseCallback closeFunc);
```

8.2.24 libxml2/libxml/xmlautomata.h

```
typedef struct _xmlAutomataState xmlAutomataState;
typedef xmlAutomataState *xmlAutomataStatePtr;
typedef struct _xmlAutomata xmlAutomata;
typedef xmlAutomata *xmlAutomataPtr;
extern xmlRegexpPtr xmlAutomataCompile(xmlAutomataPtr);
extern xmlAutomataStatePtr xmlAutomataGetInitState(xmlAutomataPtr);
extern int xmlAutomataIsDeterminist(xmlAutomataPtr);
extern xmlAutomataStatePtr xmlAutomataNewAllTrans(xmlAutomataPtr,
                                          xmlAutomataStatePtr,
                                          xmlAutomataStatePtr,
                                          int);
extern xmlAutomataStatePtr xmlAutomataNewCountTrans(xmlAutomataPtr,
                                           xmlAutomataStatePtr,
                                           xmlAutomataStatePtr,
                                           const xmlChar *, int,
                                           int, void *);
extern                                        xmlAutomataStatePtr
xmlAutomataNewCountTrans2(xmlAutomataPtr,
                                            xmlAutomataStatePtr,
                                            xmlAutomataStatePtr,
                                            const xmlChar *,
                                            const xmlChar *, int,
                                            int, void *);
extern                                           xmlAutomataStatePtr
xmlAutomataNewCountedTrans(xmlAutomataPtr,
                                             xmlAutomataStatePtr,
                                             xmlAutomataStatePtr,
```

```
                                                int);
extern int xmlAutomataNewCounter(xmlAutomataPtr, int, int);
extern                                            xmlAutomataStatePtr
xmlAutomataNewCounterTrans(xmlAutomataPtr,
                                                xmlAutomataStatePtr,
                                                xmlAutomataStatePtr,
                                                int);
extern xmlAutomataStatePtr xmlAutomataNewEpsilon(xmlAutomataPtr,
                                            xmlAutomataStatePtr,
                                            xmlAutomataStatePtr);
extern xmlAutomataStatePtr xmlAutomataNewNegTrans(xmlAutomataPtr,
                                             xmlAutomataStatePtr,
                                             xmlAutomataStatePtr,
                                             const xmlChar *,
                                             const xmlChar *, void *);
extern xmlAutomataStatePtr xmlAutomataNewOnceTrans(xmlAutomataPtr,
                                              xmlAutomataStatePtr,
                                              xmlAutomataStatePtr,
                                              const xmlChar *, int,
                                              int, void *);
extern xmlAutomataStatePtr xmlAutomataNewOnceTrans2(xmlAutomataPtr,
                                               xmlAutomataStatePtr,
                                               xmlAutomataStatePtr,
                                               const xmlChar *,
                                               const xmlChar *, int,
                                               int, void *);
extern xmlAutomataStatePtr xmlAutomataNewState(xmlAutomataPtr);
extern xmlAutomataStatePtr xmlAutomataNewTransition(xmlAutomataPtr,
                                               xmlAutomataStatePtr,
                                               xmlAutomataStatePtr,
                                               const xmlChar *,
                                               void *);
extern                                             xmlAutomataStatePtr
xmlAutomataNewTransition2(xmlAutomataPtr,
                                                xmlAutomataStatePtr,
                                                xmlAutomataStatePtr,
                                                const xmlChar *,
                                                const xmlChar *,
                                                void *);
extern        int        xmlAutomataSetFinalState(xmlAutomataPtr,
xmlAutomataStatePtr);
extern void xmlFreeAutomata(xmlAutomataPtr);
extern xmlAutomataPtr xmlNewAutomata(void);
```

8.2.25 libxml2/libxml/xmlerror.h

```
typedef int (*xmlInputReadCallback) (void *, char *, int);
typedef int (*xmlInputCloseCallback) (void *);
typedef struct _xmlParserInputBuffer {
    void *context;
    xmlInputReadCallback readcallback;
    xmlInputCloseCallback closecallback;
    xmlCharEncodingHandlerPtr encoder;
    xmlBufferPtr buffer;
    xmlBufferPtr raw;
    int compressed;
    int error;
    long unsigned int rawconsumed;
} xmlParserInputBuffer;
typedef xmlParserInputBuffer *xmlParserInputBufferPtr;
typedef void (*xmlParserInputDeallocate) (xmlChar *);
typedef struct _xmlParserInput {
    xmlParserInputBufferPtr buf;
    const char *filename;
    const char *directory;
```



```
    const xmlChar *base;
    const xmlChar *cur;
    const xmlChar *end;
    int length;
    int line;
    int col;
    long unsigned int consumed;
    xmlParserInputDeallocate free;
    const xmlChar *encoding;
    const xmlChar *version;
    int standalone;
    int id;
} xmlParserInput;
typedef xmlParserInput *xmlParserInputPtr;
typedef void (*xmlGenericErrorFunc) (void *, const char *, ...);
typedef enum {
    XML_ERR_NONE = 0,
    XML_ERR_WARNING = 1,
    XML_ERR_ERROR = 2,
    XML_ERR_FATAL = 3
} xmlErrorLevel;
typedef struct _xmlError {
    int domain;
    int code;
    char *message;
    xmlErrorLevel level;
    char *file;
    int line;
    char *str1;
    char *str2;
    char *str3;
    int int1;
    int int2;
    void *ctxt;
    void *node;
} xmlError;
typedef xmlError *xmlErrorPtr;
typedef void (*xmlStructuredErrorFunc) (void *, xmlErrorPtr);
typedef enum {
    XML_FROM_NONE = 0,
    XML_FROM_PARSER,
    XML_FROM_TREE,
    XML_FROM_NAMESPACE,
    XML_FROM_DTD,
    XML_FROM_HTML,
    XML_FROM_MEMORY,
    XML_FROM_OUTPUT,
    XML_FROM_IO,
    XML_FROM_FTP,
    XML_FROM_HTTP,
    XML_FROM_XINCLUDE,
    XML_FROM_XPATH,
    XML_FROM_XPOINTER,
    XML_FROM_REGEXP,
    XML_FROM_DATATYPE,
    XML_FROM_SCHEMASP,
    XML_FROM_SCHEMASV,
    XML_FROM_RELAXNGP,
    XML_FROM_RELAXNGV,
    XML_FROM_CATALOG,
    XML_FROM_C14N,
    XML_FROM_XSLT,
    XML_FROM_VALID,
    XML_FROM_CHECK,
    XML_FROM_WRITER,
    XML_FROM_MODULE,
```

```
    XML_FROM_I18N
} xmlErrorDomain;
typedef enum {
    XML_ERR_OK = 0,
    XML_ERR_INTERNAL_ERROR,
    XML_ERR_NO_MEMORY,
    XML_ERR_DOCUMENT_START,
    XML_ERR_DOCUMENT_EMPTY,
    XML_ERR_DOCUMENT_END,
    XML_ERR_INVALID_HEX_CHARREF,
    XML_ERR_INVALID_DEC_CHARREF,
    XML_ERR_INVALID_CHARREF,
    XML_ERR_INVALID_CHAR,
    XML_ERR_CHARREF_AT_EOF,
    XML_ERR_CHARREF_IN_PROLOG,
    XML_ERR_CHARREF_IN_EPILOG,
    XML_ERR_CHARREF_IN_DTD,
    XML_ERR_ENTITYREF_AT_EOF,
    XML_ERR_ENTITYREF_IN_PROLOG,
    XML_ERR_ENTITYREF_IN_EPILOG,
    XML_ERR_ENTITYREF_IN_DTD,
    XML_ERR_PEREF_AT_EOF,
    XML_ERR_PEREF_IN_PROLOG,
    XML_ERR_PEREF_IN_EPILOG,
    XML_ERR_PEREF_IN_INT_SUBSET,
    XML_ERR_ENTITYREF_NO_NAME,
    XML_ERR_ENTITYREF_SEMICOL_MISSING,
    XML_ERR_PEREF_NO_NAME,
    XML_ERR_PEREF_SEMICOL_MISSING,
    XML_ERR_UNDECLARED_ENTITY,
    XML_WAR_UNDECLARED_ENTITY,
    XML_ERR_UNPARSED_ENTITY,
    XML_ERR_ENTITY_IS_EXTERNAL,
    XML_ERR_ENTITY_IS_PARAMETER,
    XML_ERR_UNKNOWN_ENCODING,
    XML_ERR_UNSUPPORTED_ENCODING,
    XML_ERR_STRING_NOT_STARTED,
    XML_ERR_STRING_NOT_CLOSED,
    XML_ERR_NS_DECL_ERROR,
    XML_ERR_ENTITY_NOT_STARTED,
    XML_ERR_ENTITY_NOT_FINISHED,
    XML_ERR_LT_IN_ATTRIBUTE,
    XML_ERR_ATTRIBUTE_NOT_STARTED,
    XML_ERR_ATTRIBUTE_NOT_FINISHED,
    XML_ERR_ATTRIBUTE_WITHOUT_VALUE,
    XML_ERR_ATTRIBUTE_REDEFINED,
    XML_ERR_LITERAL_NOT_STARTED,
    XML_ERR_LITERAL_NOT_FINISHED,
    XML_ERR_COMMENT_NOT_FINISHED,
    XML_ERR_PI_NOT_STARTED,
    XML_ERR_PI_NOT_FINISHED,
    XML_ERR_NOTATION_NOT_STARTED,
    XML_ERR_NOTATION_NOT_FINISHED,
    XML_ERR_ATTLIST_NOT_STARTED,
    XML_ERR_ATTLIST_NOT_FINISHED,
    XML_ERR_MIXED_NOT_STARTED,
    XML_ERR_MIXED_NOT_FINISHED,
    XML_ERR_ELEMCONTENT_NOT_STARTED,
    XML_ERR_ELEMCONTENT_NOT_FINISHED,
    XML_ERR_XMLDECL_NOT_STARTED,
    XML_ERR_XMLDECL_NOT_FINISHED,
    XML_ERR_CONDSEC_NOT_STARTED,
    XML_ERR_CONDSEC_NOT_FINISHED,
    XML_ERR_EXT_SUBSET_NOT_FINISHED,
    XML_ERR_DOCTYPE_NOT_FINISHED,
    XML_ERR_MISPLACED_CDATA_END,
```



```
XML_ERR_CDATA_NOT_FINISHED,
XML_ERR_RESERVED_XML_NAME,
XML_ERR_SPACE_REQUIRED,
XML_ERR_SEPARATOR_REQUIRED,
XML_ERR_NMTOKEN_REQUIRED,
XML_ERR_NAME_REQUIRED,
XML_ERR_PCDATA_REQUIRED,
XML_ERR_URI_REQUIRED,
XML_ERR_PUBID_REQUIRED,
XML_ERR_LT_REQUIRED,
XML_ERR_GT_REQUIRED,
XML_ERR_LTSLASH_REQUIRED,
XML_ERR_EQUAL_REQUIRED,
XML_ERR_TAG_NAME_MISMATCH,
XML_ERR_TAG_NOT_FINISHED,
XML_ERR_STANDALONE_VALUE,
XML_ERR_ENCODING_NAME,
XML_ERR_HYPHEN_IN_COMMENT,
XML_ERR_INVALID_ENCODING,
XML_ERR_EXT_ENTITY_STANDALONE,
XML_ERR_CONDSEC_INVALID,
XML_ERR_VALUE_REQUIRED,
XML_ERR_NOT_WELL_BALANCED,
XML_ERR_EXTRA_CONTENT,
XML_ERR_ENTITY_CHAR_ERROR,
XML_ERR_ENTITY_PE_INTERNAL,
XML_ERR_ENTITY_LOOP,
XML_ERR_ENTITY_BOUNDARY,
XML_ERR_INVALID_URI,
XML_ERR_URI_FRAGMENT,
XML_WAR_CATALOG_PI,
XML_ERR_NO_DTD,
XML_ERR_CONDSEC_INVALID_KEYWORD,
XML_ERR_VERSION_MISSING,
XML_WAR_UNKNOWN_VERSION,
XML_WAR_LANG_VALUE,
XML_WAR_NS_URI,
XML_WAR_NS_URI_RELATIVE,
XML_ERR_MISSING_ENCODING,
XML_WAR_SPACE_VALUE,
XML_ERR_NOT_STANDALONE,
XML_ERR_ENTITY_PROCESSING,
XML_ERR_NOTATION_PROCESSING,
XML_WAR_NS_COLUMN,
XML_WAR_ENTITY_REDEFINED,
XML_NS_ERR_XML_NAMESPACE = 200,
XML_NS_ERR_UNDEFINED_NAMESPACE,
XML_NS_ERR_QNAME,
XML_NS_ERR_ATTRIBUTE_REDEFINED,
XML_NS_ERR_EMPTY,
XML_DTD_ATTRIBUTE_DEFAULT = 500,
XML_DTD_ATTRIBUTE_REDEFINED,
XML_DTD_ATTRIBUTE_VALUE,
XML_DTD_CONTENT_ERROR,
XML_DTD_CONTENT_MODEL,
XML_DTD_CONTENT_NOT_DETERMINIST,
XML_DTD_DIFFERENT_PREFIX,
XML_DTD_ELEM_DEFAULT_NAMESPACE,
XML_DTD_ELEM_NAMESPACE,
XML_DTD_ELEM_REDEFINED,
XML_DTD_EMPTY_NOTATION,
XML_DTD_ENTITY_TYPE,
XML_DTD_ID_FIXED,
XML_DTD_ID_REDEFINED,
XML_DTD_ID_SUBSET,
XML_DTD_INVALID_CHILD,
```

```
XML_DTD_INVALID_DEFAULT,
XML_DTD_LOAD_ERROR,
XML_DTD_MISSING_ATTRIBUTE,
XML_DTD_MIXED_CORRUPT,
XML_DTD_MULTIPLE_ID,
XML_DTD_NO_DOC,
XML_DTD_NO_DTD,
XML_DTD_NO_ELEM_NAME,
XML_DTD_NO_PREFIX,
XML_DTD_NO_ROOT,
XML_DTD_NOTATION_REDEFINED,
XML_DTD_NOTATION_VALUE,
XML_DTD_NOT_EMPTY,
XML_DTD_NOT_PCDATA,
XML_DTD_NOT_STANDALONE,
XML_DTD_ROOT_NAME,
XML_DTD_STANDALONE_WHITE_SPACE,
XML_DTD_UNKNOWN_ATTRIBUTE,
XML_DTD_UNKNOWN_ELEM,
XML_DTD_UNKNOWN_ENTITY,
XML_DTD_UNKNOWN_ID,
XML_DTD_UNKNOWN_NOTATION,
XML_DTD_STANDALONE_DEFAULTED,
XML_DTD_XMLID_VALUE,
XML_DTD_XMLID_TYPE,
XML_HTML_STRUCURE_ERROR = 800,
XML_HTML_UNKNOWN_TAG,
XML_RNGP_ANYNAME_ATTR_ANCESTOR = 1000,
XML_RNGP_ATTR_CONFLICT,
XML_RNGP_ATTRIBUTE_CHILDREN,
XML_RNGP_ATTRIBUTE_CONTENT,
XML_RNGP_ATTRIBUTE_EMPTY,
XML_RNGP_ATTRIBUTE_NOOP,
XML_RNGP_CHOICE_CONTENT,
XML_RNGP_CHOICE_EMPTY,
XML_RNGP_CREATE_FAILURE,
XML_RNGP_DATA_CONTENT,
XML_RNGP_DEF_CHOICE_AND_INTERLEAVE,
XML_RNGP_DEFINE_CREATE_FAILED,
XML_RNGP_DEFINE_EMPTY,
XML_RNGP_DEFINE_MISSING,
XML_RNGP_DEFINE_NAME_MISSING,
XML_RNGP_ELEM_CONTENT_EMPTY,
XML_RNGP_ELEM_CONTENT_ERROR,
XML_RNGP_ELEMENT_EMPTY,
XML_RNGP_ELEMENT_CONTENT,
XML_RNGP_ELEMENT_NAME,
XML_RNGP_ELEMENT_NO_CONTENT,
XML_RNGP_ELEM_TEXT_CONFLICT,
XML_RNGP_EMPTY,
XML_RNGP_EMPTY_CONSTRUCT,
XML_RNGP_EMPTY_CONTENT,
XML_RNGP_EMPTY_NOT_EMPTY,
XML_RNGP_ERROR_TYPE_LIB,
XML_RNGP_EXCEPT_EMPTY,
XML_RNGP_EXCEPT_MISSING,
XML_RNGP_EXCEPT_MULTIPLE,
XML_RNGP_EXCEPT_NO_CONTENT,
XML_RNGP_EXTERNALREF_EMTPY,
XML_RNGP_EXTERNAL_REF_FAILURE,
XML_RNGP_EXTERNALREF_RECURSE,
XML_RNGP_FORBIDDEN_ATTRIBUTE,
XML_RNGP_FOREIGN_ELEMENT,
XML_RNGP_GRAMMAR_CONTENT,
XML_RNGP_GRAMMAR_EMPTY,
XML_RNGP_GRAMMAR_MISSING,
```



```
XML_RNGP_GRAMMAR_NO_START,
XML_RNGP_GROUP_ATTR_CONFLICT,
XML_RNGP_HREF_ERROR,
XML_RNGP_INCLUDE_EMPTY,
XML_RNGP_INCLUDE_FAILURE,
XML_RNGP_INCLUDE_RECURSE,
XML_RNGP_INTERLEAVE_ADD,
XML_RNGP_INTERLEAVE_CREATE_FAILED,
XML_RNGP_INTERLEAVE_EMPTY,
XML_RNGP_INTERLEAVE_NO_CONTENT,
XML_RNGP_INVALID_DEFINE_NAME,
XML_RNGP_INVALID_URI,
XML_RNGP_INVALID_VALUE,
XML_RNGP_MISSING_HREF,
XML_RNGP_NAME_MISSING,
XML_RNGP_NEED_COMBINE,
XML_RNGP_NOTALLOWED_NOT_EMPTY,
XML_RNGP_NSNAME_ATTR_ANCESTOR,
XML_RNGP_NSNAME_NO_NS,
XML_RNGP_PARAM_FORBIDDEN,
XML_RNGP_PARAM_NAME_MISSING,
XML_RNGP_PARENTREF_CREATE_FAILED,
XML_RNGP_PARENTREF_NAME_INVALID,
XML_RNGP_PARENTREF_NO_NAME,
XML_RNGP_PARENTREF_NO_PARENT,
XML_RNGP_PARENTREF_NOT_EMPTY,
XML_RNGP_PARSE_ERROR,
XML_RNGP_PAT_ANYNAME_EXCEPT_ANYNAME,
XML_RNGP_PAT_ATTR_ATTR,
XML_RNGP_PAT_ATTR_ELEM,
XML_RNGP_PAT_DATA_EXCEPT_ATTR,
XML_RNGP_PAT_DATA_EXCEPT_ELEM,
XML_RNGP_PAT_DATA_EXCEPT_EMPTY,
XML_RNGP_PAT_DATA_EXCEPT_GROUP,
XML_RNGP_PAT_DATA_EXCEPT_INTERLEAVE,
XML_RNGP_PAT_DATA_EXCEPT_LIST,
XML_RNGP_PAT_DATA_EXCEPT_ONEMORE,
XML_RNGP_PAT_DATA_EXCEPT_REF,
XML_RNGP_PAT_DATA_EXCEPT_TEXT,
XML_RNGP_PAT_LIST_ATTR,
XML_RNGP_PAT_LIST_ELEM,
XML_RNGP_PAT_LIST_INTERLEAVE,
XML_RNGP_PAT_LIST_LIST,
XML_RNGP_PAT_LIST_REF,
XML_RNGP_PAT_LIST_TEXT,
XML_RNGP_PAT_NSNAME_EXCEPT_ANYNAME,
XML_RNGP_PAT_NSNAME_EXCEPT_NSNAME,
XML_RNGP_PAT_ONEMORE_GROUP_ATTR,
XML_RNGP_PAT_ONEMORE_INTERLEAVE_ATTR,
XML_RNGP_PAT_START_ATTR,
XML_RNGP_PAT_START_DATA,
XML_RNGP_PAT_START_EMPTY,
XML_RNGP_PAT_START_GROUP,
XML_RNGP_PAT_START_INTERLEAVE,
XML_RNGP_PAT_START_LIST,
XML_RNGP_PAT_START_ONEMORE,
XML_RNGP_PAT_START_TEXT,
XML_RNGP_PAT_START_VALUE,
XML_RNGP_PREFIX_UNDEFINED,
XML_RNGP_REF_CREATE_FAILED,
XML_RNGP_REF_CYCLE,
XML_RNGP_REF_NAME_INVALID,
XML_RNGP_REF_NO_DEF,
XML_RNGP_REF_NO_NAME,
XML_RNGP_REF_NOT_EMPTY,
XML_RNGP_START_CHOICE_AND_INTERLEAVE,
```

```
XML_RNGP_START_CONTENT,
XML_RNGP_START_EMPTY,
XML_RNGP_START_MISSING,
XML_RNGP_TEXT_EXPECTED,
XML_RNGP_TEXT_HAS_CHILD,
XML_RNGP_TYPE_MISSING,
XML_RNGP_TYPE_NOT_FOUND,
XML_RNGP_TYPE_VALUE,
XML_RNGP_UNKNOWN_ATTRIBUTE,
XML_RNGP_UNKNOWN_COMBINE,
XML_RNGP_UNKNOWN_CONSTRUCT,
XML_RNGP_UNKNOWN_TYPE_LIB,
XML_RNGP_URI_FRAGMENT,
XML_RNGP_URI_NOT_ABSOLUTE,
XML_RNGP_VALUE_EMPTY,
XML_RNGP_VALUE_NO_CONTENT,
XML_RNGP_XMLNS_NAME,
XML_RNGP_XML_NS,
XML_XPATH_EXPRESSION_OK = 1200,
XML_XPATH_NUMBER_ERROR,
XML_XPATH_UNFINISHED_LITERAL_ERROR,
XML_XPATH_START_LITERAL_ERROR,
XML_XPATH_VARIABLE_REF_ERROR,
XML_XPATH_UNDEF_VARIABLE_ERROR,
XML_XPATH_INVALID_PREDICATE_ERROR,
XML_XPATH_EXPR_ERROR,
XML_XPATH_UNCLOSED_ERROR,
XML_XPATH_UNKNOWN_FUNC_ERROR,
XML_XPATH_INVALID_OPERAND,
XML_XPATH_INVALID_TYPE,
XML_XPATH_INVALID_ARITY,
XML_XPATH_INVALID_CTXT_SIZE,
XML_XPATH_INVALID_CTXT_POSITION,
XML_XPATH_MEMORY_ERROR,
XML_XPTR_SYNTAX_ERROR,
XML_XPTR_RESOURCE_ERROR,
XML_XPTR_SUB_RESOURCE_ERROR,
XML_XPATH_UNDEF_PREFIX_ERROR,
XML_XPATH_ENCODING_ERROR,
XML_XPATH_INVALID_CHAR_ERROR,
XML_TREE_INVALID_HEX = 1300,
XML_TREE_INVALID_DEC,
XML_TREE_UNTERMINATED_ENTITY,
XML_SAVE_NOT_UTF8 = 1400,
XML_SAVE_CHAR_INVALID,
XML_SAVE_NO_DOCTYPE,
XML_SAVE_UNKNOWN_ENCODING,
XML_REGEXP_COMPILE_ERROR = 1450,
XML_IO_UNKNOWN = 1500,
XML_IO_EACCES,
XML_IO_EAGAIN,
XML_IO_EBADF,
XML_IO_EBADMSG,
XML_IO_EBUSY,
XML_IO_ECANCELED,
XML_IO_ECHILD,
XML_IO_EDEADLK,
XML_IO_EDOM,
XML_IO_EEXIST,
XML_IO_EFAULT,
XML_IO_EFBIG,
XML_IO_EINPROGRESS,
XML_IO_EINTR,
XML_IO_EINVAL,
XML_IO_EIO,
XML_IO_EISDIR,
```



