Free C or C++ XML Parser Libraries

Last Update: 2012-02-15.

This page tries to give a comparison of existing free C or C++ XML parser libraries. It includes both full blown as well as lightweight parsers.

This list is of course not exhaustive, but it claims to include all free parser libraries that have a significant user base, that are more or less actively maintained and that cover the most widely used desktop PC platforms, i.e. Win32, Unix, Linux, Mac-OS, BSD. So if you feel i have missed one, please tell me.

I'll start with a comparison table giving a quick overview of all available libraires. The following section gives some more detail, mostly based on my personal impressions and/or experiences on the Win32 platform. If you find any inaccuracies or want to contribute to this comparison table by providing a feedback with your own experiences, you are welcome to send me an email at lars.ruoff@gmx.net.

Comparison Table

The following table gives an overview of the most popular free C or C++ XML parser libraries. Most of them are multi-platform and come with C/C++ source code. See my personal comments below.

Library	Language	Features/Interfaces	License	Platforms/Packaging
Full blown (SAX,DOM,Validating) XML Parser Libraries:				
Xerces http://xml.apache.org/xerces-c/index.html	C++	DOM, SAX2, XML Schema, Namespaces, Validation	Apache License, Version 2.0	Precompiled libraries for Windows, UNIX, Linux, etc. Optionally source+makefiles.
Arabica http://www.jezuk.co.uk/cgi-bin/view/arabica	C++	DOM, SAX2, XPath	BSD style	C++ source with VC7, VC8 and Unix makefiles
libxml2 http://xmlsoft.org/	C, various language bindings	DOM, SAX (limited), XPath, XmlTextReader (pull)	MIT License	Linux, Unix, Windows, CygWin, MacOS, MacOS X, RISC Os, OS/2, VMS, QNX, MVS, ...
libxml++ http://libxmlplusplus.sourceforge.net/	C++	DOM, SAX	Gnu LGPL	(see libxml2)
MSXML MSXML at msdn.microsoft.com	C++, COM interface	DOM, SAX2, XSLT, XML Schemas	Proprietary (Microsoft)	MS Windows only. Comes with Windows XP/Vista.
Fast/Low footprint XML Parser Libraries:				
expat http://expat.sourceforge.net/	C, various 3rd party language bindings available	Stream oriented. Event based. Various third party SAX/DOM wrappers and language bindings.	MIT license	C source
TinyXml http://www.grinninglizard.com/tinyxml/	C++	DOM	zlib license	C++ source with VC6 and Unix makefiles
Xmlio http://www.fxtech.com/xmlio/	C++	C++ I/O stream oriented. "pull" model.	Gnu LGPL	C++ source with VC6 and Unix makefiles
XmlLite Article at msdn.microsoft.com	C++	Stream oriented, "pull" model.	Proprietary (Microsoft)	MS Windows only. Part of Vista SDK.
RapidXml http://rapidxml.sourceforge.net/index.htm	C++	DOM-style	Boost 1.0 or MIT License	Portable C++, header only.
XMLParser library from Frank Vanden Berghen http://www.applied-mathematics.net/tools/xmlParser.html	C++	DOM-style	Aladdin Free Public License	Source code with makefiles and VC6 workspace.
		DOM style, with		

PugiXML http://code.google.com/p/pugixml/	C++	interface to walk the tree. XPath	MIT License	C++ source
libroxml http://www.libroxml.net/	C	DOM style	LGPL v2.1	Source tar.gz or debian package

My Comments

Xerces

http://xml.apache.org/xerces-c/index.html

Fully validating XML 1.0 parser. Huge and powerful. Not personally tested.

Size of Windows DLL is 1.8Mb. Said to be relatively slow. Good documentation, examples and tutorial. Compliant to XML 1.0, partially XML 1.1, DOM lv1, DOM lv2 Core, partially DOM lv3, SAX 1.0/2.0, Namespaces, XML Schema.

Arabica

http://www.jezuk.co.uk/cgi-bin/view/arabica

SAX2, DOM, XPath and partial XSLT implementation to be used with expat, libxml, Xerces or MSXML. Not tested.

Arabica is not an XML parser on its own. Instead, it uses an underlying parser like expat, libxml, Xerces or MSXML to do the low level parsing. The objective of Arabica is to provide a nice standard C++ implementation of DOM and SAX2 interfaces on top of these parsers. Hence, Arabica has to be set up and built for one of the underlying parsers before use. This can make the installation a bit fiddly and requires some additional time for setup. Arabica+expat seems to be an interesting combination, both in terms of performance/footprint and ease of use. (Not tested myself though).

Package includes short example code for each API.

libxml2

http://xmlsoft.org/

Gnome's XML parser library. It's the standard XML lib of the GNU folks. Can be cumbersome to get up and running on Windows, since it heavily depends on other GNU infrastructure (glib etc.). Included documentation is sparse but there are a lot of external sites providing docs, examples, tutorials.

libxml++

http://libxmlplusplus.sourceforge.net/

C++ bindings for libxml2. I.e. you have to have libxml2 installed in order to use libxml++. Features a nice standard C++ interface to libxml2. On Windows it is difficult to install and set up due to multiple dependencies on other GNU libs. For example uses Glib::ustring instead of std:strings. A short manual and reference is included.

Warning: For Win32, libxml++1.0 is better suited than libxml++2.x, due to a nasty glibmm dependency.

expat

http://expat.sourceforge.net/

Very fast, low level XML parser for small or embedded applications or as a basis for higher level parser APIs. Stream oriented. Event based. Various third party SAX/DOM wrappers and language bindings. Especially, see expatpp C++ wrapper for expat. (Not tested).

Said to be robust and well tested. Is used with many open source projects.

TinyXml

http://www.grinninglizard.com/tinyxml/

Not tested. There is also another project, TiCPP - TinyXML++, which aims to be a more advanced "C++ style" interface to TinyXml.

Xmlio

http://www.fxtech.com/xmlio/

Fast and tiny. Uses a proprietary stream oriented "pull" model. Limitations:

- only ASCII encoding right now
- no namespaces
- no PI handlers

My tests showed that other than ASCII characters can be read.

XmlLite

Article at msdn.microsoft.com
XmlLite Documentation

High performance, lightweight native C++ XML parser from Microsoft. "The primary goals of XmlLite are ease of use, performance, and standards compliance." Not tested.

It aims to provide a more lightweight and hassle-free alternative to MSXML with a focus on native C++ development (as opposed to managed/.NET). XmlLite features a simple "pull" programming model with a stream-oriented XmlReader class. It has support for a large set of common character encodings but only limited support for DTD. As usual for a Microsoft product, there are lots of high quality documentation and articles.

RapidXml

http://rapidxml.sourceforge.net/index.htm

It is a DOM-style in-situ parser written in modern C++, which tries to be as fast as possible. Claims to be a "seriously fast and small parser, [with] hassle-free integration". Entire library is contained in a single header file, and requires no building or configuration. A concise online manual including examples is avaialble. Not tested.

You may also like to check Boost.PropertyTree library, which presents a higher level interface, and uses RapidXml as its default XML parser.

XMLParser library from Frank Vanden Berghen

http://www.applied-mathematics.net/tools/xmlParser.html

Simple and fast DOM-type parser. Supports XML namespaces, wide range of character sets & encodings. No library dependencies other than <stdio.h>. A small tutorial is included on the website. Not tested myself.

[Direct link to the library file]

PugiXML