```
XML_IO_EMFILE,
XML_IO_EMLINK,
XML_IO_EMSGSIZE,
XML_IO_ENAMETOOLONG,
XML_IO_ENFILE,
XML_IO_ENODEV,
XML_IO_ENOENT,
XML_IO_ENOEXEC,
XML_IO_ENOLCK,
XML_IO_ENOMEM,
XML_IO_ENOSPC,
XML_IO_ENOSYS,
XML_IO_ENOTDIR,
XML_IO_ENOTEMPTY,
XML_IO_ENOTSUP,
XML_IO_ENOTTY,
XML_IO_ENXIO,
XML_IO_EPERM,
XML_IO_EPIPE,
XML_IO_ERANGE,
XML_IO_EROFS,
XML_IO_ESPIPE,
XML_IO_ESRCH,
XML_IO_ETIMEDOUT,
XML_IO_EXDEV,
XML_IO_NETWORK_ATTEMPT,
XML_IO_ENCODER,
XML_IO_FLUSH,
XML_IO_WRITE,
XML_IO_NO_INPUT,
XML_IO_BUFFER_FULL,
XML_IO_LOAD_ERROR,
XML_IO_ENOTSOCK,
XML_IO_EISCONN,
XML_IO_ECONNREFUSED,
XML_IO_ENETUNREACH,
XML_IO_EADDRINUSE,
XML_IO_EALREADY,
XML_IO_EAFNOSUPPORT,
XML_XINCLUDE_RECURSION = 1600,
XML_XINCLUDE_PARSE_VALUE,
XML_XINCLUDE_ENTITY_DEF_MISMATCH,
XML_XINCLUDE_NO_HREF,
XML_XINCLUDE_NO_FALLBACK,
XML_XINCLUDE_HREF_URI,
XML_XINCLUDE_TEXT_FRAGMENT,
XML_XINCLUDE_TEXT_DOCUMENT,
XML_XINCLUDE_INVALID_CHAR,
XML_XINCLUDE_BUILD_FAILED,
XML_XINCLUDE_UNKNOWN_ENCODING,
XML_XINCLUDE_MULTIPLE_ROOT,
XML_XINCLUDE_XPTR_FAILED,
XML_XINCLUDE_XPTR_RESULT,
XML_XINCLUDE_INCLUDE_IN_INCLUDE,
XML_XINCLUDE_FALLBACKS_IN_INCLUDE,
XML_XINCLUDE_FALLBACK_NOT_IN_INCLUDE,
XML_XINCLUDE_DEPRECATED_NS,
XML_XINCLUDE_FRAGMENT_ID,
XML_CATALOG_MISSING_ATTR = 1650,
XML_CATALOG_ENTRY_BROKEN,
XML_CATALOG_PREFER_VALUE,
XML_CATALOG_NOT_CATALOG,
XML_CATALOG_RECURSION,
XML_SCHEMAP_PREFIX_UNDEFINED = 1700,
XML_SCHEMAP_ATTRFORMDEFAULT_VALUE,
XML_SCHEMAP_ATTRGRP_NONAME_NOREF,
```

```
XML_SCHEMAP_ATTR_NONAME_NOREF,
XML_SCHEMAP_COMPLEXTYPE_NONAME_NOREF,
XML_SCHEMAP_ELEMFORMDEFAULT_VALUE,
XML_SCHEMAP_ELEM_NONAME_NOREF,
XML_SCHEMAP_EXTENSION_NO_BASE,
XML_SCHEMAP_FACET_NO_VALUE,
XML_SCHEMAP_FAILED_BUILD_IMPORT,
XML_SCHEMAP_GROUP_NONAME_NOREF,
XML_SCHEMAP_IMPORT_NAMESPACE_NOT_URI,
XML_SCHEMAP_IMPORT_REDEFINE_NSNAME,
XML_SCHEMAP_IMPORT_SCHEMA_NOT_URI,
XML_SCHEMAP_INVALID_BOOLEAN,
XML_SCHEMAP_INVALID_ENUM,
XML_SCHEMAP_INVALID_FACET,
XML_SCHEMAP_INVALID_FACET_VALUE,
XML_SCHEMAP_INVALID_MAXOCCURS,
XML_SCHEMAP_INVALID_MINOCCURS,
XML_SCHEMAP_INVALID_REF_AND_SUBTYPE,
XML_SCHEMAP_INVALID_WHITE_SPACE,
XML_SCHEMAP_NOATTR_NOREF,
XML_SCHEMAP_NOTATION_NO_NAME,
XML_SCHEMAP_NOTYPE_NOREF,
XML_SCHEMAP_REF_AND_SUBTYPE,
XML_SCHEMAP_RESTRICTION_NONAME_NOREF,
XML_SCHEMAP_SIMPLETYPE_NONAME,
XML_SCHEMAP_TYPE_AND_SUBTYPE,
XML_SCHEMAP_UNKNOWN_ALL_CHILD,
XML_SCHEMAP_UNKNOWN_ANYATTRIBUTE_CHILD,
XML_SCHEMAP_UNKNOWN_ATTR_CHILD,
XML_SCHEMAP_UNKNOWN_ATTRGRP_CHILD,
XML_SCHEMAP_UNKNOWN_ATTRIBUTE_GROUP,
XML_SCHEMAP_UNKNOWN_BASE_TYPE,
XML_SCHEMAP_UNKNOWN_CHOICE_CHILD,
XML_SCHEMAP_UNKNOWN_COMPLEXCONTENT_CHILD,
XML_SCHEMAP_UNKNOWN_COMPLEXTYPE_CHILD,
XML_SCHEMAP_UNKNOWN_ELEM_CHILD,
XML_SCHEMAP_UNKNOWN_EXTENSION_CHILD,
XML_SCHEMAP_UNKNOWN_FACET_CHILD,
XML_SCHEMAP_UNKNOWN_FACET_TYPE,
XML_SCHEMAP_UNKNOWN_GROUP_CHILD,
XML_SCHEMAP_UNKNOWN_IMPORT_CHILD,
XML_SCHEMAP_UNKNOWN_LIST_CHILD,
XML_SCHEMAP_UNKNOWN_NOTATION_CHILD,
XML_SCHEMAP_UNKNOWN_PROCESSCONTENT_CHILD,
XML_SCHEMAP_UNKNOWN_REF,
XML_SCHEMAP_UNKNOWN_RESTRICTION_CHILD,
XML_SCHEMAP_UNKNOWN_SCHEMAS_CHILD,
XML_SCHEMAP_UNKNOWN_SEQUENCE_CHILD,
XML_SCHEMAP_UNKNOWN_SIMPLECONTENT_CHILD,
XML_SCHEMAP_UNKNOWN_SIMPLETYPE_CHILD,
XML_SCHEMAP_UNKNOWN_TYPE,
XML_SCHEMAP_UNKNOWN_UNION_CHILD,
XML_SCHEMAP_ELEM_DEFAULT_FIXED,
XML_SCHEMAP_REGEXP_INVALID,
XML_SCHEMAP_FAILED_LOAD,
XML_SCHEMAP_NOTHING_TO_PARSE,
XML_SCHEMAP_NOROOT,
XML_SCHEMAP_REDEFINED_GROUP,
XML_SCHEMAP_REDEFINED_TYPE,
XML_SCHEMAP_REDEFINED_ELEMENT,
XML_SCHEMAP_REDEFINED_ATTRGROUP,
XML_SCHEMAP_REDEFINED_ATTR,
XML_SCHEMAP_REDEFINED_NOTATION,
XML_SCHEMAP_FAILED_PARSE,
XML_SCHEMAP_UNKNOWN_PREFIX,
XML_SCHEMAP_DEF_AND_PREFIX,
```



```
XML_SCHEMAP_UNKNOWN_INCLUDE_CHILD,
XML_SCHEMAP_INCLUDE_SCHEMA_NOT_URI,
XML_SCHEMAP_INCLUDE_SCHEMA_NO_URI,
XML_SCHEMAP_NOT_SCHEMA,
XML_SCHEMAP_UNKNOWN_MEMBER_TYPE,
XML_SCHEMAP_INVALID_ATTR_USE,
XML_SCHEMAP_RECURSIVE,
XML_SCHEMAP_SUPERNUMEROUS_LIST_ITEM_TYPE,
XML_SCHEMAP_INVALID_ATTR_COMBINATION,
XML_SCHEMAP_INVALID_ATTR_INLINE_COMBINATION,
XML_SCHEMAP_MISSING_SIMPLETYPE_CHILD,
XML_SCHEMAP_INVALID_ATTR_NAME,
XML_SCHEMAP_REF_AND_CONTENT,
XML_SCHEMAP_CT_PROPS_CORRECT_1,
XML_SCHEMAP_CT_PROPS_CORRECT_2,
XML_SCHEMAP_CT_PROPS_CORRECT_3,
XML_SCHEMAP_CT_PROPS_CORRECT_4,
XML_SCHEMAP_CT_PROPS_CORRECT_5,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_1,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_2_1_1,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_2_1_2,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_2_2,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_3,
XML_SCHEMAP_WILDCARD_INVALID_NS_MEMBER,
XML_SCHEMAP_INTERSECTION_NOT_EXPRESSIBLE,
XML_SCHEMAP_UNION_NOT_EXPRESSIBLE,
XML_SCHEMAP_SRC_IMPORT_3_1,
XML_SCHEMAP_SRC_IMPORT_3_2,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_4_1,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_4_2,
XML_SCHEMAP_DERIVATION_OK_RESTRICTION_4_3,
XML_SCHEMAP_COS_CT_EXTENDS_1_3,
XML_SCHEMAV_NOROOT = 1801,
XML_SCHEMAV_UNDECLAREDELEM,
XML_SCHEMAV_NOTTOPLEVEL,
XML_SCHEMAV_MISSING,
XML_SCHEMAV_WRONGELEM,
XML_SCHEMAV_NOTYPE,
XML_SCHEMAV_NOROLLBACK,
XML_SCHEMAV_ISABSTRACT,
XML_SCHEMAV_NOTEMPTY,
XML_SCHEMAV_ELEMCONT,
XML_SCHEMAV_HAVEDEFAULT,
XML_SCHEMAV_NOTNILLABLE,
XML_SCHEMAV_EXTRACONTENT,
XML_SCHEMAV_INVALIDATTR,
XML_SCHEMAV_INVALIDELEM,
XML_SCHEMAV_NOTDETERMINIST,
XML_SCHEMAV_CONSTRUCT,
XML_SCHEMAV_INTERNAL,
XML_SCHEMAV_NOTSIMPLE,
XML_SCHEMAV_ATTRUNKNOWN,
XML_SCHEMAV_ATTRINVALID,
XML_SCHEMAV_VALUE,
XML_SCHEMAV_FACET,
XML_SCHEMAV_CVC_DATATYPE_VALID_1_2_1,
XML_SCHEMAV_CVC_DATATYPE_VALID_1_2_2,
XML_SCHEMAV_CVC_DATATYPE_VALID_1_2_3,
XML_SCHEMAV_CVC_TYPE_3_1_1,
XML_SCHEMAV_CVC_TYPE_3_1_2,
XML_SCHEMAV_CVC_FACET_VALID,
XML_SCHEMAV_CVC_LENGTH_VALID,
XML_SCHEMAV_CVC_MINLENGTH_VALID,
XML_SCHEMAV_CVC_MAXLENGTH_VALID,
XML_SCHEMAV_CVC_MININCLUSIVE_VALID,
XML_SCHEMAV_CVC_MAXINCLUSIVE_VALID,
```

```
XML_SCHEMAV_CVC_MINEXCLUSIVE_VALID,
XML_SCHEMAV_CVC_MAXEXCLUSIVE_VALID,
XML_SCHEMAV_CVC_TOTALDIGITS_VALID,
XML_SCHEMAV_CVC_FRACTIONDIGITS_VALID,
XML_SCHEMAV_CVC_PATTERN_VALID,
XML_SCHEMAV_CVC_ENUMERATION_VALID,
XML_SCHEMAV_CVC_COMPLEX_TYPE_2_1,
XML_SCHEMAV_CVC_COMPLEX_TYPE_2_2,
XML_SCHEMAV_CVC_COMPLEX_TYPE_2_3,
XML_SCHEMAV_CVC_COMPLEX_TYPE_2_4,
XML_SCHEMAV_CVC_ELT_1,
XML_SCHEMAV_CVC_ELT_2,
XML_SCHEMAV_CVC_ELT_3_1,
XML_SCHEMAV_CVC_ELT_3_2_1,
XML_SCHEMAV_CVC_ELT_3_2_2,
XML_SCHEMAV_CVC_ELT_4_1,
XML_SCHEMAV_CVC_ELT_4_2,
XML_SCHEMAV_CVC_ELT_4_3,
XML_SCHEMAV_CVC_ELT_5_1_1,
XML_SCHEMAV_CVC_ELT_5_1_2,
XML_SCHEMAV_CVC_ELT_5_2_1,
XML_SCHEMAV_CVC_ELT_5_2_2_1,
XML_SCHEMAV_CVC_ELT_5_2_2_2_1,
XML_SCHEMAV_CVC_ELT_5_2_2_2_2,
XML_SCHEMAV_CVC_ELT_6,
XML_SCHEMAV_CVC_ELT_7,
XML_SCHEMAV_CVC_ATTRIBUTE_1,
XML_SCHEMAV_CVC_ATTRIBUTE_2,
XML_SCHEMAV_CVC_ATTRIBUTE_3,
XML_SCHEMAV_CVC_ATTRIBUTE_4,
XML_SCHEMAV_CVC_COMPLEX_TYPE_3_1,
XML_SCHEMAV_CVC_COMPLEX_TYPE_3_2_1,
XML_SCHEMAV_CVC_COMPLEX_TYPE_3_2_2,
XML_SCHEMAV_CVC_COMPLEX_TYPE_4,
XML_SCHEMAV_CVC_COMPLEX_TYPE_5_1,
XML_SCHEMAV_CVC_COMPLEX_TYPE_5_2,
XML_SCHEMAV_ELEMENT_CONTENT,
XML_SCHEMAV_DOCUMENT_ELEMENT_MISSING,
XML_SCHEMAV_CVC_COMPLEX_TYPE_1,
XML_SCHEMAV_CVC_AU,
XML_SCHEMAV_CVC_TYPE_1,
XML_SCHEMAV_CVC_TYPE_2,
XML_SCHEMAV_CVC_IDC,
XML_SCHEMAV_CVC_WILDCARD,
XML_XPTR_UNKNOWN_SCHEME = 1900,
XML_XPTR_CHILDSEQ_START,
XML_XPTR_EVAL_FAILED,
XML_XPTR_EXTRA_OBJECTS,
XML_C14N_CREATE_CTXT = 1950,
XML_C14N_REQUIRES_UTF8,
XML_C14N_CREATE_STACK,
XML_C14N_INVALID_NODE,
XML_C14N_UNKNOW_NODE,
XML_C14N_RELATIVE_NAMESPACE,
XML_FTP_PASV_ANSWER = 2000,
XML_FTP_EPSV_ANSWER,
XML_FTP_ACCNT,
XML_FTP_URL_SYNTAX,
XML_HTTP_URL_SYNTAX = 2020,
XML_HTTP_USE_IP,
XML_HTTP_UNKNOWN_HOST,
XML_SCHEMAP_SRC_SIMPLE_TYPE_1 = 3000,
XML_SCHEMAP_SRC_SIMPLE_TYPE_2,
XML_SCHEMAP_SRC_SIMPLE_TYPE_3,
XML_SCHEMAP_SRC_SIMPLE_TYPE_4,
XML_SCHEMAP_SRC_RESOLVE,
```



```
XML_SCHEMAP_SRC_RESTRICTION_BASE_OR_SIMPLETYPE,
XML_SCHEMAP_SRC_LIST_ITEMTYPE_OR_SIMPLETYPE,
XML_SCHEMAP_SRC_UNION_MEMBERTYPES_OR_SIMPLETYPES,
XML_SCHEMAP_ST_PROPS_CORRECT_1,
XML_SCHEMAP_ST_PROPS_CORRECT_2,
XML_SCHEMAP_ST_PROPS_CORRECT_3,
XML_SCHEMAP_COS_ST_RESTRICTS_1_1,
XML_SCHEMAP_COS_ST_RESTRICTS_1_2,
XML_SCHEMAP_COS_ST_RESTRICTS_1_3_1,
XML_SCHEMAP_COS_ST_RESTRICTS_1_3_2,
XML_SCHEMAP_COS_ST_RESTRICTS_2_1,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_1_1,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_1_2,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_2_1,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_2_2,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_2_3,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_2_4,
XML_SCHEMAP_COS_ST_RESTRICTS_2_3_2_5,
XML_SCHEMAP_COS_ST_RESTRICTS_3_1,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_1,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_1_2,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_2_2,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_2_1,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_2_3,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_2_4,
XML_SCHEMAP_COS_ST_RESTRICTS_3_3_2_5,
XML_SCHEMAP_COS_ST_DERIVED_OK_2_1,
XML_SCHEMAP_COS_ST_DERIVED_OK_2_2,
XML_SCHEMAP_S4S_ELEM_NOT_ALLOWED,
XML_SCHEMAP_S4S_ELEM_MISSING,
XML_SCHEMAP_S4S_ATTR_NOT_ALLOWED,
XML_SCHEMAP_S4S_ATTR_MISSING,
XML_SCHEMAP_S4S_ATTR_INVALID_VALUE,
XML_SCHEMAP_SRC_ELEMENT_1,
XML_SCHEMAP_SRC_ELEMENT_2_1,
XML_SCHEMAP_SRC_ELEMENT_2_2,
XML_SCHEMAP_SRC_ELEMENT_3,
XML_SCHEMAP_P_PROPS_CORRECT_1,
XML_SCHEMAP_P_PROPS_CORRECT_2_1,
XML_SCHEMAP_P_PROPS_CORRECT_2_2,
XML_SCHEMAP_E_PROPS_CORRECT_2,
XML_SCHEMAP_E_PROPS_CORRECT_3,
XML_SCHEMAP_E_PROPS_CORRECT_4,
XML_SCHEMAP_E_PROPS_CORRECT_5,
XML_SCHEMAP_E_PROPS_CORRECT_6,
XML_SCHEMAP_SRC_INCLUDE,
XML_SCHEMAP_SRC_ATTRIBUTE_1,
XML_SCHEMAP_SRC_ATTRIBUTE_2,
XML_SCHEMAP_SRC_ATTRIBUTE_3_1,
XML_SCHEMAP_SRC_ATTRIBUTE_3_2,
XML_SCHEMAP_SRC_ATTRIBUTE_4,
XML_SCHEMAP_NO_XMLNS,
XML_SCHEMAP_NO_XSI,
XML_SCHEMAP_COS_VALID_DEFAULT_1,
XML_SCHEMAP_COS_VALID_DEFAULT_2_1,
XML_SCHEMAP_COS_VALID_DEFAULT_2_2_1,
XML_SCHEMAP_COS_VALID_DEFAULT_2_2_2,
XML_SCHEMAP_CVC_SIMPLE_TYPE,
XML_SCHEMAP_COS_CT_EXTENDS_1_1,
XML_SCHEMAP_SRC_IMPORT_1_1,
XML_SCHEMAP_SRC_IMPORT_1_2,
XML_SCHEMAP_SRC_IMPORT_2,
XML_SCHEMAP_SRC_IMPORT_2_1,
XML_SCHEMAP_SRC_IMPORT_2_2,
XML_SCHEMAP_INTERNAL,
XML_SCHEMAP_NOT_DETERMINISTIC,
```

```
    XML_SCHEMAP_SRC_ATTRIBUTE_GROUP_1,
    XML_SCHEMAP_SRC_ATTRIBUTE_GROUP_2,
    XML_SCHEMAP_SRC_ATTRIBUTE_GROUP_3,
    XML_SCHEMAP_MG_PROPS_CORRECT_1,
    XML_SCHEMAP_MG_PROPS_CORRECT_2,
    XML_SCHEMAP_SRC_CT_1,
    XML_SCHEMAP_DERIVATION_OK_RESTRICTION_2_1_3,
    XML_SCHEMAP_AU_PROPS_CORRECT_2,
    XML_SCHEMAP_A_PROPS_CORRECT_2,
    XML_SCHEMAP_C_PROPS_CORRECT,
    XML_SCHEMAP_SRC_REDEFINE,
    XML_SCHEMAP_SRC_IMPORT,
    XML_SCHEMAP_WARN_SKIP_SCHEMA,
    XML_SCHEMAP_WARN_UNLOCATED_SCHEMA,
    XML_SCHEMAP_WARN_ATTR_REDECL_PROH,
    XML_SCHEMAP_WARN_ATTR_POINTLESS_PROH,
    XML_MODULE_OPEN = 4900,
    XML_MODULE_CLOSE,
    XML_CHECK_FOUND_ELEMENT = 5000,
    XML_CHECK_FOUND_ATTRIBUTE,
    XML_CHECK_FOUND_TEXT,
    XML_CHECK_FOUND_CDATA,
    XML_CHECK_FOUND_ENTITYREF,
    XML_CHECK_FOUND_ENTITY,
    XML_CHECK_FOUND_PI,
    XML_CHECK_FOUND_COMMENT,
    XML_CHECK_FOUND_DOCTYPE,
    XML_CHECK_FOUND_FRAGMENT,
    XML_CHECK_FOUND_NOTATION,
    XML_CHECK_UNKNOWN_NODE,
    XML_CHECK_ENTITY_TYPE,
    XML_CHECK_NO_PARENT,
    XML_CHECK_NO_DOC,
    XML_CHECK_NO_NAME,
    XML_CHECK_NO_ELEM,
    XML_CHECK_WRONG_DOC,
    XML_CHECK_NO_PREV,
    XML_CHECK_WRONG_PREV,
    XML_CHECK_NO_NEXT,
    XML_CHECK_WRONG_NEXT,
    XML_CHECK_NOT_DTD,
    XML_CHECK_NOT_ATTR,
    XML_CHECK_NOT_ATTR_DECL,
    XML_CHECK_NOT_ELEM_DECL,
    XML_CHECK_NOT_ENTITY_DECL,
    XML_CHECK_NOT_NS_DECL,
    XML_CHECK_NO_HREF,
    XML_CHECK_WRONG_PARENT,
    XML_CHECK_NS_SCOPE,
    XML_CHECK_NS_ANCESTOR,
    XML_CHECK_NOT_UTF8,
    XML_CHECK_NO_DICT,
    XML_CHECK_NOT_NCNAME,
    XML_CHECK_OUTSIDE_DICT,
    XML_CHECK_WRONG_NAME,
    XML_CHECK_NAME_NOT_NULL,
    XML_I18N_NO_NAME = 6000,
    XML_I18N_NO_HANDLER,
    XML_I18N_EXCESS_HANDLER,
    XML_I18N_CONV_FAILED,
    XML_I18N_NO_OUTPUT
} xmlParserErrors;
extern   void   initGenericErrorDefaultFunc(xmlGenericErrorFunc   *
handler);
extern int xmlCopyError(xmlErrorPtr from, xmlErrorPtr to);
extern xmlErrorPtr xmlCtxtGetLastError(void *ctx);
```



```
extern void xmlCtxtResetLastError(void *ctx);
extern xmlErrorPtr xmlGetLastError(void);
extern void xmlParserError(void *ctx, const char *msg, ...);
extern void xmlParserPrintFileContext(xmlParserInputPtr input);
extern void xmlParserPrintFileInfo(xmlParserInputPtr input);
extern void xmlParserValidityError(void *ctx, const char *msg, ...);
extern void xmlParserValidityWarning(void *ctx, const char
*msg, ...);
extern void xmlParserWarning(void *ctx, const char *msg, ...);
extern void xmlResetError(xmlErrorPtr err);
extern void xmlResetLastError(void);
extern void xmlSetGenericErrorFunc(void *ctx, xmlGenericErrorFunc
handler);
extern void xmlSetStructuredErrorFunc(void *ctx,
                                      xmlStructuredErrorFunc handler);
```

8.2.26 libxml2/libxml/xmlexports.h

```
#define XMLCALL
#define XMLCDECL
#define XMLPUBFUN
#define XMLPUBVAR      extern
#define LIBXML_DLL_IMPORT      XMLPUBVAR
```

8.2.27 libxml2/libxml/xmlmemory.h

```
typedef void (*xmlFreeFunc) (void *);
typedef void *(*xmlMallocFunc) (size_t);
typedef void *(*xmlReallocFunc) (void *, size_t);
typedef char *(*xmlStrdupFunc) (const char *);
extern void xmlCleanupMemory(void);
extern int xmlGcMemGet(xmlFreeFunc * freeFunc, xmlMallocFunc *
mallocFunc,
                       xmlMallocFunc * mallocAtomicFunc,
                       xmlReallocFunc * reallocFunc,
                       xmlStrdupFunc * strdupFunc);
extern int xmlGcMemSetup(xmlFreeFunc freeFunc, xmlMallocFunc
mallocFunc,
                         xmlMallocFunc mallocAtomicFunc,
                         xmlReallocFunc reallocFunc,
                         xmlStrdupFunc strdupFunc);
extern int xmlInitMemory(void);
extern void *xmlMallocAtomicLoc(size_t size, const char *file, int
line);
extern void *xmlMallocLoc(size_t size, const char *file, int line);
extern int xmlMemBlocks(void);
extern void xmlMemDisplay(FILE * fp);
extern void xmlMemFree(void *ptr);
extern int xmlMemGet(xmlFreeFunc * freeFunc, xmlMallocFunc *
mallocFunc,
                     xmlReallocFunc * reallocFunc,
                     xmlStrdupFunc * strdupFunc);
extern void *xmlMemMalloc(size_t size);
extern void *xmlMemRealloc(void *ptr, size_t size);
extern int xmlMemSetup(xmlFreeFunc freeFunc, xmlMallocFunc
mallocFunc,
                       xmlReallocFunc reallocFunc,
                       xmlStrdupFunc strdupFunc);
extern void xmlMemShow(FILE * fp, int nr);
extern char *xmlMemStrdupLoc(const char *str, const char *file, int
line);
extern int xmlMemUsed(void);
extern void xmlMemoryDump(void);
```



```
extern char *xmlMemoryStrdup(const char *str);
extern void *xmlReallocLoc(void *ptr, size_t size, const char *file,
                           int line);
```

8.2.28 libxml2/libxml/xmlmodule.h

```
typedef struct _xmlModule xmlModule;
typedef xmlModule *xmlModulePtr;
typedef enum {
    XML_MODULE_LAZY = 1,
    XML_MODULE_LOCAL = 2
} xmlModuleOption;
extern int xmlModuleClose(xmlModulePtr module);
extern int xmlModuleFree(xmlModulePtr module);
extern  xmlModulePtr  xmlModuleOpen(const  char  *filename,  int
options);
extern int xmlModuleSymbol(xmlModulePtr module, const char *name,
                           void **result);
```

8.2.29 libxml2/libxml/xmlreader.h

```
typedef struct _xmlTextReader xmlTextReader;
typedef xmlTextReader *xmlTextReaderPtr;
typedef enum {
    XML_PARSER_SEVERITY_VALIDITY_WARNING = 1,
    XML_PARSER_SEVERITY_VALIDITY_ERROR = 2,
    XML_PARSER_SEVERITY_WARNING = 3,
    XML_PARSER_SEVERITY_ERROR = 4
} xmlParserSeverities;
typedef void *xmlTextReaderLocatorPtr;
typedef void (*xmlTextReaderErrorFunc) (void *, const char *,
                                        xmlParserSeverities,
                                        xmlTextReaderLocatorPtr);
typedef enum {
    XML_PARSER_LOADDTD = 1,
    XML_PARSER_DEFAULTATTRS = 2,
    XML_PARSER_VALIDATE = 3,
    XML_PARSER_SUBST_ENTITIES = 4
} xmlParserProperties;
typedef enum {
    XML_READER_TYPE_NONE = 0,
    XML_READER_TYPE_ELEMENT = 1,
    XML_READER_TYPE_ATTRIBUTE = 2,
    XML_READER_TYPE_TEXT = 3,
    XML_READER_TYPE_CDATA = 4,
    XML_READER_TYPE_ENTITY_REFERENCE = 5,
    XML_READER_TYPE_ENTITY = 6,
    XML_READER_TYPE_PROCESSING_INSTRUCTION = 7,
    XML_READER_TYPE_COMMENT = 8,
    XML_READER_TYPE_DOCUMENT = 9,
    XML_READER_TYPE_DOCUMENT_TYPE = 10,
    XML_READER_TYPE_DOCUMENT_FRAGMENT = 11,
    XML_READER_TYPE_NOTATION = 12,
    XML_READER_TYPE_WHITESPACE = 13,
    XML_READER_TYPE_SIGNIFICANT_WHITESPACE = 14,
    XML_READER_TYPE_END_ELEMENT = 15,
    XML_READER_TYPE_END_ENTITY = 16,
    XML_READER_TYPE_XML_DECLARATION = 17
} xmlReaderTypes;
typedef enum {
    XML_TEXTREADER_MODE_INITIAL = 0,
    XML_TEXTREADER_MODE_INTERACTIVE = 1,
    XML_TEXTREADER_MODE_ERROR = 2,
```



```
    XML_TEXTREADER_MODE_EOF = 3,
    XML_TEXTREADER_MODE_CLOSED = 4,
    XML_TEXTREADER_MODE_READING = 5
} xmlTextReaderMode;
extern void xmlFreeTextReader(xmlTextReaderPtr reader);
extern  xmlTextReaderPtr  xmlNewTextReader(xmlParserInputBufferPtr
input,
                                    const char *URI);
extern xmlTextReaderPtr xmlNewTextReaderFilename(const char *URI);
extern xmlTextReaderPtr xmlReaderForDoc(const xmlChar * cur,
                                    const char *URL,
                                    const    char    *encoding,    int
options);
extern xmlTextReaderPtr xmlReaderForFd(int fd, const char *URL,
                                   const char *encoding, int options);
extern xmlTextReaderPtr xmlReaderForFile(const char *filename,
                                    const char *encoding,
                                    int options);
extern xmlTextReaderPtr xmlReaderForIO(xmlInputReadCallback ioread,
                                   xmlInputCloseCallback ioclose,
                                   void *ioctx, const char *URL,
                                   const char *encoding, int options);
extern xmlTextReaderPtr xmlReaderForMemory(const char *buffer, int
size,
                                     const char *URL,
                                     const char *encoding,
                                     int options);
extern int xmlReaderNewDoc(xmlTextReaderPtr reader, const xmlChar
* cur,
                       const char *URL, const char *encoding,
                       int options);
extern int xmlReaderNewFd(xmlTextReaderPtr reader, int fd, const
char *URL,
                      const char *encoding, int options);
extern int xmlReaderNewFile(xmlTextReaderPtr reader, const char
*filename,
                       const char *encoding, int options);
extern int xmlReaderNewIO(xmlTextReaderPtr reader,
                      xmlInputReadCallback ioread,
                      xmlInputCloseCallback ioclose, void *ioctx,
                      const char *URL, const char *encoding,
                      int options);
extern int xmlReaderNewMemory(xmlTextReaderPtr reader, const char
*buffer,
                        int size, const char *URL,
                        const char *encoding, int options);
extern int xmlReaderNewWalker(xmlTextReaderPtr reader, xmlDocPtr
doc);
extern xmlTextReaderPtr xmlReaderWalker(xmlDocPtr doc);
extern int xmlTextReaderAttributeCount(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderBaseUri(xmlTextReaderPtr reader);
extern long int xmlTextReaderByteConsumed(xmlTextReaderPtr reader);
extern int xmlTextReaderClose(xmlTextReaderPtr reader);
extern  const  xmlChar  *xmlTextReaderConstBaseUri(xmlTextReaderPtr
reader);
extern const xmlChar *xmlTextReaderConstEncoding(xmlTextReaderPtr
reader);
extern const xmlChar *xmlTextReaderConstLocalName(xmlTextReaderPtr
reader);
extern   const   xmlChar   *xmlTextReaderConstName(xmlTextReaderPtr
reader);
extern                          const                          xmlChar
*xmlTextReaderConstNamespaceUri(xmlTextReaderPtr
                                                reader);
extern  const  xmlChar  *xmlTextReaderConstPrefix(xmlTextReaderPtr
reader);
```

```
extern const xmlChar *xmlTextReaderConstString(xmlTextReaderPtr
reader,
                                               const xmlChar * str);
extern const xmlChar *xmlTextReaderConstValue(xmlTextReaderPtr
reader);
extern const xmlChar *xmlTextReaderConstXmlLang(xmlTextReaderPtr
reader);
extern                          const                          xmlChar
*xmlTextReaderConstXmlVersion(xmlTextReaderPtr
                                                    reader);
extern xmlDocPtr xmlTextReaderCurrentDoc(xmlTextReaderPtr reader);
extern    xmlNodePtr    xmlTextReaderCurrentNode(xmlTextReaderPtr
reader);
extern int xmlTextReaderDepth(xmlTextReaderPtr reader);
extern xmlNodePtr xmlTextReaderExpand(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderGetAttribute(xmlTextReaderPtr reader,
                                          const xmlChar * name);
extern    xmlChar    *xmlTextReaderGetAttributeNo(xmlTextReaderPtr
reader,
                                           int no);
extern    xmlChar    *xmlTextReaderGetAttributeNs(xmlTextReaderPtr
reader,
                                           const xmlChar * localName,
                                           const xmlChar * namespaceURI);
extern void xmlTextReaderGetErrorHandler(xmlTextReaderPtr reader,
                                         xmlTextReaderErrorFunc * f,
                                         void **arg);
extern   int   xmlTextReaderGetParserColumnNumber(xmlTextReaderPtr
reader);
extern    int    xmlTextReaderGetParserLineNumber(xmlTextReaderPtr
reader);
extern int xmlTextReaderGetParserProp(xmlTextReaderPtr reader, int
prop);
extern                                      xmlParserInputBufferPtr
xmlTextReaderGetRemainder(xmlTextReaderPtr
                                                        reader);
extern int xmlTextReaderHasAttributes(xmlTextReaderPtr reader);
extern int xmlTextReaderHasValue(xmlTextReaderPtr reader);
extern int xmlTextReaderIsDefault(xmlTextReaderPtr reader);
extern int xmlTextReaderIsEmptyElement(xmlTextReaderPtr reader);
extern int xmlTextReaderIsNamespaceDecl(xmlTextReaderPtr reader);
extern int xmlTextReaderIsValid(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderLocalName(xmlTextReaderPtr reader);
extern                                                        xmlChar
*xmlTextReaderLocatorBaseURI(xmlTextReaderLocatorPtr
                                           locator);
extern int xmlTextReaderLocatorLineNumber(xmlTextReaderLocatorPtr
locator);
extern    xmlChar    *xmlTextReaderLookupNamespace(xmlTextReaderPtr
reader,
                                            const xmlChar * prefix);
extern int xmlTextReaderMoveToAttribute(xmlTextReaderPtr reader,
                                        const xmlChar * name);
extern int xmlTextReaderMoveToAttributeNo(xmlTextReaderPtr reader,
int no);
extern int xmlTextReaderMoveToAttributeNs(xmlTextReaderPtr reader,
                                          const xmlChar * localName,
                                          const xmlChar * namespaceURI);
extern int xmlTextReaderMoveToElement(xmlTextReaderPtr reader);
extern    int    xmlTextReaderMoveToFirstAttribute(xmlTextReaderPtr
reader);
extern    int    xmlTextReaderMoveToNextAttribute(xmlTextReaderPtr
reader);
extern xmlChar *xmlTextReaderName(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderNamespaceUri(xmlTextReaderPtr reader);
extern int xmlTextReaderNext(xmlTextReaderPtr reader);
```



```
extern int xmlTextReaderNextSibling(xmlTextReaderPtr reader);
extern int xmlTextReaderNodeType(xmlTextReaderPtr reader);
extern int xmlTextReaderNormalization(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderPrefix(xmlTextReaderPtr reader);
extern xmlNodePtr xmlTextReaderPreserve(xmlTextReaderPtr reader);
extern int xmlTextReaderPreservePattern(xmlTextReaderPtr reader,
                                        const xmlChar * pattern,
                                        const xmlChar * *namespaces);
extern int xmlTextReaderQuoteChar(xmlTextReaderPtr reader);
extern int xmlTextReaderRead(xmlTextReaderPtr reader);
extern    int    xmlTextReaderReadAttributeValue(xmlTextReaderPtr
reader);
extern xmlChar *xmlTextReaderReadInnerXml(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderReadOuterXml(xmlTextReaderPtr reader);
extern int xmlTextReaderReadState(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderReadString(xmlTextReaderPtr reader);
extern int xmlTextReaderRelaxNGSetSchema(xmlTextReaderPtr reader,
                                         xmlRelaxNGPtr schema);
extern int xmlTextReaderRelaxNGValidate(xmlTextReaderPtr reader,
                                        const char *rng);
extern int xmlTextReaderSchemaValidate(xmlTextReaderPtr reader,
                                       const char *xsd);
extern void xmlTextReaderSetErrorHandler(xmlTextReaderPtr reader,
                                         xmlTextReaderErrorFunc f,
                                         void *arg);
extern int xmlTextReaderSetParserProp(xmlTextReaderPtr reader, int
prop,
                                      int value);
extern int xmlTextReaderSetSchema(xmlTextReaderPtr reader,
                                  xmlSchemaPtr schema);
extern                                                      void
xmlTextReaderSetStructuredErrorHandler(xmlTextReaderPtr reader,
                                                xmlStructuredErrorFunc
                                                f, void *arg);
extern int xmlTextReaderStandalone(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderValue(xmlTextReaderPtr reader);
extern xmlChar *xmlTextReaderXmlLang(xmlTextReaderPtr reader);
```

8.2.30 libxml2/libxml/xmlregexp.h

```
typedef struct _xmlRegexp xmlRegexp;
typedef xmlRegexp *xmlRegexpPtr;
typedef struct _xmlRegExecCtxt xmlRegExecCtxt;
typedef xmlRegExecCtxt *xmlRegExecCtxtPtr;
typedef struct _xmlExpNode xmlExpNode;
typedef xmlExpNode *xmlExpNodePtr;
typedef  void  (*xmlRegExecCallbacks)  (xmlRegExecCtxtPtr,  const
xmlChar *,
                                        void *, void *);
typedef struct _xmlExpCtxt xmlExpCtxt;
typedef xmlExpCtxt *xmlExpCtxtPtr;
typedef enum {
    XML_EXP_EMPTY = 0,
    XML_EXP_FORBID = 1,
    XML_EXP_ATOM = 2,
    XML_EXP_SEQ = 3,
    XML_EXP_OR = 4,
    XML_EXP_COUNT = 5
} xmlExpNodeType;
extern xmlExpNodePtr emptyExp;
extern xmlExpNodePtr forbiddenExp;
extern int xmlExpCtxtNbCons(xmlExpCtxtPtr ctxt);
extern int xmlExpCtxtNbNodes(xmlExpCtxtPtr ctxt);
extern void xmlExpDump(xmlBufferPtr buf, xmlExpNodePtr expr);
extern xmlExpNodePtr xmlExpExpDerive(xmlExpCtxtPtr ctxt,
```



```
                                    xmlExpNodePtr expr,
                                    xmlExpNodePtr sub);
extern void xmlExpFree(xmlExpCtxtPtr ctxt, xmlExpNodePtr expr);
extern void xmlExpFreeCtxt(xmlExpCtxtPtr ctxt);
extern int xmlExpGetLanguage(xmlExpCtxtPtr ctxt, xmlExpNodePtr expr,
                             const xmlChar * *langList, int len);
extern int xmlExpGetStart(xmlExpCtxtPtr ctxt, xmlExpNodePtr expr,
                          const xmlChar * *tokList, int len);
extern int xmlExpIsNillable(xmlExpNodePtr expr);
extern int xmlExpMaxToken(xmlExpNodePtr expr);
extern xmlExpNodePtr xmlExpNewAtom(xmlExpCtxtPtr ctxt,
                                   const xmlChar * name, int len);
extern xmlExpCtxtPtr xmlExpNewCtxt(int maxNodes, xmlDictPtr dict);
extern xmlExpNodePtr xmlExpNewOr(xmlExpCtxtPtr ctxt, xmlExpNodePtr
left,
                                 xmlExpNodePtr right);
extern xmlExpNodePtr xmlExpNewRange(xmlExpCtxtPtr ctxt,
                                    xmlExpNodePtr subset, int min,
                                    int max);
extern     xmlExpNodePtr     xmlExpNewSeq(xmlExpCtxtPtr     ctxt,
xmlExpNodePtr left,
                                  xmlExpNodePtr right);
extern  xmlExpNodePtr  xmlExpParse(xmlExpCtxtPtr  ctxt,  const  char
*expr);
extern void xmlExpRef(xmlExpNodePtr expr);
extern xmlExpNodePtr xmlExpStringDerive(xmlExpCtxtPtr ctxt,
                                        xmlExpNodePtr expr,
                                        const xmlChar * str, int len);
extern int xmlExpSubsume(xmlExpCtxtPtr ctxt, xmlExpNodePtr expr,
                         xmlExpNodePtr sub);
extern int xmlRegExecErrInfo(xmlRegExecCtxtPtr exec,
                             const xmlChar * *string, int *nbval,
                             int   *nbneg,   xmlChar   *   *values,   int
*terminal);
extern int xmlRegExecNextValues(xmlRegExecCtxtPtr exec, int *nbval,
                                int *nbneg, xmlChar * *values,
                                int *terminal);
extern int xmlRegExecPushString(xmlRegExecCtxtPtr exec,
                                const xmlChar * value, void *data);
extern int xmlRegExecPushString2(xmlRegExecCtxtPtr exec,
                                 const xmlChar * value,
                                 const xmlChar * value2, void *data);
extern void xmlRegFreeExecCtxt(xmlRegExecCtxtPtr exec);
extern void xmlRegFreeRegexp(xmlRegexpPtr regexp);
extern xmlRegExecCtxtPtr xmlRegNewExecCtxt(xmlRegexpPtr comp,
                                           xmlRegExecCallbacks callback,
                                           void *data);
extern xmlRegexpPtr xmlRegexpCompile(const xmlChar * regexp);
extern int xmlRegexpExec(xmlRegexpPtr comp, const xmlChar * value);
extern int xmlRegexpIsDeterminist(xmlRegexpPtr comp);
extern void xmlRegexpPrint(FILE * output, xmlRegexpPtr regexp);
```

8.2.31 libxml2/libxml/xmlsave.h

```
typedef struct _xmlSaveCtxt xmlSaveCtxt;
typedef xmlSaveCtxt *xmlSaveCtxtPtr;
typedef enum {
    XML_SAVE_FORMAT = 1 << 0,
    XML_SAVE_NO_DECL = 1 << 1,
    XML_SAVE_NO_EMPTY = 1 << 2,
    XML_SAVE_NO_XHTML = 1 << 3
} xmlSaveOption;
extern int xmlSaveClose(xmlSaveCtxtPtr ctxt);
extern long int xmlSaveDoc(xmlSaveCtxtPtr ctxt, xmlDocPtr doc);
extern int xmlSaveFlush(xmlSaveCtxtPtr ctxt);
```



```
extern int xmlSaveSetAttrEscape(xmlSaveCtxtPtr ctxt,
                                xmlCharEncodingOutputFunc escape);
extern int xmlSaveSetEscape(xmlSaveCtxtPtr ctxt,
                            xmlCharEncodingOutputFunc escape);
extern xmlSaveCtxtPtr xmlSaveToFd(int fd, const char *encoding,
                                  int options);
extern xmlSaveCtxtPtr xmlSaveToFilename(const char *filename,
                                        const    char    *encoding,    int
options);
extern xmlSaveCtxtPtr xmlSaveToIO(xmlOutputWriteCallback iowrite,
                                  xmlOutputCloseCallback ioclose,
                                  void *ioctx, const char *encoding,
                                  int options);
extern long int xmlSaveTree(xmlSaveCtxtPtr ctxt, xmlNodePtr node);
```

8.2.32 libxml2/libxml/xmlschemas.h

```
typedef struct _xmlSchemaValidCtxt xmlSchemaValidCtxt;
typedef xmlSchemaValidCtxt *xmlSchemaValidCtxtPtr;
typedef struct _xmlSchemaSAXPlug xmlSchemaSAXPlugStruct;
typedef xmlSchemaSAXPlugStruct *xmlSchemaSAXPlugPtr;
typedef struct _xmlSchemaParserCtxt xmlSchemaParserCtxt;
typedef xmlSchemaParserCtxt *xmlSchemaParserCtxtPtr;
typedef  void  (*xmlSchemaValidityErrorFunc)  (void  *,  const  char
*, ...);
typedef void (*xmlSchemaValidityWarningFunc) (void *, const char
*, ...);
typedef struct _xmlSchema xmlSchema;
typedef xmlSchema *xmlSchemaPtr;
typedef enum {
    XML_SCHEMA_VAL_VC_I_CREATE = 1 << 0
} xmlSchemaValidOption;
extern void xmlSchemaDump(FILE * output, xmlSchemaPtr schema);
extern void xmlSchemaFree(xmlSchemaPtr schema);
extern void xmlSchemaFreeParserCtxt(xmlSchemaParserCtxtPtr ctxt);
extern void xmlSchemaFreeValidCtxt(xmlSchemaValidCtxtPtr ctxt);
extern int xmlSchemaGetParserErrors(xmlSchemaParserCtxtPtr ctxt,
                                    xmlSchemaValidityErrorFunc * err,
                                    xmlSchemaValidityWarningFunc * warn,
                                    void **ctx);
extern int xmlSchemaGetValidErrors(xmlSchemaValidCtxtPtr ctxt,
                                   xmlSchemaValidityErrorFunc * err,
                                   xmlSchemaValidityWarningFunc * warn,
                                   void **ctx);
extern int xmlSchemaIsValid(xmlSchemaValidCtxtPtr ctxt);
extern xmlSchemaParserCtxtPtr  xmlSchemaNewDocParserCtxt(xmlDocPtr
doc);
extern xmlSchemaParserCtxtPtr xmlSchemaNewMemParserCtxt(const char
*buffer,
                                                        int size);
extern  xmlSchemaParserCtxtPtr  xmlSchemaNewParserCtxt(const  char
*URL);
extern  xmlSchemaValidCtxtPtr  xmlSchemaNewValidCtxt(xmlSchemaPtr
schema);
extern xmlSchemaPtr xmlSchemaParse(xmlSchemaParserCtxtPtr ctxt);
extern  xmlSchemaSAXPlugPtr  xmlSchemaSAXPlug(xmlSchemaValidCtxtPtr
ctxt,
                                              xmlSAXHandlerPtr * sax,
                                              void **user_data);
extern int xmlSchemaSAXUnplug(xmlSchemaSAXPlugPtr plug);
extern void xmlSchemaSetParserErrors(xmlSchemaParserCtxtPtr ctxt,
                                     xmlSchemaValidityErrorFunc err,
                                     xmlSchemaValidityWarningFunc warn,
                                     void *ctx);
extern void xmlSchemaSetValidErrors(xmlSchemaValidCtxtPtr ctxt,
```

```
                                xmlSchemaValidityErrorFunc err,
                                xmlSchemaValidityWarningFunc warn,
                                void *ctx);
extern int xmlSchemaSetValidOptions(xmlSchemaValidCtxtPtr ctxt,
                                int options);
extern                                                  void
xmlSchemaSetValidStructuredErrors(xmlSchemaValidCtxtPtr ctxt,
                                      xmlStructuredErrorFunc
                                      serror, void *ctx);
extern    int    xmlSchemaValidCtxtGetOptions(xmlSchemaValidCtxtPtr
ctxt);
extern int xmlSchemaValidateDoc(xmlSchemaValidCtxtPtr ctxt,
                           xmlDocPtr instance);
extern int xmlSchemaValidateFile(xmlSchemaValidCtxtPtr ctxt,
                            const char *filename, int options);
extern int xmlSchemaValidateOneElement(xmlSchemaValidCtxtPtr ctxt,
                              xmlNodePtr elem);
extern int xmlSchemaValidateStream(xmlSchemaValidCtxtPtr ctxt,
                            xmlParserInputBufferPtr input,
                            xmlCharEncoding enc,
                            xmlSAXHandlerPtr      sax,      void
*user_data);
```

8.2.33 libxml2/libxml/xmlschemastypes.h

```
typedef struct _xmlSchemaType xmlSchemaType;
typedef xmlSchemaType *xmlSchemaTypePtr;
typedef struct _xmlSchemaVal xmlSchemaVal;
typedef xmlSchemaVal *xmlSchemaValPtr;
typedef enum {
    XML_SCHEMAS_UNKNOWN = 0,
    XML_SCHEMAS_STRING = 1,
    XML_SCHEMAS_NORMSTRING = 2,
    XML_SCHEMAS_DECIMAL = 3,
    XML_SCHEMAS_TIME = 4,
    XML_SCHEMAS_GDAY = 5,
    XML_SCHEMAS_GMONTH = 6,
    XML_SCHEMAS_GMONTHDAY = 7,
    XML_SCHEMAS_GYEAR = 8,
    XML_SCHEMAS_GYEARMONTH = 9,
    XML_SCHEMAS_DATE = 10,
    XML_SCHEMAS_DATETIME = 11,
    XML_SCHEMAS_DURATION = 12,
    XML_SCHEMAS_FLOAT = 13,
    XML_SCHEMAS_DOUBLE = 14,
    XML_SCHEMAS_BOOLEAN = 15,
    XML_SCHEMAS_TOKEN = 16,
    XML_SCHEMAS_LANGUAGE = 17,
    XML_SCHEMAS_NMTOKEN = 18,
    XML_SCHEMAS_NMTOKENS = 19,
    XML_SCHEMAS_NAME = 20,
    XML_SCHEMAS_QNAME = 21,
    XML_SCHEMAS_NCNAME = 22,
    XML_SCHEMAS_ID = 23,
    XML_SCHEMAS_IDREF = 24,
    XML_SCHEMAS_IDREFS = 25,
    XML_SCHEMAS_ENTITY = 26,
    XML_SCHEMAS_ENTITIES = 27,
    XML_SCHEMAS_NOTATION = 28,
    XML_SCHEMAS_ANYURI = 29,
    XML_SCHEMAS_INTEGER = 30,
    XML_SCHEMAS_NPINTEGER = 31,
    XML_SCHEMAS_NINTEGER = 32,
    XML_SCHEMAS_NNINTEGER = 33,
    XML_SCHEMAS_PINTEGER = 34,
```



```
    XML_SCHEMAS_INT = 35,
    XML_SCHEMAS_UINT = 36,
    XML_SCHEMAS_LONG = 37,
    XML_SCHEMAS_ULONG = 38,
    XML_SCHEMAS_SHORT = 39,
    XML_SCHEMAS_USHORT = 40,
    XML_SCHEMAS_BYTE = 41,
    XML_SCHEMAS_UBYTE = 42,
    XML_SCHEMAS_HEXBINARY = 43,
    XML_SCHEMAS_BASE64BINARY = 44,
    XML_SCHEMAS_ANYTYPE = 45,
    XML_SCHEMAS_ANYSIMPLETYPE = 46
} xmlSchemaValType;
extern void xmlSchemaCleanupTypes(void);
extern xmlChar *xmlSchemaCollapseString(const xmlChar * value);
extern      int      xmlSchemaCompareValues(xmlSchemaValPtr      x,
xmlSchemaValPtr y);
extern void xmlSchemaFreeValue(xmlSchemaValPtr val);
extern  xmlSchemaTypePtr  xmlSchemaGetBuiltInType(xmlSchemaValType
type);
extern int xmlSchemaGetCanonValue(xmlSchemaValPtr val,
                                  const xmlChar * *retValue);
extern xmlSchemaValType xmlSchemaGetValType(xmlSchemaValPtr val);
extern void xmlSchemaInitTypes(void);
extern int xmlSchemaValPredefTypeNode(xmlSchemaTypePtr type,
                                      const xmlChar * value,
                                      xmlSchemaValPtr * val,
                                      xmlNodePtr node);
```

8.2.34 libxml2/libxml/xmlstring.h

```
#define BAD_CAST       (xmlChar *)

typedef unsigned char xmlChar;
extern xmlChar *xmlCharStrdup(const char *cur);
extern xmlChar *xmlCharStrndup(const char *cur, int len);
extern int xmlCheckUTF8(const unsigned char *utf);
extern int xmlGetUTF8Char(const unsigned char *utf, int *len);
extern int xmlStrEqual(const xmlChar * str1, const xmlChar * str2);
extern  int  xmlStrPrintf(xmlChar  *  buf,  int  len,  const  xmlChar  *
msg, ...);
extern int xmlStrQEqual(const xmlChar * pref, const xmlChar * name,
                        const xmlChar * str);
extern int xmlStrVPrintf(xmlChar * buf, int len, const xmlChar *
msg,
                         va_list ap);
extern  int  xmlStrcasecmp(const  xmlChar  *  str1,  const  xmlChar  *
str2);
extern const xmlChar *xmlStrcasestr(const xmlChar * str,
                                    const xmlChar * val);
extern xmlChar *xmlStrcat(xmlChar * cur, const xmlChar * add);
extern const xmlChar *xmlStrchr(const xmlChar * str, xmlChar val);
extern int xmlStrcmp(const xmlChar * str1, const xmlChar * str2);
extern xmlChar *xmlStrdup(const xmlChar * cur);
extern int xmlStrlen(const xmlChar * str);
extern  int  xmlStrncasecmp(const  xmlChar  *  str1,  const  xmlChar  *
str2,
                            int len);
extern xmlChar *xmlStrncat(xmlChar * cur, const xmlChar * add, int
len);
extern xmlChar *xmlStrncatNew(const xmlChar * str1, const xmlChar
* str2,
                              int len);
extern int xmlStrncmp(const xmlChar * str1, const xmlChar * str2,
int len);
```

```
extern xmlChar *xmlStrndup(const xmlChar * cur, int len);
extern const xmlChar *xmlStrstr(const xmlChar * str, const xmlChar
* val);
extern xmlChar *xmlStrsub(const xmlChar * str, int start, int len);
extern int xmlUTF8Charcmp(const xmlChar * utf1, const xmlChar *
utf2);
extern int xmlUTF8Size(const xmlChar * utf);
extern int xmlUTF8Strlen(const xmlChar * utf);
extern int xmlUTF8Strloc(const xmlChar * utf, const xmlChar *
utfchar);
extern xmlChar *xmlUTF8Strndup(const xmlChar * utf, int len);
extern const xmlChar *xmlUTF8Strpos(const xmlChar * utf, int pos);
extern int xmlUTF8Strsize(const xmlChar * utf, int len);
extern xmlChar *xmlUTF8Strsub(const xmlChar * utf, int start, int
len);
```

8.2.35 libxml2/libxml/xmlversion.h

```
#define LIBXML_AUTOMATA_ENABLED
#define LIBXML_C14N_ENABLED
#define LIBXML_CATALOG_ENABLED
#define LIBXML_DEBUG_ENABLED
#define LIBXML_DOCB_ENABLED
#define LIBXML_EXPR_ENABLED
#define LIBXML_FTP_ENABLED
#define LIBXML_HTML_ENABLED
#define LIBXML_HTTP_ENABLED
#define LIBXML_ICONV_ENABLED
#define LIBXML_ISO8859X_ENABLED
#define LIBXML_LEGACY_ENABLED
#define LIBXML_MODULES_ENABLED
#define LIBXML_OUTPUT_ENABLED
#define LIBXML_PATTERN_ENABLED
#define LIBXML_PUSH_ENABLED
#define LIBXML_READER_ENABLED
#define LIBXML_REGEXP_ENABLED
#define LIBXML_SAX1_ENABLED
#define LIBXML_SCHEMAS_ENABLED
#define LIBXML_SCHEMATRON_ENABLED
#define LIBXML_THREAD_ENABLED
#define LIBXML_TREE_ENABLED
#define LIBXML_UNICODE_ENABLED
#define LIBXML_VALID_ENABLED
#define LIBXML_VERSION_EXTRA    ""
#define LIBXML_WRITER_ENABLED
#define LIBXML_XINCLUDE_ENABLED
#define LIBXML_XPATH_ENABLED
#define LIBXML_XPTR_ENABLED
#define WITHOUT_TRIO
#define LIBXML_DOTTED_VERSION   "2.6.22"
#define LIBXML_VERSION_STRING   "20622"
#define LIBXML_MODULE_EXTENSION ".so"
#define LIBXML_VERSION  20622
#define LIBXML_TEST_VERSION    xmlCheckVersion(20622);
#define ATTRIBUTE_UNUSED       __attribute__((unused))

extern void xmlCheckVersion(int version);
```

8.2.36 libxml2/libxml/xmlwriter.h

```
#define xmlTextWriterWriteDocType      xmlTextWriterWriteDTD
#define                    xmlTextWriterWriteProcessingInstruction
xmlTextWriterWritePI
```



```
typedef struct _xmlTextWriter xmlTextWriter;
typedef xmlTextWriter *xmlTextWriterPtr;
extern void xmlFreeTextWriter(xmlTextWriterPtr writer);
extern xmlTextWriterPtr xmlNewTextWriter(xmlOutputBufferPtr out);
extern xmlTextWriterPtr xmlNewTextWriterDoc(xmlDocPtr * doc,
                                            int compression);
extern xmlTextWriterPtr xmlNewTextWriterFilename(const char *uri,
                                                 int compression);
extern xmlTextWriterPtr xmlNewTextWriterMemory(xmlBufferPtr buf,
                                               int compression);
extern                                                  xmlTextWriterPtr
xmlNewTextWriterPushParser(xmlParserCtxtPtr ctxt,
                                                int compression);
extern xmlTextWriterPtr xmlNewTextWriterTree(xmlDocPtr doc,
                                             xmlNodePtr node,
                                             int compression);
extern int xmlTextWriterEndAttribute(xmlTextWriterPtr writer);
extern int xmlTextWriterEndCDATA(xmlTextWriterPtr writer);
extern int xmlTextWriterEndComment(xmlTextWriterPtr writer);
extern int xmlTextWriterEndDTD(xmlTextWriterPtr writer);
extern int xmlTextWriterEndDTDAttlist(xmlTextWriterPtr writer);
extern int xmlTextWriterEndDTDElement(xmlTextWriterPtr writer);
extern int xmlTextWriterEndDTDEntity(xmlTextWriterPtr writer);
extern int xmlTextWriterEndDocument(xmlTextWriterPtr writer);
extern int xmlTextWriterEndElement(xmlTextWriterPtr writer);
extern int xmlTextWriterEndPI(xmlTextWriterPtr writer);
extern int xmlTextWriterFlush(xmlTextWriterPtr writer);
extern int xmlTextWriterFullEndElement(xmlTextWriterPtr writer);
extern  int  xmlTextWriterSetIndent(xmlTextWriterPtr  writer,  int
indent);
extern int xmlTextWriterSetIndentString(xmlTextWriterPtr writer,
                                        const xmlChar * str);
extern int xmlTextWriterStartAttribute(xmlTextWriterPtr writer,
                                       const xmlChar * name);
extern int xmlTextWriterStartAttributeNS(xmlTextWriterPtr writer,
                                         const xmlChar * prefix,
                                         const xmlChar * name,
                                         const xmlChar * namespaceURI);
extern int xmlTextWriterStartCDATA(xmlTextWriterPtr writer);
extern int xmlTextWriterStartComment(xmlTextWriterPtr writer);
extern int xmlTextWriterStartDTD(xmlTextWriterPtr writer,
                                 const xmlChar * name,
                                 const xmlChar * pubid,
                                 const xmlChar * sysid);
extern int xmlTextWriterStartDTDAttlist(xmlTextWriterPtr writer,
                                        const xmlChar * name);
extern int xmlTextWriterStartDTDElement(xmlTextWriterPtr writer,
                                        const xmlChar * name);
extern  int  xmlTextWriterStartDTDEntity(xmlTextWriterPtr  writer,
int pe,
                                         const xmlChar * name);
extern int xmlTextWriterStartDocument(xmlTextWriterPtr writer,
                                      const char *version,
                                      const char *encoding,
                                      const char *standalone);
extern int xmlTextWriterStartElement(xmlTextWriterPtr writer,
                                     const xmlChar * name);
extern int xmlTextWriterStartElementNS(xmlTextWriterPtr writer,
                                       const xmlChar * prefix,
                                       const xmlChar * name,
                                       const xmlChar * namespaceURI);
extern int xmlTextWriterStartPI(xmlTextWriterPtr writer,
                                const xmlChar * target);
extern int xmlTextWriterWriteAttribute(xmlTextWriterPtr writer,
                                       const xmlChar * name,
```

```
                                    const xmlChar * content);
extern int xmlTextWriterWriteAttributeNS(xmlTextWriterPtr writer,
                                     const xmlChar * prefix,
                                     const xmlChar * name,
                                     const xmlChar * namespaceURI,
                                     const xmlChar * content);
extern int xmlTextWriterWriteBase64(xmlTextWriterPtr writer,
                                    const char *data, int start, int len);
extern int xmlTextWriterWriteBinHex(xmlTextWriterPtr writer,
                                    const char *data, int start, int len);
extern int xmlTextWriterWriteCDATA(xmlTextWriterPtr writer,
                                   const xmlChar * content);
extern int xmlTextWriterWriteComment(xmlTextWriterPtr writer,
                                     const xmlChar * content);
extern int xmlTextWriterWriteDTD(xmlTextWriterPtr writer,
                                 const xmlChar * name,
                                 const xmlChar * pubid,
                                 const xmlChar * sysid,
                                 const xmlChar * subset);
extern int xmlTextWriterWriteDTDAttlist(xmlTextWriterPtr writer,
                                    const xmlChar * name,
                                    const xmlChar * content);
extern int xmlTextWriterWriteDTDElement(xmlTextWriterPtr writer,
                                    const xmlChar * name,
                                    const xmlChar * content);
extern  int  xmlTextWriterWriteDTDEntity(xmlTextWriterPtr  writer,
int pe,
                                    const xmlChar * name,
                                    const xmlChar * pubid,
                                    const xmlChar * sysid,
                                    const xmlChar * ndataid,
                                    const xmlChar * content);
extern  int  xmlTextWriterWriteDTDExternalEntity(xmlTextWriterPtr
writer,
                                         int pe,
                                         const xmlChar * name,
                                         const xmlChar * pubid,
                                         const xmlChar * sysid,
                                         const xmlChar * ndataid);
extern                                                       int
xmlTextWriterWriteDTDExternalEntityContents(xmlTextWriterPtr
                                                writer,
                                                const xmlChar *
                                                pubid,
                                                const xmlChar *
                                                sysid,
                                                const xmlChar *
                                                ndataid);
extern  int  xmlTextWriterWriteDTDInternalEntity(xmlTextWriterPtr
writer,
                                         int pe,
                                         const xmlChar * name,
                                         const xmlChar * content);
extern int xmlTextWriterWriteDTDNotation(xmlTextWriterPtr writer,
                                     const xmlChar * name,
                                     const xmlChar * pubid,
                                     const xmlChar * sysid);
extern int xmlTextWriterWriteElement(xmlTextWriterPtr writer,
                                 const xmlChar * name,
                                 const xmlChar * content);
extern int xmlTextWriterWriteElementNS(xmlTextWriterPtr writer,
                                   const xmlChar * prefix,
                                   const xmlChar * name,
                                   const xmlChar * namespaceURI,
                                   const xmlChar * content);
```



```
extern    int    xmlTextWriterWriteFormatAttribute(xmlTextWriterPtr
writer,
                                        const xmlChar * name,
                                        const char *format, ...);
extern   int   xmlTextWriterWriteFormatAttributeNS(xmlTextWriterPtr
writer,
                                         const xmlChar * prefix,
                                         const xmlChar * name,
                                         const xmlChar *
                                         namespaceURI,
                                         const char *format, ...);
extern int xmlTextWriterWriteFormatCDATA(xmlTextWriterPtr writer,
                                    const char *format, ...);
extern int xmlTextWriterWriteFormatComment(xmlTextWriterPtr writer,
                                     const char *format, ...);
extern int xmlTextWriterWriteFormatDTD(xmlTextWriterPtr writer,
                                  const xmlChar * name,
                                  const xmlChar * pubid,
                                  const xmlChar * sysid,
                                  const char *format, ...);
extern   int   xmlTextWriterWriteFormatDTDAttlist(xmlTextWriterPtr
writer,
                                         const xmlChar * name,
                                         const char *format, ...);
extern   int   xmlTextWriterWriteFormatDTDElement(xmlTextWriterPtr
writer,
                                         const xmlChar * name,
                                         const char *format, ...);
extern                                                        int
xmlTextWriterWriteFormatDTDInternalEntity(xmlTextWriterPtr
                                             writer, int pe,
                                             const xmlChar * name,
                                             const char *format,
                                             ...);
extern int xmlTextWriterWriteFormatElement(xmlTextWriterPtr writer,
                                     const xmlChar * name,
                                     const char *format, ...);
extern    int    xmlTextWriterWriteFormatElementNS(xmlTextWriterPtr
writer,
                                         const xmlChar * prefix,
                                         const xmlChar * name,
                                         const xmlChar * namespaceURI,
                                         const char *format, ...);
extern int xmlTextWriterWriteFormatPI(xmlTextWriterPtr writer,
                                const xmlChar * target,
                                const char *format, ...);
extern int xmlTextWriterWriteFormatRaw(xmlTextWriterPtr writer,
                                 const char *format, ...);
extern int xmlTextWriterWriteFormatString(xmlTextWriterPtr writer,
                                    const char *format, ...);
extern int xmlTextWriterWritePI(xmlTextWriterPtr writer,
                           const xmlChar * target,
                           const xmlChar * content);
extern int xmlTextWriterWriteRaw(xmlTextWriterPtr writer,
                            const xmlChar * content);
extern int xmlTextWriterWriteRawLen(xmlTextWriterPtr writer,
                               const xmlChar * content, int len);
extern int xmlTextWriterWriteString(xmlTextWriterPtr writer,
                               const xmlChar * content);
extern   int   xmlTextWriterWriteVFormatAttribute(xmlTextWriterPtr
writer,
                                         const xmlChar * name,
                                         const char *format,
                                         va_list argptr);
extern  int  xmlTextWriterWriteVFormatAttributeNS(xmlTextWriterPtr
writer,
```



```
                                        const xmlChar * prefix,
                                        const xmlChar * name,
                                        const xmlChar *
                                        namespaceURI,
                                        const char *format,
                                        va_list argptr);
extern int xmlTextWriterWriteVFormatCDATA(xmlTextWriterPtr writer,
                                    const char *format,
                                    va_list argptr);
extern    int    xmlTextWriterWriteVFormatComment(xmlTextWriterPtr
writer,
                                     const char *format,
                                     va_list argptr);
extern int xmlTextWriterWriteVFormatDTD(xmlTextWriterPtr writer,
                                  const xmlChar * name,
                                  const xmlChar * pubid,
                                  const xmlChar * sysid,
                                  const char *format,
                                  va_list argptr);
extern   int   xmlTextWriterWriteVFormatDTDAttlist(xmlTextWriterPtr
writer,
                                        const xmlChar * name,
                                        const char *format,
                                        va_list argptr);
extern   int   xmlTextWriterWriteVFormatDTDElement(xmlTextWriterPtr
writer,
                                        const xmlChar * name,
                                        const char *format,
                                        va_list argptr);
extern                                                        int
xmlTextWriterWriteVFormatDTDInternalEntity(xmlTextWriterPtr
                                             writer, int pe,
                                             const xmlChar * name,
                                             const char *format,
                                             va_list argptr);
extern    int    xmlTextWriterWriteVFormatElement(xmlTextWriterPtr
writer,
                                     const xmlChar * name,
                                     const char *format,
                                     va_list argptr);
extern   int   xmlTextWriterWriteVFormatElementNS(xmlTextWriterPtr
writer,
                                       const xmlChar * prefix,
                                       const xmlChar * name,
                                       const xmlChar * namespaceURI,
                                       const char *format,
                                       va_list argptr);
extern int xmlTextWriterWriteVFormatPI(xmlTextWriterPtr writer,
                                 const xmlChar * target,
                                 const   char   *format,   va_list
argptr);
extern int xmlTextWriterWriteVFormatRaw(xmlTextWriterPtr writer,
                                  const char *format,
                                  va_list argptr);
extern int xmlTextWriterWriteVFormatString(xmlTextWriterPtr writer,
                                     const char *format,
                                     va_list argptr);
```

8.2.37 libxml2/libxml/xpath.h

```
#define xmlXPathNodeSetItem(ns,index)    \
       ((((ns) != NULL) && ((index) >= 0) && ((index) < (ns)-
>nodeNr)) ? \
       (ns)->nodeTab[(index)] : NULL)
#define xmlXPathNodeSetIsEmpty(ns)      \
```



```
       (((ns) == NULL) || ((ns)->nodeNr == 0) || ((ns)->nodeTab ==
NULL))
#define xmlXPathNodeSetGetLength(ns)    ((ns) ? (ns)->nodeNr : 0)
#define XML_XPATH_CHECKNS      (1<<0)
#define XML_XPATH_NOVAR (1<<1)

typedef struct _xmlXPathCompExpr xmlXPathCompExpr;
typedef xmlXPathCompExpr *xmlXPathCompExprPtr;
typedef enum {
    XPATH_UNDEFINED = 0,
    XPATH_NODESET = 1,
    XPATH_BOOLEAN = 2,
    XPATH_NUMBER = 3,
    XPATH_STRING = 4,
    XPATH_POINT = 5,
    XPATH_RANGE = 6,
    XPATH_LOCATIONSET = 7,
    XPATH_USERS = 8,
    XPATH_XSLT_TREE = 9
} xmlXPathObjectType;
typedef struct _xmlNodeSet {
    int nodeNr;
    int nodeMax;
    xmlNodePtr *nodeTab;
} xmlNodeSet;
typedef xmlNodeSet *xmlNodeSetPtr;
typedef struct _xmlXPathObject {
    xmlXPathObjectType type;
    xmlNodeSetPtr nodesetval;
    int boolval;
    double floatval;
    xmlChar *stringval;
    void *user;
    int index;
    void *user2;
    int index2;
} xmlXPathObject;
typedef xmlXPathObject *xmlXPathObjectPtr;
typedef int (*xmlXPathConvertFunc) (xmlXPathObjectPtr, int);
typedef struct _xmlXPathType {
    const xmlChar *name;
    xmlXPathConvertFunc func;
} xmlXPathType;
typedef xmlXPathType *xmlXPathTypePtr;
typedef struct _xmlXPathContext {
    xmlDocPtr doc;
    xmlNodePtr node;
    int nb_variables_unused;
    int max_variables_unused;
    xmlHashTablePtr varHash;
    int nb_types;
    int max_types;
    xmlXPathTypePtr types;
    int nb_funcs_unused;
    int max_funcs_unused;
    xmlHashTablePtr funcHash;
    int nb_axis;
    int max_axis;
    xmlXPathAxisPtr axis;
    xmlNsPtr *namespaces;
    int nsNr;
    void *user;
    int contextSize;
    int proximityPosition;
    int xptr;
    xmlNodePtr here;
```

```
    xmlNodePtr origin;
    xmlHashTablePtr nsHash;
    xmlXPathVariableLookupFunc varLookupFunc;
    void *varLookupData;
    void *extra;
    const xmlChar *function;
    const xmlChar *functionURI;
    xmlXPathFuncLookupFunc funcLookupFunc;
    void *funcLookupData;
    xmlNsPtr *tmpNsList;
    int tmpNsNr;
    void *userData;
    xmlStructuredErrorFunc error;
    xmlError lastError;
    xmlNodePtr debugNode;
    xmlDictPtr dict;
    int flags;
} xmlXPathContext;
typedef xmlXPathContext *xmlXPathContextPtr;
typedef struct _xmlXPathParserContext {
    const xmlChar *cur;
    const xmlChar *base;
    int error;
    xmlXPathContextPtr context;
    xmlXPathObjectPtr value;
    int valueNr;
    int valueMax;
    xmlXPathObjectPtr *valueTab;
    xmlXPathCompExprPtr comp;
    int xptr;
    xmlNodePtr ancestor;
} xmlXPathParserContext;
typedef xmlXPathParserContext *xmlXPathParserContextPtr;
typedef                          xmlXPathObjectPtr(*xmlXPathAxisFunc)
(xmlXPathParserContextPtr,
                                                 xmlXPathObjectPtr);
typedef struct _xmlXPathAxis {
    const xmlChar *name;
    xmlXPathAxisFunc func;
} xmlXPathAxis;
typedef xmlXPathAxis *xmlXPathAxisPtr;
typedef xmlXPathObjectPtr(*xmlXPathVariableLookupFunc) (void *,
                                                        const xmlChar *,
                                                        const xmlChar *);
typedef void (*xmlXPathFunction) (xmlXPathParserContextPtr, int);
typedef xmlXPathFunction(*xmlXPathFuncLookupFunc)  (void  *,  const
xmlChar *,
                                                    const xmlChar *);
typedef enum {
    XPATH_EXPRESSION_OK = 0,
    XPATH_NUMBER_ERROR,
    XPATH_UNFINISHED_LITERAL_ERROR,
    XPATH_START_LITERAL_ERROR,
    XPATH_VARIABLE_REF_ERROR,
    XPATH_UNDEF_VARIABLE_ERROR,
    XPATH_INVALID_PREDICATE_ERROR,
    XPATH_EXPR_ERROR,
    XPATH_UNCLOSED_ERROR,
    XPATH_UNKNOWN_FUNC_ERROR,
    XPATH_INVALID_OPERAND,
    XPATH_INVALID_TYPE,
    XPATH_INVALID_ARITY,
    XPATH_INVALID_CTXT_SIZE,
    XPATH_INVALID_CTXT_POSITION,
    XPATH_MEMORY_ERROR,
    XPTR_SYNTAX_ERROR,
```



```
    XPTR_RESOURCE_ERROR,
    XPTR_SUB_RESOURCE_ERROR,
    XPATH_UNDEF_PREFIX_ERROR,
    XPATH_ENCODING_ERROR,
    XPATH_INVALID_CHAR_ERROR,
    XPATH_INVALID_CTXT
} xmlXPathError;
typedef void (*xmlXPathEvalFunc) (xmlXPathParserContextPtr, int);
typedef struct _xmlXPathFunct {
    const xmlChar *name;
    xmlXPathEvalFunc func;
} xmlXPathFunct;
typedef struct _xmlXPathVariable {
    const xmlChar *name;
    xmlXPathObjectPtr value;
} xmlXPathVariable;
typedef xmlXPathVariable *xmlXPathVariablePtr;
typedef xmlXPathFunct *xmlXPathFuncPtr;
extern double xmlXPathCastBooleanToNumber(int val);
extern xmlChar *xmlXPathCastBooleanToString(int val);
extern int xmlXPathCastNodeSetToBoolean(xmlNodeSetPtr ns);
extern double xmlXPathCastNodeSetToNumber(xmlNodeSetPtr ns);
extern xmlChar *xmlXPathCastNodeSetToString(xmlNodeSetPtr ns);
extern double xmlXPathCastNodeToNumber(xmlNodePtr node);
extern xmlChar *xmlXPathCastNodeToString(xmlNodePtr node);
extern int xmlXPathCastNumberToBoolean(double val);
extern xmlChar *xmlXPathCastNumberToString(double val);
extern int xmlXPathCastStringToBoolean(const xmlChar * val);
extern double xmlXPathCastStringToNumber(const xmlChar * val);
extern int xmlXPathCastToBoolean(xmlXPathObjectPtr val);
extern double xmlXPathCastToNumber(xmlXPathObjectPtr val);
extern xmlChar *xmlXPathCastToString(xmlXPathObjectPtr val);
extern int xmlXPathCmpNodes(xmlNodePtr node1, xmlNodePtr node2);
extern xmlXPathCompExprPtr xmlXPathCompile(const xmlChar * str);
extern xmlXPathObjectPtr xmlXPathCompiledEval(xmlXPathCompExprPtr
comp,
                                              xmlXPathContextPtr ctx);
extern xmlXPathObjectPtr xmlXPathConvertBoolean(xmlXPathObjectPtr
val);
extern xmlXPathObjectPtr xmlXPathConvertNumber(xmlXPathObjectPtr
val);
extern xmlXPathObjectPtr xmlXPathConvertString(xmlXPathObjectPtr
val);
extern xmlXPathCompExprPtr xmlXPathCtxtCompile(xmlXPathContextPtr
ctxt,
                                               const xmlChar * str);
extern xmlXPathObjectPtr xmlXPathEval(const xmlChar * str,
                                      xmlXPathContextPtr ctx);
extern xmlXPathObjectPtr xmlXPathEvalExpression(const xmlChar * str,
                                                xmlXPathContextPtr ctxt);
extern int xmlXPathEvalPredicate(xmlXPathContextPtr ctxt,
                                 xmlXPathObjectPtr res);
extern void xmlXPathFreeCompExpr(xmlXPathCompExprPtr comp);
extern void xmlXPathFreeContext(xmlXPathContextPtr ctxt);
extern void xmlXPathFreeNodeSet(xmlNodeSetPtr obj);
extern void xmlXPathFreeNodeSetList(xmlXPathObjectPtr obj);
extern void xmlXPathFreeObject(xmlXPathObjectPtr obj);
extern void xmlXPathInit(void);
extern int xmlXPathIsInf(double val);
extern int xmlXPathIsNaN(double val);
extern double xmlXPathNAN;
extern double xmlXPathNINF;
extern xmlXPathContextPtr xmlXPathNewContext(xmlDocPtr doc);
extern xmlNodeSetPtr xmlXPathNodeSetCreate(xmlNodePtr val);
extern xmlXPathObjectPtr xmlXPathObjectCopy(xmlXPathObjectPtr val);
extern long int xmlXPathOrderDocElems(xmlDocPtr doc);
```



```
extern double xmlXPathPINF;
```

8.2.38 libxml2/libxml/xpathInternals.h

```
#define xmlXPathStackIsNodeSet(ctxt)     \
       (((ctxt)->value   !=   NULL)   &&   (((ctxt)->value->type   ==
XPATH_NODESET) \
       || ((ctxt)->value->type == XPATH_XSLT_TREE)))
#define xmlXPathStackIsExternal(ctxt)    \
       ((ctxt->value != NULL) && (ctxt->value->type == XPATH_USERS))
#define CAST_TO_BOOLEAN  \
       if   ((ctxt->value   !=   NULL)   &&   (ctxt->value->type   !=
XPATH_BOOLEAN)) \
       xmlXPathBooleanFunction(ctxt, 1);
#define CAST_TO_NUMBER   \
       if   ((ctxt->value   !=   NULL)   &&   (ctxt->value->type   !=
XPATH_NUMBER)) \
       xmlXPathNumberFunction(ctxt, 1);
#define CAST_TO_STRING   \
       if   ((ctxt->value   !=   NULL)   &&   (ctxt->value->type   !=
XPATH_STRING)) \
       xmlXPathStringFunction(ctxt, 1);
#define CHECK_TYPE(typeval)      \
       if ((ctxt->value == NULL) || (ctxt->value->type != typeval))
\
       XP_ERROR(XPATH_INVALID_TYPE)
#define CHECK_TYPE0(typeval)     \
       if ((ctxt->value == NULL) || (ctxt->value->type != typeval))
\
       XP_ERROR0(XPATH_INVALID_TYPE)
#define CHECK_ARITY(x)   \
       if (ctxt == NULL) return; if (nargs != (x)) \
       XP_ERROR(XPATH_INVALID_ARITY);
#define xmlXPathReturnBoolean(ctxt,val)  \
       valuePush((ctxt), xmlXPathNewBoolean(val))
#define xmlXPathReturnEmptyString(ctxt)  \
       valuePush((ctxt), xmlXPathNewCString(""))
#define xmlXPathReturnNumber(ctxt,val)   \
       valuePush((ctxt), xmlXPathNewFloat(val))
#define xmlXPathReturnEmptyNodeSet(ctxt)         \
       valuePush((ctxt), xmlXPathNewNodeSet(NULL))
#define xmlXPathReturnExternal(ctxt,val)         \
       valuePush((ctxt), xmlXPathWrapExternal(val))
#define xmlXPathReturnNodeSet(ctxt,ns)   \
       valuePush((ctxt), xmlXPathWrapNodeSet(ns))
#define xmlXPathReturnString(ctxt,str)   \
       valuePush((ctxt), xmlXPathWrapString(str))
#define xmlXPathSetArityError(ctxt)      \
       xmlXPathSetError((ctxt), XPATH_INVALID_ARITY)
#define xmlXPathSetTypeError(ctxt)       \
       xmlXPathSetError((ctxt), XPATH_INVALID_TYPE)
#define xmlXPathEmptyNodeSet(ns)         \
       { while ((ns)->nodeNr > 0) (ns)->nodeTab[(ns)->nodeNr--] =
NULL; }
#define xmlXPathSetError(ctxt,err)       \
       {  xmlXPatherror((ctxt),  __FILE__,  __LINE__,  (err));  if
((ctxt) != \
       NULL) (ctxt)->error = (err); }
#define xmlXPathGetDocument(ctxt)       ((ctxt)->context->doc)
#define xmlXPathGetContextNode(ctxt)    ((ctxt)->context->node)
#define  xmlXPathCheckError(ctxt)                ((ctxt)->error  !=
XPATH_EXPRESSION_OK)
#define xmlXPathGetError(ctxt)  ((ctxt)->error)
#define  CHECK_ERROR       if  (ctxt->error  !=  XPATH_EXPRESSION_OK)
return
```



```
#define CHECK_ERROR0     if (ctxt->error != XPATH_EXPRESSION_OK)
return(0)
#define                                    xmlXPathReturnFalse(ctxt)
xmlXPathReturnBoolean((ctxt), 0)
#define                                     xmlXPathReturnTrue(ctxt)
xmlXPathReturnBoolean((ctxt), 1)
#define XP_ERROR0(X)    { xmlXPathErr(ctxt, X); return(0); }
#define XP_ERROR(X)     { xmlXPathErr(ctxt, X); return; }

extern xmlXPathObjectPtr valuePop(xmlXPathParserContextPtr ctxt);
extern int valuePush(xmlXPathParserContextPtr ctxt,
                     xmlXPathObjectPtr value);
extern void xmlXPathAddValues(xmlXPathParserContextPtr ctxt);
extern void xmlXPathBooleanFunction(xmlXPathParserContextPtr ctxt,
                                    int nargs);
extern void xmlXPathCeilingFunction(xmlXPathParserContextPtr ctxt,
                                    int nargs);
extern  int  xmlXPathCompareValues(xmlXPathParserContextPtr  ctxt,
int inf,
                                   int strict);
extern void xmlXPathConcatFunction(xmlXPathParserContextPtr ctxt,
                                   int nargs);
extern void xmlXPathContainsFunction(xmlXPathParserContextPtr ctxt,
                                    int nargs);
extern void xmlXPathCountFunction(xmlXPathParserContextPtr ctxt,
                                  int nargs);
extern void xmlXPathDebugDumpCompExpr(FILE * output,
                                      xmlXPathCompExprPtr    comp,    int
depth);
extern     void     xmlXPathDebugDumpObject(FILE     *     output,
xmlXPathObjectPtr cur,
                                   int depth);
extern xmlNodeSetPtr xmlXPathDifference(xmlNodeSetPtr nodes1,
                                        xmlNodeSetPtr nodes2);
extern xmlNodeSetPtr xmlXPathDistinct(xmlNodeSetPtr nodes);
extern xmlNodeSetPtr xmlXPathDistinctSorted(xmlNodeSetPtr nodes);
extern void xmlXPathDivValues(xmlXPathParserContextPtr ctxt);
extern int xmlXPathEqualValues(xmlXPathParserContextPtr ctxt);
extern void xmlXPathErr(xmlXPathParserContextPtr ctxt, int error);
extern void xmlXPathEvalExpr(xmlXPathParserContextPtr ctxt);
extern                                                              int
xmlXPathEvaluatePredicateResult(xmlXPathParserContextPtr ctxt,
                                        xmlXPathObjectPtr res);
extern void xmlXPathFalseFunction(xmlXPathParserContextPtr ctxt,
                                  int nargs);
extern void xmlXPathFloorFunction(xmlXPathParserContextPtr ctxt,
                                  int nargs);
extern   void   xmlXPathFreeParserContext(xmlXPathParserContextPtr
ctxt);
extern  xmlXPathFunction  xmlXPathFunctionLookup(xmlXPathContextPtr
ctxt,
                                         const xmlChar * name);
extern                                              xmlXPathFunction
xmlXPathFunctionLookupNS(xmlXPathContextPtr ctxt,
                                           const xmlChar * name,
                                           const xmlChar * ns_uri);
extern int xmlXPathHasSameNodes(xmlNodeSetPtr nodes1,
                                xmlNodeSetPtr nodes2);
extern void xmlXPathIdFunction(xmlXPathParserContextPtr ctxt, int
nargs);
extern xmlNodeSetPtr xmlXPathIntersection(xmlNodeSetPtr nodes1,
                                          xmlNodeSetPtr nodes2);
extern int xmlXPathIsNodeType(const xmlChar * name);
extern  void  xmlXPathLangFunction(xmlXPathParserContextPtr  ctxt,
int nargs);
```



```
extern  void  xmlXPathLastFunction(xmlXPathParserContextPtr  ctxt,
int nargs);
extern xmlNodeSetPtr xmlXPathLeading(xmlNodeSetPtr nodes1,
                                     xmlNodeSetPtr nodes2);
extern xmlNodeSetPtr xmlXPathLeadingSorted(xmlNodeSetPtr nodes1,
                                           xmlNodeSetPtr nodes2);
extern   void   xmlXPathLocalNameFunction(xmlXPathParserContextPtr
ctxt,
                                     int nargs);
extern void xmlXPathModValues(xmlXPathParserContextPtr ctxt);
extern void xmlXPathMultValues(xmlXPathParserContextPtr ctxt);
extern  void  xmlXPathNamespaceURIFunction(xmlXPathParserContextPtr
ctxt,
                                           int nargs);
extern xmlXPathObjectPtr xmlXPathNewBoolean(int val);
extern xmlXPathObjectPtr xmlXPathNewCString(const char *val);
extern xmlXPathObjectPtr xmlXPathNewFloat(double val);
extern xmlXPathObjectPtr xmlXPathNewNodeSet(xmlNodePtr val);
extern xmlXPathObjectPtr xmlXPathNewNodeSetList(xmlNodeSetPtr val);
extern   xmlXPathParserContextPtr   xmlXPathNewParserContext(const
xmlChar *
                                                             str,
                                                             xmlXPathContextPtr
                                                             ctxt);
extern xmlXPathObjectPtr xmlXPathNewString(const xmlChar * val);
extern xmlXPathObjectPtr xmlXPathNewValueTree(xmlNodePtr val);
extern   xmlNodePtr   xmlXPathNextAncestor(xmlXPathParserContextPtr
ctxt,
                                           xmlNodePtr cur);
extern                                                       xmlNodePtr
xmlXPathNextAncestorOrSelf(xmlXPathParserContextPtr ctxt,
                                            xmlNodePtr cur);
extern  xmlNodePtr  xmlXPathNextAttribute(xmlXPathParserContextPtr
ctxt,
                                          xmlNodePtr cur);
extern xmlNodePtr xmlXPathNextChild(xmlXPathParserContextPtr ctxt,
                                    xmlNodePtr cur);
extern  xmlNodePtr  xmlXPathNextDescendant(xmlXPathParserContextPtr
ctxt,
                                           xmlNodePtr cur);
extern                                                       xmlNodePtr
xmlXPathNextDescendantOrSelf(xmlXPathParserContextPtr
                                                 ctxt, xmlNodePtr cur);
extern  xmlNodePtr  xmlXPathNextFollowing(xmlXPathParserContextPtr
ctxt,
                                          xmlNodePtr cur);
extern                                                       xmlNodePtr
xmlXPathNextFollowingSibling(xmlXPathParserContextPtr
                                                 ctxt, xmlNodePtr cur);
extern  xmlNodePtr  xmlXPathNextNamespace(xmlXPathParserContextPtr
ctxt,
                                          xmlNodePtr cur);
extern xmlNodePtr xmlXPathNextParent(xmlXPathParserContextPtr ctxt,
                                     xmlNodePtr cur);
extern  xmlNodePtr  xmlXPathNextPreceding(xmlXPathParserContextPtr
ctxt,
                                          xmlNodePtr cur);
extern                                                       xmlNodePtr
xmlXPathNextPrecedingSibling(xmlXPathParserContextPtr
                                                 ctxt, xmlNodePtr cur);
extern xmlNodePtr xmlXPathNextSelf(xmlXPathParserContextPtr ctxt,
                                   xmlNodePtr cur);
extern xmlNodeSetPtr xmlXPathNodeLeading(xmlNodeSetPtr nodes,
                                         xmlNodePtr node);
extern xmlNodeSetPtr xmlXPathNodeLeadingSorted(xmlNodeSetPtr nodes,
                                               xmlNodePtr node);
```



```
extern void xmlXPathNodeSetAdd(xmlNodeSetPtr cur, xmlNodePtr val);
extern void xmlXPathNodeSetAddNs(xmlNodeSetPtr cur, xmlNodePtr node,
                                 xmlNsPtr ns);
extern void xmlXPathNodeSetAddUnique(xmlNodeSetPtr cur, xmlNodePtr
val);
extern  int  xmlXPathNodeSetContains(xmlNodeSetPtr  cur,  xmlNodePtr
val);
extern void xmlXPathNodeSetDel(xmlNodeSetPtr cur, xmlNodePtr val);
extern void xmlXPathNodeSetFreeNs(xmlNsPtr ns);
extern xmlNodeSetPtr xmlXPathNodeSetMerge(xmlNodeSetPtr val1,
                                          xmlNodeSetPtr val2);
extern void xmlXPathNodeSetRemove(xmlNodeSetPtr cur, int val);
extern void xmlXPathNodeSetSort(xmlNodeSetPtr set);
extern xmlNodeSetPtr xmlXPathNodeTrailing(xmlNodeSetPtr nodes,
                                          xmlNodePtr node);
extern    xmlNodeSetPtr    xmlXPathNodeTrailingSorted(xmlNodeSetPtr
nodes,
                                                 xmlNodePtr node);
extern   void   xmlXPathNormalizeFunction(xmlXPathParserContextPtr
ctxt,
                                       int nargs);
extern int xmlXPathNotEqualValues(xmlXPathParserContextPtr ctxt);
extern void xmlXPathNotFunction(xmlXPathParserContextPtr ctxt, int
nargs);
extern const xmlChar *xmlXPathNsLookup(xmlXPathContextPtr ctxt,
                                       const xmlChar * prefix);
extern void xmlXPathNumberFunction(xmlXPathParserContextPtr ctxt,
                                   int nargs);
extern xmlChar *xmlXPathParseNCName(xmlXPathParserContextPtr ctxt);
extern xmlChar *xmlXPathParseName(xmlXPathParserContextPtr ctxt);
extern int xmlXPathPopBoolean(xmlXPathParserContextPtr ctxt);
extern void *xmlXPathPopExternal(xmlXPathParserContextPtr ctxt);
extern  xmlNodeSetPtr  xmlXPathPopNodeSet(xmlXPathParserContextPtr
ctxt);
extern double xmlXPathPopNumber(xmlXPathParserContextPtr ctxt);
extern xmlChar *xmlXPathPopString(xmlXPathParserContextPtr ctxt);
extern void xmlXPathPositionFunction(xmlXPathParserContextPtr ctxt,
                                     int nargs);
extern void xmlXPathRegisterAllFunctions(xmlXPathContextPtr ctxt);
extern int xmlXPathRegisterFunc(xmlXPathContextPtr ctxt,
                                const  xmlChar  *  name,  xmlXPathFunction
f);
extern void xmlXPathRegisterFuncLookup(xmlXPathContextPtr ctxt,
                                       xmlXPathFuncLookupFunc f,
                                       void *funcCtxt);
extern int xmlXPathRegisterFuncNS(xmlXPathContextPtr ctxt,
                                  const xmlChar * name,
                                  const xmlChar * ns_uri,
                                  xmlXPathFunction f);
extern int xmlXPathRegisterNs(xmlXPathContextPtr ctxt,
                              const xmlChar * prefix,
                              const xmlChar * ns_uri);
extern int xmlXPathRegisterVariable(xmlXPathContextPtr ctxt,
                                    const xmlChar * name,
                                    xmlXPathObjectPtr value);
extern void xmlXPathRegisterVariableLookup(xmlXPathContextPtr ctxt,
                                           xmlXPathVariableLookupFunc f,
                                           void *data);
extern int xmlXPathRegisterVariableNS(xmlXPathContextPtr ctxt,
                                      const xmlChar * name,
                                      const xmlChar * ns_uri,
                                      xmlXPathObjectPtr value);
extern    void    xmlXPathRegisteredFuncsCleanup(xmlXPathContextPtr
ctxt);
extern void xmlXPathRegisteredNsCleanup(xmlXPathContextPtr ctxt);
```



```
extern void xmlXPathRegisteredVariablesCleanup(xmlXPathContextPtr
ctxt);
extern void xmlXPathRoot(xmlXPathParserContextPtr ctxt);
extern void xmlXPathRoundFunction(xmlXPathParserContextPtr ctxt,
                                  int nargs);
extern void xmlXPathStartsWithFunction(xmlXPathParserContextPtr
ctxt,
                                       int nargs);
extern double xmlXPathStringEvalNumber(const xmlChar * str);
extern void xmlXPathStringFunction(xmlXPathParserContextPtr ctxt,
                                   int nargs);
extern void xmlXPathStringLengthFunction(xmlXPathParserContextPtr
ctxt,
                                         int nargs);
extern void xmlXPathSubValues(xmlXPathParserContextPtr ctxt);
extern void
xmlXPathSubstringAfterFunction(xmlXPathParserContextPtr ctxt,
                               int nargs);
extern void
xmlXPathSubstringBeforeFunction(xmlXPathParserContextPtr ctxt,
                                int nargs);
extern void xmlXPathSubstringFunction(xmlXPathParserContextPtr
ctxt,
                                      int nargs);
extern void xmlXPathSumFunction(xmlXPathParserContextPtr ctxt, int
nargs);
extern xmlNodeSetPtr xmlXPathTrailing(xmlNodeSetPtr nodes1,
                                      xmlNodeSetPtr nodes2);
extern xmlNodeSetPtr xmlXPathTrailingSorted(xmlNodeSetPtr nodes1,
                                            xmlNodeSetPtr nodes2);
extern void xmlXPathTranslateFunction(xmlXPathParserContextPtr
ctxt,
                                      int nargs);
extern void xmlXPathTrueFunction(xmlXPathParserContextPtr ctxt,
int nargs);
extern void xmlXPathValueFlipSign(xmlXPathParserContextPtr ctxt);
extern xmlXPathObjectPtr xmlXPathVariableLookup(xmlXPathContextPtr
ctxt,
                                                const xmlChar * name);
extern xmlXPathObjectPtr
xmlXPathVariableLookupNS(xmlXPathContextPtr ctxt,
                                                  const xmlChar * name,
                                                  const xmlChar * ns_uri);
extern xmlXPathObjectPtr xmlXPathWrapCString(char *val);
extern xmlXPathObjectPtr xmlXPathWrapExternal(void *val);
extern xmlXPathObjectPtr xmlXPathWrapNodeSet(xmlNodeSetPtr val);
extern xmlXPathObjectPtr xmlXPathWrapString(xmlChar * val);
extern void xmlXPatherror(xmlXPathParserContextPtr ctxt, const char
*file,
                          int line, int no);
```

8.2.39 libxml2/libxml/xpointer.h

```
typedef struct _xmlLocationSet {
    int locNr;
    int locMax;
    xmlXPathObjectPtr *locTab;
} xmlLocationSet;
typedef xmlLocationSet *xmlLocationSetPtr;
extern xmlNodePtr xmlXPtrBuildNodeList(xmlXPathObjectPtr obj);
extern xmlXPathObjectPtr xmlXPtrEval(const xmlChar * str,
                                     xmlXPathContextPtr ctx);
extern void xmlXPtrEvalRangePredicate(xmlXPathParserContextPtr
ctxt);
extern void xmlXPtrFreeLocationSet(xmlLocationSetPtr obj);
```



```
extern void xmlXPtrLocationSetAdd(xmlLocationSetPtr cur,
                                  xmlXPathObjectPtr val);
extern                                              xmlLocationSetPtr
xmlXPtrLocationSetCreate(xmlXPathObjectPtr val);
extern void xmlXPtrLocationSetDel(xmlLocationSetPtr cur,
                                  xmlXPathObjectPtr val);
extern xmlLocationSetPtr xmlXPtrLocationSetMerge(xmlLocationSetPtr
val1,
                                                 xmlLocationSetPtr val2);
extern  void  xmlXPtrLocationSetRemove(xmlLocationSetPtr  cur,  int
val);
extern    xmlXPathObjectPtr    xmlXPtrNewCollapsedRange(xmlNodePtr
start);
extern    xmlXPathContextPtr    xmlXPtrNewContext(xmlDocPtr    doc,
xmlNodePtr here,
                                            xmlNodePtr origin);
extern                                              xmlXPathObjectPtr
xmlXPtrNewLocationSetNodeSet(xmlNodeSetPtr set);
extern   xmlXPathObjectPtr   xmlXPtrNewLocationSetNodes(xmlNodePtr
start,
                                                 xmlNodePtr end);
extern  xmlXPathObjectPtr  xmlXPtrNewRange(xmlNodePtr  start,  int
startindex,
                                       xmlNodePtr end, int endindex);
extern    xmlXPathObjectPtr    xmlXPtrNewRangeNodeObject(xmlNodePtr
start,
                                                 xmlXPathObjectPtr end);
extern xmlXPathObjectPtr xmlXPtrNewRangeNodePoint(xmlNodePtr start,
                                                  xmlXPathObjectPtr end);
extern xmlXPathObjectPtr xmlXPtrNewRangeNodes(xmlNodePtr start,
                                              xmlNodePtr end);
extern                                              xmlXPathObjectPtr
xmlXPtrNewRangePointNode(xmlXPathObjectPtr start,
                                               xmlNodePtr end);
extern  xmlXPathObjectPtr  xmlXPtrNewRangePoints(xmlXPathObjectPtr
start,
                                             xmlXPathObjectPtr end);
extern void xmlXPtrRangeToFunction(xmlXPathParserContextPtr ctxt,
                                   int nargs);
extern  xmlXPathObjectPtr  xmlXPtrWrapLocationSet(xmlLocationSetPtr
val);
```

V XSLT library

9 Libraries

9.1 Interfaces for libxslt

Table 9-1 defines the library name and shared object name for the libxslt library

Table 9-1 libxslt Definition

Library:	libxslt
SONAME:	libxslt.so.1

The behavior of the interfaces in this library is specified by the following specifications:

[libxslt] Reference Manual for libxslt

9.1.1 libxslt interfaces

9.1.1.1 Interfaces for libxslt interfaces

An LSB conforming implementation shall provide the generic functions for libxslt interfaces specified in Table 9-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-2 libxslt - libxslt interfaces Function Interfaces

xslAddCall(LIBXML2_1.0.11) [libxslt]	xslDropCall(LIBXML2_1.0.11) [libxslt]	xsltAddKey(LIBXML2_1.0.11) [libxslt]
xsltAddStackElemList(LIBXML2_1.0.11) [libxslt]	xsltAddTemplate(LIBXML2_1.0.11) [libxslt]	xsltAllocateExtra(LIBXML2_1.0.12) [libxslt]
xsltAllocateExtraCtxt(LIBXML2_1.0.12) [libxslt]	xsltApplyAttributeSet(LIBXML2_1.0.11) [libxslt]	xsltApplyImports(LIBXML2_1.0.11) [libxslt]
xsltApplyOneTemplate(LIBXML2_1.0.11) [libxslt]	xsltApplyStripSpaces(LIBXML2_1.0.11) [libxslt]	xsltApplyStylesheet(LIBXML2_1.0.11) [libxslt]
xsltApplyStylesheetUser(LIBXML2_1.0.11) [libxslt]	xsltApplyTemplates(LIBXML2_1.0.11) [libxslt]	xsltAttrListTemplateProcess(LIBXML2_1.0.11) [libxslt]
xsltAttrTemplateProcess(LIBXML2_1.0.11) [libxslt]	xsltAttrTemplateValueProcess(LIBXML2_1.0.11) [libxslt]	xsltAttrTemplateValueProcessNode(LIBXML2_1.0.22) [libxslt]
xsltAttribute(LIBXML2_1.0.11) [libxslt]	xsltCalibrateAdjust(LIBXML2_1.0.11) [libxslt]	xsltCallTemplate(LIBXML2_1.0.11) [libxslt]
xsltCheckExtPrefix(LIBXML2_1.0.11) [libxslt]	xsltCheckExtURI(LIBXML2_1.1.24) [libxslt]	xsltCheckRead(LIBXML2_1.0.22) [libxslt]
xsltCheckWrite(LIBXML2_1.0.22) [libxslt]	xsltChoose(LIBXML2_1.0.11) [libxslt]	xsltCleanupGlobals(LIBXML2_1.0.11) [libxslt]
xsltCleanupTemplates(LIBXML2_1.0.11) [libxslt]	xsltComment(LIBXML2_1.0.11) [libxslt]	xsltCompileAttr(LIBXML2_1.1.3) [libxslt]

xsltCompilePattern(LIBXML2_1.0.11) [libxslt]	xsltComputeSortResult(LIBXML2_1.0.24) [libxslt]	xsltCopy(LIBXML2_1.0.11) [libxslt]
xsltCopyNamespace(LIBXML2_1.0.11) [libxslt]	xsltCopyNamespaceList(LIBXML2_1.0.11) [libxslt]	xsltCopyOf(LIBXML2_1.0.11) [libxslt]
xsltCopyTextString(LIBXML2_1.0.32) [libxslt]	xsltCreateRVT(LIBXML2_1.0.30) [libxslt]	xsltDebug(LIBXML2_1.0.11) [libxslt]
xsltDebugDumpExtensions(LIBXML2_1.0.18) [libxslt]	xsltDebugGetDefaultTrace(LIBXML2_1.1.1) [libxslt]	xsltDebugSetDefaultTrace(LIBXML2_1.1.1) [libxslt]
xsltDecimalFormatGetByName(LIBXML2_1.0.11) [libxslt]	xsltDefaultSortFunction(LIBXML2_1.0.24) [libxslt]	xsltDoSortFunction(LIBXML2_1.0.11) [libxslt]
xsltDocumentComp(LIBXML2_1.0.11) [libxslt]	xsltDocumentElem(LIBXML2_1.0.11) [libxslt]	xsltDocumentFunction(LIBXML2_1.0.11) [libxslt]
xsltDocumentSortFunction(LIBXML2_1.0.11) [libxslt]	xsltElement(LIBXML2_1.0.11) [libxslt]	xsltElementAvailableFunction(LIBXML2_1.0.11) [libxslt]
xsltEvalAVT(LIBXML2_1.1.3) [libxslt]	xsltEvalAttrValueTemplate(LIBXML2_1.0.11) [libxslt]	xsltEvalGlobalVariables(LIBXML2_1.0.11) [libxslt]
xsltEvalOneUserParam(LIBXML2_1.0.11) [libxslt]	xsltEvalStaticAttrValueTemplate(LIBXML2_1.0.11) [libxslt]	xsltEvalTemplateString(LIBXML2_1.0.11) [libxslt]
xsltEvalUserParams(LIBXML2_1.0.11) [libxslt]	xsltEvalXPathPredicate(LIBXML2_1.0.11) [libxslt]	xsltEvalXPathString(LIBXML2_1.0.11) [libxslt]
xsltEvalXPathStringNs(LIBXML2_1.0.22) [libxslt]	xsltExtElementLookup(LIBXML2_1.0.11) [libxslt]	xsltExtModuleElementLookup(LIBXML2_1.0.11) [libxslt]
xsltExtModuleElementPreComputeLookup(LIBXML2_1.0.13) [libxslt]	xsltExtModuleFunctionLookup(LIBXML2_1.0.11) [libxslt]	xsltExtModuleTopLevelLookup(LIBXML2_1.0.11) [libxslt]
xsltExtensionInstructionResultFinalize(LIBXML2_1.1.18) [libxslt]	xsltExtensionInstructionResultRegister(LIBXML2_1.1.18) [libxslt]	xsltFindDocument(LIBXML2_1.0.11) [libxslt]
xsltFindElemSpaceHandling(LIBXML2_1.0.11) [libxslt]	xsltFindTemplate(LIBXML2_1.0.11) [libxslt]	xsltForEach(LIBXML2_1.0.11) [libxslt]
xsltFormatNumberConversion(LIBXML2_1.0.11) [libxslt]	xsltFormatNumberFunction(LIBXML2_1.0.11) [libxslt]	xsltFreeAVTList(LIBXML2_1.1.3) [libxslt]

xsltFreeAttributeSetsHashes(LIBXML2_1.0.11) [libxslt]	xsltFreeCompMatchList(LIBXML2_1.0.11) [libxslt]	xsltFreeCtxtExts(LIBXML2_1.0.11) [libxslt]
xsltFreeDocumentKeys(LIBXML2_1.0.11) [libxslt]	xsltFreeDocuments(LIBXML2_1.0.11) [libxslt]	xsltFreeExts(LIBXML2_1.0.11) [libxslt]
xsltFreeGlobalVariables(LIBXML2_1.0.11) [libxslt]	xsltFreeKeys(LIBXML2_1.0.11) [libxslt]	xsltFreeLocale(LIBXML2_1.1.25) [libxslt]
xsltFreeNamespaceAliasHashes(LIBXML2_1.0.11) [libxslt]	xsltFreeRVTs(LIBXML2_1.0.30) [libxslt]	xsltFreeSecurityPrefs(LIBXML2_1.0.22) [libxslt]
xsltFreeStackElemList(LIBXML2_1.0.11) [libxslt]	xsltFreeStyleDocuments(LIBXML2_1.0.11) [libxslt]	xsltFreeStylePreComps(LIBXML2_1.0.11) [libxslt]
xsltFreeStylesheet(LIBXML2_1.0.11) [libxslt]	xsltFreeTemplateHashes(LIBXML2_1.0.11) [libxslt]	xsltFreeTransformContext(LIBXML2_1.0.11) [libxslt]
xsltFunctionAvailableFunction(LIBXML2_1.0.11) [libxslt]	xsltFunctionNodeSet(LIBXML2_1.0.11) [libxslt]	xsltGenerateIdFunction(LIBXML2_1.0.11) [libxslt]
xsltGetCNsProp(LIBXML2_1.1.3) [libxslt]	xsltGetDebuggerStatus(LIBXML2_1.1.0) [libxslt]	xsltGetDefaultSecurityPrefs(LIBXML2_1.0.22) [libxslt]
xsltGetExtData(LIBXML2_1.0.11) [libxslt]	xsltGetExtInfo(LIBXML2_1.0.32) [libxslt]	xsltGetKey(LIBXML2_1.0.11) [libxslt]
xsltGetNamespace(LIBXML2_1.0.11) [libxslt]	xsltGetNsProp(LIBXML2_1.0.11) [libxslt]	xsltGetPlainNamespace(LIBXML2_1.1.7) [libxslt]
xsltGetProfileInformation(LIBXML2_1.0.24) [libxslt]	xsltGetQNameURI(LIBXML2_1.0.11) [libxslt]	xsltGetQNameURI2(LIBXML2_1.1.5) [libxslt]
xsltGetSecurityPrefs(LIBXML2_1.0.22) [libxslt]	xsltGetSpecialNamespace(LIBXML2_1.0.11) [libxslt]	xsltGetTemplate(LIBXML2_1.0.11) [libxslt]
xsltGetUTF8Char(LIBXML2_1.0.24) [libxslt]	xsltGetXIncludeDefault(LIBXML2_1.0.11) [libxslt]	xsltIf(LIBXML2_1.0.11) [libxslt]
xsltInit(LIBXML2_1.1.18) [libxslt]	xsltInitAllDocKeys(LIBXML2_1.1.23) [libxslt]	xsltInitCtxtExts(LIBXML2_1.0.11) [libxslt]
xsltInitCtxtKey(LIBXML2_1.1.18) [libxslt]	xsltInitCtxtKeys(LIBXML2_1.0.11) [libxslt]	xsltInitElemPreComp(LIBXML2_1.0.11) [libxslt]
xsltInitGlobals(LIBXML2_1.1.25) [libxslt]	xsltIsBlank(LIBXML2_1.0.11) [libxslt]	xsltKeyFunction(LIBXML2_1.0.11) [libxslt]

xsltLoadDocument(LIBXML2_1.0.11) [libxslt]	xsltLoadStyleDocument(LIBXML2_1.0.11) [libxslt]	xsltLoadStylesheetPI(LIBXML2_1.0.11) [libxslt]
xsltLocalVariablePop(LIBXML2_1.1.20) [libxslt]	xsltLocalVariablePush(LIBXML2_1.1.20) [libxslt]	xsltLocaleStrcmp(LIBXML2_1.1.25) [libxslt]
xsltMessage(LIBXML2_1.0.11) [libxslt]	xsltNamespaceAlias(LIBXML2_1.0.11) [libxslt]	xsltNeedElemSpaceHandling(LIBXML2_1.0.11) [libxslt]
xsltNewDocument(LIBXML2_1.0.11) [libxslt]	xsltNewElemPreComp(LIBXML2_1.0.11) [libxslt]	xsltNewLocale(LIBXML2_1.1.25) [libxslt]
xsltNewSecurityPrefs(LIBXML2_1.0.22) [libxslt]	xsltNewStyleDocument(LIBXML2_1.0.11) [libxslt]	xsltNewStylesheet(LIBXML2_1.0.11) [libxslt]
xsltNewTransformContext(LIBXML2_1.0.11) [libxslt]	xsltNextImport(LIBXML2_1.0.11) [libxslt]	xsltNormalizeCompSteps(LIBXML2_1.0.33) [libxslt]
xsltNumber(LIBXML2_1.0.11) [libxslt]	xsltNumberFormat(LIBXML2_1.0.11) [libxslt]	xsltParseGlobalParam(LIBXML2_1.0.11) [libxslt]
xsltParseGlobalVariable(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetAttributeSet(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetCallerParam(LIBXML2_1.0.11) [libxslt]
xsltParseStylesheetDoc(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetFile(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetImport(LIBXML2_1.0.11) [libxslt]
xsltParseStylesheetImportedDoc(LIBXML2_1.0.24) [libxslt]	xsltParseStylesheetInclude(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetOutput(LIBXML2_1.0.11) [libxslt]
xsltParseStylesheetParam(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetProcess(LIBXML2_1.0.11) [libxslt]	xsltParseStylesheetVariable(LIBXML2_1.0.11) [libxslt]
xsltParseTemplateContent(LIBXML2_1.0.11) [libxslt]	xsltPreComputeExtModuleElement(LIBXML2_1.0.11) [libxslt]	xsltPrintErrorContext(LIBXML2_1.0.11) [libxslt]
xsltProcessOneNode(LIBXML2_1.1.26) [libxslt]	xsltProcessingInstruction(LIBXML2_1.0.11) [libxslt]	xsltProfileStylesheet(LIBXML2_1.0.11) [libxslt]
xsltQuoteOneUserParam(LIBXML2_1.0.11) [libxslt]	xsltQuoteUserParams(LIBXML2_1.0.11) [libxslt]	xsltRegisterAllElement(LIBXML2_1.0.11) [libxslt]
xsltRegisterAllExtras(LIBXML2_1.0.11) [libxslt]	xsltRegisterAllFunctions(LIBXML2_1.0.11) [libxslt]	xsltRegisterExtElement(LIBXML2_1.0.11) [libxslt]

xsltRegisterExtFunction (LIBXML2_1.0.11) [libxslt]	xsltRegisterExtModule(LIBXML2_1.0.11) [libxslt]	xsltRegisterExtModuleE lement(LIBXML2_1.0.11) [libxslt]
xsltRegisterExtModuleF ull(LIBXML2_1.0.11) [libxslt]	xsltRegisterExtModuleF unction(LIBXML2_1.0.1 1) [libxslt]	xsltRegisterExtModuleT opLevel(LIBXML2_1.0.1 1) [libxslt]
xsltRegisterExtPrefix(LI BXML2_1.0.11) [libxslt]	xsltRegisterExtras(LIBX ML2_1.0.11) [libxslt]	xsltRegisterLocalRVT(L IBXML2_1.1.18) [libxslt]
xsltRegisterPersistRVT(LIBXML2_1.0.30) [libxslt]	xsltRegisterTestModule (LIBXML2_1.0.11) [libxslt]	xsltRegisterTmpRVT(LI BXML2_1.0.30) [libxslt]
xsltReleaseRVT(LIBXM L2_1.1.18) [libxslt]	xsltResolveStylesheetAt tributeSet(LIBXML2_1.0 .16) [libxslt]	xsltRunStylesheet(LIBX ML2_1.0.11) [libxslt]
xsltRunStylesheetUser(LIBXML2_1.0.17) [libxslt]	xsltSaveProfiling(LIBX ML2_1.0.11) [libxslt]	xsltSaveResultTo(LIBX ML2_1.0.11) [libxslt]
xsltSaveResultToFd(LIB XML2_1.0.11) [libxslt]	xsltSaveResultToFile(LI BXML2_1.0.11) [libxslt]	xsltSaveResultToFilena me(LIBXML2_1.0.11) [libxslt]
xsltSaveResultToString(LIBXML2_1.0.18) [libxslt]	xsltSecurityAllow(LIBX ML2_1.0.22) [libxslt]	xsltSecurityForbid(LIBX ML2_1.0.22) [libxslt]
xsltSetCtxtParseOptions (LIBXML2_1.1.2) [libxslt]	xsltSetCtxtSecurityPrefs (LIBXML2_1.0.22) [libxslt]	xsltSetCtxtSortFunc(LIB XML2_1.0.24) [libxslt]
xsltSetDebuggerCallbac ks(LIBXML2_1.0.11) [libxslt]	xsltSetDebuggerStatus(LIBXML2_1.1.0) [libxslt]	xsltSetDefaultSecurityP refs(LIBXML2_1.0.22) [libxslt]
xsltSetGenericDebugFu nc(LIBXML2_1.0.11) [libxslt]	xsltSetGenericErrorFun c(LIBXML2_1.0.11) [libxslt]	xsltSetLoaderFunc(LIB XML2_1.1.9) [libxslt]
xsltSetSecurityPrefs(LIB XML2_1.0.22) [libxslt]	xsltSetSortFunc(LIBXM L2_1.0.24) [libxslt]	xsltSetTransformErrorF unc(LIBXML2_1.0.22) [libxslt]
xsltSetXIncludeDefault(LIBXML2_1.0.11) [libxslt]	xsltShutdownCtxtExts(LIBXML2_1.0.11) [libxslt]	xsltShutdownExts(LIBX ML2_1.0.11) [libxslt]
xsltSort(LIBXML2_1.0.1 1) [libxslt]	xsltSplitQName(LIBXM L2_1.1.3) [libxslt]	xsltStrxfrm(LIBXML2_1 .1.25) [libxslt]
xsltStyleGetExtData(LI BXML2_1.0.11) [libxslt]	xsltStylePreCompute(LI BXML2_1.0.11) [libxslt]	xsltSystemPropertyFun ction(LIBXML2_1.0.11) [libxslt]

xsltTemplateProcess(LIBXML2_1.0.11) [libxslt]	xsltTestCompMatchList(LIBXML2_1.0.11) [libxslt]	xsltText(LIBXML2_1.0.11) [libxslt]
xsltTimestamp(LIBXML2_1.0.11) [libxslt]	xsltTransformError(LIBXML2_1.0.22) [libxslt]	xsltUninit(LIBXML2_1.1.18) [libxslt]
xsltUnparsedEntityURIFunction(LIBXML2_1.0.11) [libxslt]	xsltUnregisterExtModule(LIBXML2_1.0.11) [libxslt]	xsltUnregisterExtModuleElement(LIBXML2_1.0.11) [libxslt]
xsltUnregisterExtModuleFunction(LIBXML2_1.0.11) [libxslt]	xsltUnregisterExtModuleTopLevel(LIBXML2_1.0.11) [libxslt]	xsltValueOf(LIBXML2_1.0.11) [libxslt]
xsltVariableLookup(LIBXML2_1.0.11) [libxslt]	xsltXPathCompile(LIBXML2_1.1.3) [libxslt]	xsltXPathFunctionLookup(LIBXML2_1.0.11) [libxslt]
xsltXPathGetTransformContext(LIBXML2_1.0.13) [libxslt]	xsltXPathVariableLookup(LIBXML2_1.0.11) [libxslt]	

9.2 Data Definitions for libxslt

This section defines global identifiers and their values that are associated with interfaces contained in libxslt. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

9.2.1 libxslt/attributes.h

```
extern void xsltApplyAttributeSet(xsltTransformContextPtr ctxt,
                                  xmlNodePtr node, xmlNodePtr inst,
                                  const unsigned char *attributes);
extern void xsltFreeAttributeSetsHashes(xsltStylesheetPtr style);
extern    void    xsltParseStylesheetAttributeSet(xsltStylesheetPtr
style,
                                          xmlNodePtr cur);
extern   void   xsltResolveStylesheetAttributeSet(xsltStylesheetPtr
style);
```

9.2.2 libxslt/documents.h

```
typedef enum {
    XSLT_LOAD_START,
```



```
    XSLT_LOAD_STYLESHEET,
    XSLT_LOAD_DOCUMENT
} xsltLoadType;
typedef xmlDocPtr(*xsltDocLoaderFunc) (void);
extern xsltDocLoaderFunc xsltDocDefaultLoader;
extern   xsltDocumentPtr   xsltFindDocument(xsltTransformContextPtr
ctxt,
                                xmlDocPtr doc);
extern void xsltFreeDocuments(xsltTransformContextPtr ctxt);
extern void xsltFreeStyleDocuments(xsltStylesheetPtr style);
extern   xsltDocumentPtr   xsltLoadDocument(xsltTransformContextPtr
ctxt,
                                const unsigned char *URI);
extern   xsltDocumentPtr   xsltLoadStyleDocument(xsltStylesheetPtr
style,
                                    const unsigned char *URI);
extern   xsltDocumentPtr   xsltNewDocument(xsltTransformContextPtr
ctxt,
                                xmlDocPtr doc);
extern    xsltDocumentPtr    xsltNewStyleDocument(xsltStylesheetPtr
style,
                                    xmlDocPtr doc);
extern void xsltSetLoaderFunc(xsltDocLoaderFunc f);
```

9.2.3 libxslt/extensions.h

```
typedef void *(*xsltStyleExtInitFunction) (void);
typedef void (*xsltStyleExtShutdownFunction) (void);
typedef void *(*xsltExtInitFunction) (void);
typedef void (*xsltExtShutdownFunction) (void);
typedef xsltElemPreCompPtr(*xsltPreComputeFunction) (void);
typedef void (*xsltTopLevelFunction) (void);
extern int xsltCheckExtPrefix(xsltStylesheetPtr style,
                        const unsigned char *URI);
extern int xsltCheckExtURI(xsltStylesheetPtr style,
                      const unsigned char *URI);
extern void xsltDebugDumpExtensions(FILE * output);
extern                                  xsltTransformFunction
xsltExtElementLookup(xsltTransformContextPtr
                                        ctxt,
                                        const unsigned char
                                        *name,
                                        const unsigned char
                                        *URI);
extern xsltTransformFunction    xsltExtModuleElementLookup(const
unsigned char
                                            *name,
                                            const unsigned char
                                            *URI);
extern                                  xsltPreComputeFunction
xsltExtModuleElementPreComputeLookup(const
                                                    unsigned
                                                    char
                                                    *name,
                                                    const
                                                    unsigned
                                                    char
                                                    *URI);
extern xmlXPathFunction xsltExtModuleFunctionLookup(const unsigned
char
                                        *name,
                                        const unsigned char
                                        *URI);
extern    xsltTopLevelFunction    xsltExtModuleTopLevelLookup(const
unsigned char
```

```
                                                    *name,
                                                    const unsigned char
                                                    *URI);
extern void xsltFreeCtxtExts(xsltTransformContextPtr ctxt);
extern void xsltFreeExts(xsltStylesheetPtr style);
extern void *xsltGetExtData(xsltTransformContextPtr ctxt,
                            const unsigned char *URI);
extern xmlHashTablePtr xsltGetExtInfo(xsltStylesheetPtr style,
                                      const unsigned char *URI);
extern int xsltInitCtxtExts(xsltTransformContextPtr ctxt);
extern void xsltInitElemPreComp(xsltElemPreCompPtr comp,
                                xsltStylesheetPtr   style,   xmlNodePtr
inst,
                                xsltTransformFunction function,
                                xsltElemPreCompDeallocator freeFunc);
extern void xsltInitGlobals(void);
extern   xsltElemPreCompPtr   xsltNewElemPreComp(xsltStylesheetPtr
style,
                                         xmlNodePtr inst,
                                         xsltTransformFunction
                                         function);
extern                                            xsltElemPreCompPtr
xsltPreComputeExtModuleElement(xsltStylesheetPtr
                                                    style,
                                                    xmlNodePtr inst);
extern int xsltRegisterExtElement(xsltTransformContextPtr ctxt,
                                  const unsigned char *name,
                                  const unsigned char *URI,
                                  xsltTransformFunction function);
extern int xsltRegisterExtFunction(xsltTransformContextPtr ctxt,
                                   const unsigned char *name,
                                   const unsigned char *URI,
                                   xmlXPathFunction function);
extern int xsltRegisterExtModule(const unsigned char *URI,
                                 xsltExtInitFunction initFunc,
                                 xsltExtShutdownFunction shutdownFunc);
extern int xsltRegisterExtModuleElement(const unsigned char *name,
                                        const unsigned char *URI,
                                        xsltPreComputeFunction precomp,
                                        xsltTransformFunction  transform);
extern int xsltRegisterExtModuleFull(const unsigned char *URI,
                                     xsltExtInitFunction initFunc,
                                     xsltExtShutdownFunction
shutdownFunc,
                                     xsltStyleExtInitFunction
                                     styleInitFunc,
                                     xsltStyleExtShutdownFunction
                                     styleShutdownFunc);
extern int xsltRegisterExtModuleFunction(const unsigned char *name,
                                         const unsigned char *URI,
                                         xmlXPathFunction function);
extern int xsltRegisterExtModuleTopLevel(const unsigned char *name,
                                         const unsigned char *URI,
                                         xsltTopLevelFunction function);
extern int xsltRegisterExtPrefix(xsltStylesheetPtr style,
                                 const unsigned char *prefix,
                                 const unsigned char *URI);
extern void xsltRegisterTestModule(void);
extern void xsltShutdownCtxtExts(xsltTransformContextPtr ctxt);
extern void xsltShutdownExts(xsltStylesheetPtr style);
extern void *xsltStyleGetExtData(xsltStylesheetPtr style,
                                 const unsigned char *URI);
extern int xsltUnregisterExtModule(const unsigned char *URI);
extern int xsltUnregisterExtModuleElement(const unsigned char *name,
                                          const unsigned char *URI);
```



```
extern int xsltUnregisterExtModuleFunction(const unsigned char
*name,
                                           const unsigned char *URI);
extern int xsltUnregisterExtModuleTopLevel(const unsigned char
*name,
                                           const unsigned char *URI);
extern xsltTransformContextPtr
xsltXPathGetTransformContext(xmlXPathParserContextPtr ctxt);
```

9.2.4 libxslt/extra.h

```
#define XSLT_SAXON_NAMESPACE    ((xmlChar *) "http://icl.com/saxon")
#define   XSLT_NORM_SAXON_NAMESPACE                    ((xmlChar   *)
"http://nwalsh.com/xslt/ext/com.nwalsh.saxon.CVS")
#define    XSLT_XT_NAMESPACE                            ((xmlChar    *)
"http://www.jclark.com/xt")
#define       XSLT_LIBXSLT_NAMESPACE              ((xmlChar     *)
"http://xmlsoft.org/XSLT/namespace")
#define     XSLT_XALAN_NAMESPACE                     ((xmlChar     *)
"org.apache.xalan.xslt.extensions.Redirect")

extern void xsltDebug(xsltTransformContextPtr ctxt, xmlNodePtr node,
                      xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltFunctionNodeSet(xmlXPathParserContextPtr ctxt, int
nargs);
extern void xsltRegisterAllExtras(void);
extern void xsltRegisterExtras(xsltTransformContextPtr ctxt);
```

9.2.5 libxslt/functions.h

```
#define                            XSLT_REGISTER_FUNCTION_LOOKUP(ctxt)
xmlXPathRegisterFuncLookup((ctxt)->xpathCtxt,
(xmlXPathFuncLookupFunc)  xsltXPathFunctionLookup,  (void  *)(ctxt-
>xpathCtxt));

extern  void  xsltDocumentFunction(xmlXPathParserContextPtr  ctxt,
int nargs);
extern void xsltElementAvailableFunction(xmlXPathParserContextPtr
ctxt,
                                         int nargs);
extern void xsltFormatNumberFunction(xmlXPathParserContextPtr ctxt,
                                     int nargs);
extern void xsltFunctionAvailableFunction(xmlXPathParserContextPtr
ctxt,
                                          int nargs);
extern void xsltGenerateIdFunction(xmlXPathParserContextPtr ctxt,
                                   int nargs);
extern  void  xsltKeyFunction(xmlXPathParserContextPtr  ctxt,  int
nargs);
extern void xsltRegisterAllFunctions(xmlXPathContextPtr ctxt);
extern  void  xsltSystemPropertyFunction(xmlXPathParserContextPtr
ctxt,
                                         int nargs);
extern void xsltUnparsedEntityURIFunction(xmlXPathParserContextPtr
ctxt,
                                          int nargs);
extern xmlXPathFunction xsltXPathFunctionLookup(xmlXPathContextPtr
ctxt,
                                                const unsigned char *name,
                                                const unsigned char
                                                *ns_uri);
```

9.2.6 libxslt/imports.h

```
#define XSLT_GET_IMPORT_INT(res,style,name)     { xsltStylesheetPtr
st = style; res = -1; while (st != NULL) { if (st->name != -1)
{ res = st->name; break; } st = xsltNextImport(st); }}
#define XSLT_GET_IMPORT_PTR(res,style,name)     { xsltStylesheetPtr
st = style; res = NULL; while (st != NULL) { if (st->name != NULL)
{ res = st->name; break; } st = xsltNextImport(st); }}

extern int xsltFindElemSpaceHandling(xsltTransformContextPtr ctxt,
                                     xmlNodePtr node);
extern   xsltTemplatePtr   xsltFindTemplate(xsltTransformContextPtr
ctxt,
                                      const unsigned char *name,
                                      const unsigned char *nameURI);
extern int xsltNeedElemSpaceHandling(xsltTransformContextPtr ctxt);
extern xsltStylesheetPtr xsltNextImport(xsltStylesheetPtr style);
extern int xsltParseStylesheetImport(xsltStylesheetPtr style,
                                     xmlNodePtr cur);
extern int xsltParseStylesheetInclude(xsltStylesheetPtr style,
                                      xmlNodePtr cur);
```

9.2.7 libxslt/keys.h

```
#define NODE_IS_KEYED   (1 >> 15)

extern int xsltAddKey(xsltStylesheetPtr style, const unsigned char
*name,
                      const unsigned char *nameURI,
                      const unsigned char *match, const unsigned char
*use,
                      xmlNodePtr inst);
extern void xsltFreeDocumentKeys(xsltDocumentPtr doc);
extern void xsltFreeKeys(xsltStylesheetPtr style);
extern xmlNodeSetPtr xsltGetKey(xsltTransformContextPtr ctxt,
                                const unsigned char *name,
                                const unsigned char *nameURI,
                                const unsigned char *value);
extern void xsltInitCtxtKeys(xsltTransformContextPtr ctxt,
                             xsltDocumentPtr doc);
```

9.2.8 libxslt/namespaces.h

```
#define UNDEFINED_DEFAULT_NS    (const xmlChar *) -1L

extern xmlNsPtr xsltCopyNamespace(xsltTransformContextPtr ctxt,
                                  xmlNodePtr elem, xmlNsPtr ns);
extern xmlNsPtr xsltCopyNamespaceList(xsltTransformContextPtr ctxt,
                                      xmlNodePtr node, xmlNsPtr cur);
extern void xsltFreeNamespaceAliasHashes(xsltStylesheetPtr style);
extern xmlNsPtr xsltGetNamespace(xsltTransformContextPtr ctxt,
                                 xmlNodePtr cur, xmlNsPtr ns,
                                 xmlNodePtr out);
extern xmlNsPtr xsltGetPlainNamespace(xsltTransformContextPtr ctxt,
                                      xmlNodePtr cur, xmlNsPtr ns,
                                      xmlNodePtr out);
extern   xmlNsPtr   xsltGetSpecialNamespace(xsltTransformContextPtr
ctxt,
                                      xmlNodePtr cur,
                                      const unsigned char *URI,
                                      const unsigned char *prefix,
                                      xmlNodePtr out);
```



```
extern void xsltNamespaceAlias(xsltStylesheetPtr style, xmlNodePtr
node);
```

9.2.9 libxslt/numbersInternals.h

```
typedef struct _xsltNumberData {
    const unsigned char *level;
    const unsigned char *count;
    const unsigned char *from;
    const unsigned char *value;
    const unsigned char *format;
    int has_format;
    int digitsPerGroup;
    int groupingCharacter;
    int groupingCharacterLen;
    xmlDocPtr doc;
    xmlNodePtr node;
} xsltNumberData;
typedef xsltNumberData *xsltNumberDataPtr;
typedef struct _xsltFormatNumberInfo {
    int integer_hash;
    int integer_digits;
    int frac_digits;
    int frac_hash;
    int group;
    int multiplier;
    char add_decimal;
    char is_multiplier_set;
    char is_negative_pattern;
} xsltFormatNumberInfo;
```

9.2.10 libxslt/pattern.h

```
typedef struct _xsltCompMatch xsltCompMatch;
typedef xsltCompMatch *xsltCompMatchPtr;
extern      int      xsltAddTemplate(xsltStylesheetPtr      style,
xsltTemplatePtr cur,
                    const unsigned char *mode,
                    const unsigned char *modeURI);
extern void xsltCleanupTemplates(xsltStylesheetPtr style);
extern  xsltCompMatchPtr  xsltCompilePattern(const  unsigned  char
*pattern,
                                    xmlDocPtr doc, xmlNodePtr node,
                                    xsltStylesheetPtr style,
                                    xsltTransformContextPtr
                                    runtime);
extern void xsltFreeCompMatchList(xsltCompMatchPtr comp);
extern void xsltFreeTemplateHashes(xsltStylesheetPtr style);
extern   xsltTemplatePtr   xsltGetTemplate(xsltTransformContextPtr
ctxt,
                                xmlNodePtr node,
                                xsltStylesheetPtr style);
extern void xsltNormalizeCompSteps(void *payload, void *data,
                            const unsigned char *name);
extern int xsltTestCompMatchList(xsltTransformContextPtr ctxt,
                        xmlNodePtr node, xsltCompMatchPtr comp);
```

9.2.11 libxslt/preproc.h

```
extern xsltElemPreCompPtr xsltDocumentComp(xsltStylesheetPtr style,
                                    xmlNodePtr inst,
```



```
                                        xsltTransformFunction
function);
extern const xmlChar *xsltExtMarker;
extern void xsltFreeStylePreComps(xsltStylesheetPtr style);
extern    void    xsltStylePreCompute(xsltStylesheetPtr    style,
xmlNodePtr inst);
```

9.2.12 libxslt/security.h

```
typedef struct _xsltSecurityPrefs xsltSecurityPrefs;
typedef xsltSecurityPrefs *xsltSecurityPrefsPtr;
typedef enum {
    XSLT_SECPREF_READ_FILE,
    XSLT_SECPREF_WRITE_FILE,
    XSLT_SECPREF_CREATE_DIRECTORY,
    XSLT_SECPREF_READ_NETWORK,
    XSLT_SECPREF_WRITE_NETWORK
} xsltSecurityOption;
typedef int (*xsltSecurityCheck) (void);
extern int xsltCheckRead(xsltSecurityPrefsPtr sec,
                     xsltTransformContextPtr ctxt,
                     const unsigned char *URL);
extern int xsltCheckWrite(xsltSecurityPrefsPtr sec,
                      xsltTransformContextPtr ctxt,
                      const unsigned char *URL);
extern void xsltFreeSecurityPrefs(xsltSecurityPrefsPtr sec);
extern xsltSecurityPrefsPtr xsltGetDefaultSecurityPrefs(void);
extern xsltSecurityCheck xsltGetSecurityPrefs(xsltSecurityPrefsPtr
sec,
                                              xsltSecurityOption option);
extern xsltSecurityPrefsPtr xsltNewSecurityPrefs(void);
extern int xsltSecurityAllow(xsltSecurityPrefsPtr sec,
                         xsltTransformContextPtr ctxt,
                         const char *value);
extern int xsltSecurityForbid(xsltSecurityPrefsPtr sec,
                          xsltTransformContextPtr ctxt,
                          const char *value);
extern int xsltSetCtxtSecurityPrefs(xsltSecurityPrefsPtr sec,
                                    xsltTransformContextPtr ctxt);
extern void xsltSetDefaultSecurityPrefs(xsltSecurityPrefsPtr sec);
extern int xsltSetSecurityPrefs(xsltSecurityPrefsPtr sec,
                            xsltSecurityOption option,
                            xsltSecurityCheck func);
```

9.2.13 libxslt/templates.h

```
extern                                                  xmlAttrPtr
xsltAttrListTemplateProcess(xsltTransformContextPtr ctxt,
                                        xmlNode * target,
                                        xmlAttrPtr cur);
extern  xmlAttrPtr  xsltAttrTemplateProcess(xsltTransformContextPtr
ctxt,
                                    xmlNode * target,
                                    xmlAttrPtr attr);
extern                                                    xmlChar
*xsltAttrTemplateValueProcess(xsltTransformContextPtr ctxt,
                                      const unsigned char *attr);
extern                                                    xmlChar
*xsltAttrTemplateValueProcessNode(xsltTransformContextPtr
                                        ctxt,
                                        const unsigned char *str,
                                        xmlNode * node);
```



```
extern xmlChar *xsltEvalAttrValueTemplate(xsltTransformContextPtr
ctxt,
                                     xmlNodePtr node,
                                     const unsigned char *name,
                                     const unsigned char *ns);
extern const unsigned char
   *xsltEvalStaticAttrValueTemplate(xsltStylesheetPtr style,
                                xmlNodePtr node,
                                const unsigned char *name,
                                const unsigned char *ns, int *found);
extern   xmlChar   *xsltEvalTemplateString(xsltTransformContextPtr
ctxt,
                                  xmlNodePtr contextNode,
                                  xmlNodePtr inst);
extern int xsltEvalXPathPredicate(xsltTransformContextPtr ctxt,
                              xmlXPathCompExprPtr comp,
                              xmlNs * *nsList, int nsNr);
extern xmlChar *xsltEvalXPathString(xsltTransformContextPtr ctxt,
                             xmlXPathCompExprPtr comp);
extern xmlChar *xsltEvalXPathStringNs(xsltTransformContextPtr ctxt,
                               xmlXPathCompExprPtr comp, int nsNr,
                               xmlNs * *nsList);
extern xmlNode **xsltTemplateProcess(xsltTransformContextPtr ctxt,
                               xmlNode * node);
```

9.2.14 libxslt/transform.h

```
extern void xslHandleDebugger(xmlNodePtr cur, xmlNodePtr node,
                         xsltTemplatePtr templ,
                         xsltTransformContextPtr ctxt);
extern   void   xsltApplyImports(xsltTransformContextPtr   ctxt,
xmlNodePtr node,
                        xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltApplyOneTemplate(xsltTransformContextPtr ctxt,
                         xmlNodePtr node, xmlNodePtr list,
                         xsltTemplatePtr templ,
                         xsltStackElemPtr params);
extern void xsltApplyStripSpaces(xsltTransformContextPtr ctxt,
                          xmlNodePtr node);
extern xmlDocPtr xsltApplyStylesheet(xsltStylesheetPtr style,
                               xmlDocPtr doc, const char **params);
extern xmlDocPtr xsltApplyStylesheetUser(xsltStylesheetPtr style,
                                  xmlDocPtr doc,
                                  const char **params,
                                  const char *output,
                                  FILE * profile,
                                  xsltTransformContextPtr
userCtxt);
extern void xsltApplyTemplates(xsltTransformContextPtr ctxt,
                          xmlNodePtr node, xmlNodePtr inst,
                          xsltStylePreCompPtr comp);
extern void xsltAttribute(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                     xmlNodePtr inst, xsltStylePreCompPtr comp);
extern   void   xsltCallTemplate(xsltTransformContextPtr   ctxt,
xmlNodePtr node,
                        xmlNodePtr inst, xsltStylePreCompPtr comp);
extern  void  xsltChoose(xsltTransformContextPtr  ctxt,  xmlNodePtr
node,
                   xmlNodePtr inst, xsltStylePreCompPtr comp);
extern  void  xsltComment(xsltTransformContextPtr  ctxt,  xmlNodePtr
node,
                    xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltCopy(xsltTransformContextPtr ctxt, xmlNodePtr node,
                 xmlNodePtr inst, xsltStylePreCompPtr comp);
```



```
extern void xsltCopyOf(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                       xmlNodePtr inst, xsltStylePreCompPtr comp);
extern xmlNodePtr xsltCopyTextString(xsltTransformContextPtr ctxt,
                                     xmlNodePtr target,
                                     const unsigned char *string,
                                     int noescape);
extern    void    xsltDocumentElem(xsltTransformContextPtr    ctxt,
xmlNodePtr node,
                           xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltElement(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                       xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltForEach(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                       xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltFreeTransformContext(xsltTransformContextPtr ctxt);
extern int xsltGetXIncludeDefault(void);
extern void xsltIf(xsltTransformContextPtr ctxt, xmlNodePtr node,
                   xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltLocalVariablePop(xsltTransformContextPtr ctxt, int
limitNr,
                                 int level);
extern int xsltLocalVariablePush(xsltTransformContextPtr ctxt,
                                 xsltStackElemPtr variable, int level);
extern                                      xsltTransformContextPtr
xsltNewTransformContext(xsltStylesheetPtr
                                                    style,
                                                    xmlDocPtr doc);
extern void xsltNumber(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                       xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltProcessOneNode(xsltTransformContextPtr ctxt,
                               xmlNodePtr    node,    xsltStackElemPtr
params);
extern void xsltProcessingInstruction(xsltTransformContextPtr ctxt,
                                      xmlNodePtr node, xmlNodePtr inst,
                                      xsltStylePreCompPtr comp);
extern xmlDocPtr xsltProfileStylesheet(xsltStylesheetPtr style,
                                       xmlDocPtr doc, const char **params,
                                       FILE * output);
extern void xsltRegisterAllElement(xsltTransformContextPtr ctxt);
extern int xsltRunStylesheet(xsltStylesheetPtr style, xmlDocPtr doc,
                             const char **params, const char *output,
                             xmlSAXHandlerPtr SAX,
                             xmlOutputBufferPtr IObuf);
extern    int    xsltRunStylesheetUser(xsltStylesheetPtr    style,
xmlDocPtr doc,
                                const char **params, const char *output,
                                xmlSAXHandlerPtr SAX,
                                xmlOutputBufferPtr   IObuf,   FILE   *
profile,
                                xsltTransformContextPtr userCtxt);
extern void xsltSetXIncludeDefault(int xinclude);
extern void xsltSort(xsltTransformContextPtr ctxt, xmlNodePtr node,
                     xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltText(xsltTransformContextPtr ctxt, xmlNodePtr node,
                     xmlNodePtr inst, xsltStylePreCompPtr comp);
extern void xsltValueOf(xsltTransformContextPtr ctxt, xmlNodePtr
node,
                       xmlNodePtr inst, xsltStylePreCompPtr comp);
```

9.2.15 libxslt/variables.h


```
#define                          XSLT_REGISTER_VARIABLE_LOOKUP(ctxt)
xmlXPathRegisterVariableLookup((ctxt)->xpathCtxt,
xsltXPathVariableLookup,              (void             *)(ctxt));
xsltRegisterAllFunctions((ctxt)->xpathCtxt);
xsltRegisterAllElement(ctxt); (ctxt)->xpathCtxt->extra = ctxt

extern int xsltAddStackElemList(xsltTransformContextPtr ctxt,
                            xsltStackElemPtr elems);
extern int xsltEvalGlobalVariables(xsltTransformContextPtr ctxt);
extern int xsltEvalOneUserParam(xsltTransformContextPtr ctxt,
                            const unsigned char *name,
                            const unsigned char *value);
extern int xsltEvalUserParams(xsltTransformContextPtr ctxt,
                           const char **params);
extern void xsltFreeGlobalVariables(xsltTransformContextPtr ctxt);
extern    void    xsltParseGlobalParam(xsltStylesheetPtr    style,
xmlNodePtr cur);
extern void xsltParseGlobalVariable(xsltStylesheetPtr style,
                                xmlNodePtr cur);
extern xsltStackElemPtr
xsltParseStylesheetCallerParam(xsltTransformContextPtr ctxt,
                           xmlNodePtr cur);
extern void xsltParseStylesheetParam(xsltTransformContextPtr ctxt,
                                xmlNodePtr cur);
extern  void  xsltParseStylesheetVariable(xsltTransformContextPtr
ctxt,
                                   xmlNodePtr cur);
extern int xsltQuoteOneUserParam(xsltTransformContextPtr ctxt,
                            const unsigned char *name,
                            const unsigned char *value);
extern int xsltQuoteUserParams(xsltTransformContextPtr ctxt,
                           const char **params);
extern                                          xmlXPathObjectPtr
xsltVariableLookup(xsltTransformContextPtr ctxt,
                                       const unsigned char *name,
                                       const unsigned char *ns_uri);
extern xmlXPathObjectPtr xsltXPathVariableLookup(void *ctxt,
                                          const unsigned char *name,
                                          const unsigned char
                                          *ns_uri);
```

9.2.16 libxslt/xslt.h

```
#define        XSLT_NAMESPACE                  ((xmlChar         *)
"http://www.w3.org/1999/XSL/Transform")
#define XSLT_DEFAULT_VERSION    "1.0"
#define XSLT_DEFAULT_URL       "http://xmlsoft.org/XSLT/"
#define XSLT_DEFAULT_VENDOR     "libxslt"
#define XSLT_PARSE_OPTIONS     XML_PARSE_NOENT | XML_PARSE_DTDLOAD
| XML_PARSE_DTDATTR | XML_PARSE_NOCDATA

extern void xsltCleanupGlobals(void);
extern const char *xsltEngineVersion;
extern void xsltInit(void);
extern const int xsltLibxmlVersion;
extern const int xsltLibxsltVersion;
extern int xsltMaxDepth;
```

9.2.17 libxslt/xsltInternals.h

```
#define XSLT_FAST_IF
#define XSLT_REFACTORED_KEYCOMP
#define XSLT_REFACTORED_VARS
```

```
#define XSLT_IS_TEXT_NODE(n)      ((n != NULL) && (((n)->type ==
XML_TEXT_NODE) || ((n)->type == XML_CDATA_SECTION_NODE)))
#define XSLT_IS_RES_TREE_FRAG(n)        ((n != NULL) && ((n)->type
== XML_DOCUMENT_NODE) && ((n)->name != NULL) && ((n)->name[0] == '
'))
#define  XSLT_RUNTIME_EXTRA_FREE(ctxt,nr)                  (ctxt)-
>extras[(nr)].deallocate
#define XSLT_RUNTIME_EXTRA_LST(ctxt,nr) (ctxt)->extras[(nr)].info
#define           XSLT_RUNTIME_EXTRA(ctxt,nr,typ)           (ctxt)-
>extras[(nr)].val.typ
#define  XSLT_MARK_RES_TREE_FRAG(n)          (n)->name  =  (char  *)
xmlStrdup(BAD_CAST " fake node libxslt");
#define XSLT_PAT_NO_PRIORITY    -12345789
#define XSLT_MAX_SORT   15
#define CHECK_STOPPEDE  if (ctxt->state == XSLT_STATE_STOPPED) goto
error;
#define  CHECK_STOPPED0    if  (ctxt->state  ==  XSLT_STATE_STOPPED)
return(0);
#define  CHECK_STOPPED     if  (ctxt->state  ==  XSLT_STATE_STOPPED)
return;

typedef struct _xsltRuntimeExtra {
    void *info;
    xmlFreeFunc deallocate;
    union {
        void *ptr;
        int ival;
    } val;
} xsltRuntimeExtra;
typedef xsltRuntimeExtra *xsltRuntimeExtraPtr;
typedef struct _xsltTemplate {
    struct _xsltTemplate *next;
    struct _xsltStylesheet *style;
    xmlChar *match;
    float priority;
    const unsigned char *name;
    const unsigned char *nameURI;
    const unsigned char *mode;
    const unsigned char *modeURI;
    xmlNodePtr content;
    xmlNodePtr elem;
    int inheritedNsNr;
    xmlNs **inheritedNs;
    int nbCalls;
    unsigned long int time;
    void *params;
} xsltTemplate;
typedef xsltTemplate *xsltTemplatePtr;
typedef struct _xsltDecimalFormat {
    struct _xsltDecimalFormat *next;
    xmlChar *name;
    xmlChar *digit;
    xmlChar *patternSeparator;
    xmlChar *minusSign;
    xmlChar *infinity;
    xmlChar *noNumber;
    xmlChar *decimalPoint;
    xmlChar *grouping;
    xmlChar *percent;
    xmlChar *permille;
    xmlChar *zeroDigit;
} xsltDecimalFormat;
typedef xsltDecimalFormat *xsltDecimalFormatPtr;
typedef struct _xsltDocument {
    struct _xsltDocument *next;
    int main;
```



```
    xmlDocPtr doc;
    void *keys;
    struct _xsltDocument *includes;
    int preproc;
    int nbKeysComputed;
} xsltDocument;
typedef xsltDocument *xsltDocumentPtr;
typedef struct _xsltKeyDef {
    struct _xsltKeyDef *next;
    xmlNodePtr inst;
    xmlChar *name;
    xmlChar *nameURI;
    xmlChar *match;
    xmlChar *use;
    xmlXPathCompExprPtr comp;
    xmlXPathCompExprPtr usecomp;
    xmlNs **nsList;
    int nsNr;
} xsltKeyDef;
typedef xsltKeyDef *xsltKeyDefPtr;
typedef struct _xsltKeyTable {
    struct _xsltKeyTable *next;
    xmlChar *name;
    xmlChar *nameURI;
    xmlHashTablePtr keys;
} xsltKeyTable;
typedef struct _xsltStylesheet {
    struct _xsltStylesheet *parent;
    struct _xsltStylesheet *next;
    struct _xsltStylesheet *imports;
    xsltDocumentPtr docList;
    xmlDocPtr doc;
    xmlHashTablePtr stripSpaces;
    int stripAll;
    xmlHashTablePtr cdataSection;
    xsltStackElemPtr variables;
    xsltTemplatePtr templates;
    void *templatesHash;
    void *rootMatch;
    void *keyMatch;
    void *elemMatch;
    void *attrMatch;
    void *parentMatch;
    void *textMatch;
    void *piMatch;
    void *commentMatch;
    xmlHashTablePtr nsAliases;
    xmlHashTablePtr attributeSets;
    xmlHashTablePtr nsHash;
    void *nsDefs;
    void *keys;
    xmlChar *method;
    xmlChar *methodURI;
    xmlChar *version;
    xmlChar *encoding;
    int omitXmlDeclaration;
    xsltDecimalFormatPtr decimalFormat;
    int standalone;
    xmlChar *doctypePublic;
    xmlChar *doctypeSystem;
    int indent;
    xmlChar *mediaType;
    xsltElemPreCompPtr preComps;
    int warnings;
    int errors;
    xmlChar *exclPrefix;
```

```
    xmlChar **exclPrefixTab;
    int exclPrefixNr;
    int exclPrefixMax;
    void *_private;
    xmlHashTablePtr extInfos;
    int extrasNr;
    xsltDocumentPtr includes;
    xmlDictPtr dict;
    void *attVTs;
    const unsigned char *defaultAlias;
    int nopreproc;
    int internalized;
    int literal_result;
    xsltStylesheetPtr principal;
} xsltStylesheet;
typedef xsltStylesheet *xsltStylesheetPtr;
typedef struct _xsltTransformContext {
    xsltStylesheetPtr style;
    xsltOutputType type;
    xsltTemplatePtr templ;
    int templNr;
    int templMax;
    xsltTemplatePtr *templTab;
    xsltStackElemPtr vars;
    int varsNr;
    int varsMax;
    xsltStackElemPtr *varsTab;
    int varsBase;
    xmlHashTablePtr extFunctions;
    xmlHashTablePtr extElements;
    xmlHashTablePtr extInfos;
    const unsigned char *mode;
    const unsigned char *modeURI;
    xsltDocumentPtr docList;
    xsltDocumentPtr document;
    xmlNode *node;
    xmlNodeSetPtr nodeList;
    xmlDocPtr output;
    xmlNode *insert;
    xmlXPathContextPtr xpathCtxt;
    xsltTransformState state;
    xmlHashTablePtr globalVars;
    xmlNode *inst;
    int xinclude;
    const char *outputFile;
    int profile;
    long int prof;
    int profNr;
    int profMax;
    long int *profTab;
    void *_private;
    int extrasNr;
    int extrasMax;
    xsltRuntimeExtraPtr extras;
    xsltDocumentPtr styleList;
    void *sec;
    xmlGenericErrorFunc error;
    void *errctx;
    xsltSortFunc sortfunc;
    xmlDocPtr tmpRVT;
    xmlDocPtr persistRVT;
    int ctxtflags;
    const unsigned char *lasttext;
    unsigned int lasttsize;
    unsigned int lasttuse;
    int debugStatus;
```



```
    unsigned long int *traceCode;
    int parserOptions;
    xmlDictPtr dict;
    xmlDocPtr tmpDoc;
    int internalized;
    int nbKeys;
    int hasTemplKeyPatterns;
    xsltTemplatePtr currentTemplateRule;
    xmlNode *initialContextNode;
    xmlDocPtr initialContextDoc;
    xsltTransformCachePtr cache;
    void *contextVariable;
    xmlDocPtr localRVT;
    xmlDocPtr localRVTBase;
    int keyInitLevel;
    int funcLevel;
} xsltTransformContext;
typedef xsltTransformContext *xsltTransformContextPtr;
typedef struct _xsltElemPreComp {
    xsltElemPreCompPtr next;
    xsltStyleType type;
    xsltTransformFunction func;
    xmlNode *inst;
    xsltElemPreCompDeallocator free;
} xsltElemPreComp;
typedef xsltElemPreComp *xsltElemPreCompPtr;
typedef void (*xsltTransformFunction) (void);
typedef void (*xsltSortFunc) (void);
typedef enum {
    XSLT_FUNC_COPY,
    XSLT_FUNC_SORT,
    XSLT_FUNC_TEXT,
    XSLT_FUNC_ELEMENT,
    XSLT_FUNC_ATTRIBUTE,
    XSLT_FUNC_COMMENT,
    XSLT_FUNC_PI,
    XSLT_FUNC_COPYOF,
    XSLT_FUNC_VALUEOF,
    XSLT_FUNC_NUMBER,
    XSLT_FUNC_APPLYIMPORTS,
    XSLT_FUNC_CALLTEMPLATE,
    XSLT_FUNC_APPLYTEMPLATES,
    XSLT_FUNC_CHOOSE,
    XSLT_FUNC_IF,
    XSLT_FUNC_FOREACH,
    XSLT_FUNC_DOCUMENT,
    XSLT_FUNC_WITHPARAM,
    XSLT_FUNC_PARAM,
    XSLT_FUNC_VARIABLE,
    XSLT_FUNC_WHEN,
    XSLT_FUNC_EXTENSION
} xsltStyleType;
typedef void (*xsltElemPreCompDeallocator) (void);
typedef struct _xsltStylePreComp {
    xsltElemPreCompPtr next;
    xsltStyleType type;
    xsltTransformFunction func;
    xmlNode *inst;
    const unsigned char *stype;
    int has_stype;
    int number;
    const unsigned char *order;
    int has_order;
    int descending;
    const unsigned char *lang;
    int has_lang;
```

```
    xsltLocale locale;
    const unsigned char *case_order;
    int lower_first;
    const unsigned char *use;
    int has_use;
    int noescape;
    const unsigned char *name;
    int has_name;
    const unsigned char *ns;
    int has_ns;
    const unsigned char *mode;
    const unsigned char *modeURI;
    const unsigned char *test;
    xsltTemplatePtr templ;
    const unsigned char *select;
    int ver11;
    const unsigned char *filename;
    int has_filename;
    xsltNumberData numdata;
    xmlXPathCompExprPtr comp;
    xmlNs **nsList;
    int nsNr;
} xsltStylePreComp;
typedef xsltStylePreComp *xsltStylePreCompPtr;
typedef struct _xsltStackElem {
    struct _xsltStackElem *next;
    xsltStylePreCompPtr comp;
    int computed;
    const unsigned char *name;
    const unsigned char *nameURI;
    const unsigned char *select;
    xmlNode *tree;
    xmlXPathObjectPtr value;
    xmlDocPtr fragment;
    int level;
    xsltTransformContextPtr context;
    int flags;
} xsltStackElem;
typedef xsltStackElem *xsltStackElemPtr;
typedef struct _xsltTransformCache {
    xmlDocPtr RVT;
    int nbRVT;
    xsltStackElemPtr stackItems;
    int nbStackItems;
} xsltTransformCache;
typedef xsltTransformCache *xsltTransformCachePtr;
typedef enum {
    XSLT_OUTPUT_XML,
    XSLT_OUTPUT_HTML,
    XSLT_OUTPUT_TEXT
} xsltOutputType;
typedef enum {
    XSLT_STATE_OK,
    XSLT_STATE_ERROR,
    XSLT_STATE_STOPPED
} xsltTransformState;
extern int xsltAllocateExtra(xsltStylesheetPtr style);
extern int xsltAllocateExtraCtxt(xsltTransformContextPtr ctxt);
extern  void  xsltCompileAttr(xsltStylesheetPtr  style,  xmlAttrPtr
attr);
extern xmlDocPtr xsltCreateRVT(xsltTransformContextPtr ctxt);
extern                                              xsltDecimalFormatPtr
xsltDecimalFormatGetByName(xsltStylesheetPtr
                                                    style,
                                                    xmlChar * name);
extern xmlChar *xsltEvalAVT(xsltTransformContextPtr ctxt, void *avt,
```



```
                        xmlNode * node);
extern                                                          int
xsltExtensionInstructionResultFinalize(xsltTransformContextPtr
                                        ctxt);
extern                                                          int
xsltExtensionInstructionResultRegister(xsltTransformContextPtr
                                        ctxt,
                                        xmlXPathObjectPtr obj);
extern                                                  xmlXPathError
xsltFormatNumberConversion(xsltDecimalFormatPtr self,
                                      xmlChar * format,
                                      double number,
                                      xmlChar * *result);
extern void xsltFreeAVTList(void *avt);
extern void xsltFreeRVTs(xsltTransformContextPtr ctxt);
extern void xsltFreeStackElemList(xsltStackElemPtr elem);
extern void xsltFreeStylesheet(xsltStylesheetPtr style);
extern int xsltInitAllDocKeys(xsltTransformContextPtr ctxt);
extern int xsltInitCtxtKey(xsltTransformContextPtr ctxt,
                           xsltDocumentPtr doc, xsltKeyDefPtr keyd);
extern int xsltIsBlank(xmlChar * str);
extern xsltStylesheetPtr xsltLoadStylesheetPI(xmlDocPtr doc);
extern xsltStylesheetPtr xsltNewStylesheet(void);
extern void xsltNumberFormat(xsltTransformContextPtr ctxt,
                         xsltNumberDataPtr data, xmlNode * node);
extern xsltStylesheetPtr xsltParseStylesheetDoc(xmlDocPtr doc);
extern  xsltStylesheetPtr  xsltParseStylesheetFile(const  unsigned
char
                                                    *filename);
extern  xsltStylesheetPtr  xsltParseStylesheetImportedDoc(xmlDocPtr
doc,
                                                          xsltStylesheetPtr
                                                          style);
extern void xsltParseStylesheetOutput(xsltStylesheetPtr style,
                                      xmlNode * cur);
extern                                                  xsltStylesheetPtr
xsltParseStylesheetProcess(xsltStylesheetPtr ret,
                                             xmlDocPtr doc);
extern void xsltParseTemplateContent(xsltStylesheetPtr style,
                                     xmlNode * templ);
extern int xsltRegisterLocalRVT(xsltTransformContextPtr ctxt,
                                xmlDocPtr RVT);
extern int xsltRegisterPersistRVT(xsltTransformContextPtr ctxt,
                                  xmlDocPtr RVT);
extern    int    xsltRegisterTmpRVT(xsltTransformContextPtr    ctxt,
xmlDocPtr RVT);
extern void xsltReleaseRVT(xsltTransformContextPtr ctxt, xmlDocPtr
RVT);
extern void xsltUninit(void);
```

9.2.18 libxslt/xsltconfig.h

```
#define LIBXSLT_VERSION_EXTRA   ""
#define WITH_DEBUGGER
#define WITH_MODULES
#define WITH_XSLT_DEBUG
#define XSLT_LOCALE_XLOCALE
#define LIBXSLT_DEFAULT_PLUGINS_PATH()  "/usr/lib/libxslt-plugins"
#define LIBXSLT_DOTTED_VERSION  "1.1.26"
#define LIBXSLT_VERSION 10126
#define LIBXSLT_VERSION_STRING  "10126"
```

9.2.19 libxslt/xsltexports.h

```
#define XSLTCALL
#define XSLTPUBFUN
#define XSLTPUBVAR      extern
#define LIBXSLT_PUBLIC  XSLTPUBVAR
```

9.2.20 libxslt/xsltlocale.h

```
#define XSLT_LOCALE_NONE

typedef void *xsltLocale;
typedef unsigned char xsltLocaleChar;
extern void xsltFreeLocale(xsltLocale locale);
extern int xsltLocaleStrcmp(xsltLocale locale, const xsltLocaleChar
* str1,
                        const xsltLocaleChar * str2);
extern xsltLocale xsltNewLocale(const unsigned char *langName);
extern xsltLocaleChar *xsltStrxfrm(xsltLocale locale,
                               const unsigned char *string);
```

9.2.21 libxslt/xsltutils.h

```
#define IS_XSLT_REAL_NODE(n)     (((n) != NULL) && ((n)->type ==
XML_ELEMENT_NODE) || ((n)->type == XML_TEXT_NODE) || ((n)->type ==
XML_CDATA_SECTION_NODE) || ((n)->type == XML_ATTRIBUTE_NODE) ||
((n)->type     ==     XML_DOCUMENT_NODE)     ||     ((n)->type     ==
XML_HTML_DOCUMENT_NODE) || ((n)->type == XML_COMMENT_NODE) || ((n)-
>type == XML_PI_NODE)))
#define IS_XSLT_ELEM(n)  (((n) != NULL) && ((n)->ns != NULL) &&
(xmlStrEqual((n)->ns->href, XSLT_NAMESPACE)))
#define IS_XSLT_NAME(n,val)          (xmlStrEqual((n)->name, (const
xmlChar *) (val)))
#define XSLT_TIMESTAMP_TICS_PER_SEC    1000001
#define XSLT_TRACE(ctxt,code,call)         if (ctxt->traceCode &&
(*(ctxt->traceCode) & code)) call
#define XSLT_STRANGE     xsltGenericError(xsltGenericErrorContext,
"Internal error at %s:%d\n", __FILE__, __LINE__);
#define XSLT_TODO        xsltGenericError(xsltGenericErrorContext,
"Unimplemented block at %s:%d\n", __FILE__, __LINE__);

typedef enum {
    XSLT_TRACE_ALL,
    XSLT_TRACE_NONE,
    XSLT_TRACE_COPY_TEXT,
    XSLT_TRACE_PROCESS_NODE,
    XSLT_TRACE_APPLY_TEMPLATE,
    XSLT_TRACE_COPY,
    XSLT_TRACE_COMMENT,
    XSLT_TRACE_PI,
    XSLT_TRACE_COPY_OF,
    XSLT_TRACE_VALUE_OF,
    XSLT_TRACE_CALL_TEMPLATE,
    XSLT_TRACE_APPLY_TEMPLATES,
    XSLT_TRACE_CHOOSE,
    XSLT_TRACE_IF,
    XSLT_TRACE_FOR_EACH,
    XSLT_TRACE_STRIP_SPACES,
    XSLT_TRACE_TEMPLATES,
    XSLT_TRACE_KEYS,
    XSLT_TRACE_VARIABLES
} xsltDebugTraceCodes;
```



```
extern int xslAddCall(xsltTemplatePtr templ, xmlNode * source);
extern int xslDebugStatus;
extern void xslDropCall(void);
extern void xsltCalibrateAdjust(long int delta);
extern                                                  xmlXPathObject
**xsltComputeSortResult(xsltTransformContextPtr ctxt,
                                        xmlNode * sort);
extern xsltDebugTraceCodes xsltDebugGetDefaultTrace(void);
extern void xsltDebugSetDefaultTrace(xsltDebugTraceCodes val);
extern void xsltDefaultSortFunction(xsltTransformContextPtr ctxt,
                                xmlNode * *sorts, int nbsorts);
extern void xsltDoSortFunction(xsltTransformContextPtr ctxt,
                           xmlNode * *sorts, int nbsorts);
extern void xsltDocumentSortFunction(xmlNodeSetPtr list);
extern xmlGenericErrorFunc xsltGenericDebug;
extern void *xsltGenericDebugContext;
extern xmlGenericErrorFunc xsltGenericError;
extern void *xsltGenericErrorContext;
extern const unsigned char *xsltGetCNsProp(xsltStylesheetPtr style,
                                    xmlNodePtr node,
                                    const unsigned char *name,
                                    const      unsigned      char
*nameSpace);
extern int xsltGetDebuggerStatus(void);
extern xmlChar *xsltGetNsProp(xmlNodePtr node, const unsigned char
*name,
                           const unsigned char *nameSpace);
extern xmlDocPtr xsltGetProfileInformation(xsltTransformContextPtr
ctxt);
extern const unsigned char *xsltGetQNameURI(xmlNode * node,
                                    xmlChar * *name);
extern  const  unsigned  char  *xsltGetQNameURI2(xsltStylesheetPtr
style,
                                    xmlNode * node,
                                    const unsigned char **name);
extern int xsltGetUTF8Char(const unsigned char *utf, int *len);
extern  void  xsltMessage(xsltTransformContextPtr  ctxt,  xmlNodePtr
node,
                      xmlNodePtr inst);
extern void xsltPrintErrorContext(xsltTransformContextPtr ctxt,
                              xsltStylesheetPtr style,
                              xmlNodePtr node);
extern void xsltSaveProfiling(xsltTransformContextPtr ctxt, FILE *
output);
extern  int  xsltSaveResultTo(xmlOutputBufferPtr  buf,  xmlDocPtr
result,
                          xsltStylesheetPtr style);
extern int xsltSaveResultToFd(int fd, xmlDocPtr result,
                           xsltStylesheetPtr style);
extern int xsltSaveResultToFile(FILE * file, xmlDocPtr result,
                             xsltStylesheetPtr style);
extern  int  xsltSaveResultToFilename(const  char  *URI,  xmlDocPtr
result,
                              xsltStylesheetPtr style,
                              int compression);
extern  int  xsltSaveResultToString(xmlChar  *  *doc_txt_ptr,  int
*doc_txt_len,
                              xmlDocPtr result,
                              xsltStylesheetPtr style);
extern int xsltSetCtxtParseOptions(xsltTransformContextPtr ctxt,
                               int options);
extern void xsltSetCtxtSortFunc(xsltTransformContextPtr ctxt,
                            xsltSortFunc handler);
extern int xsltSetDebuggerCallbacks(int no, void *block);
extern void xsltSetDebuggerStatus(int value);
extern void xsltSetGenericDebugFunc(void *ctx,
```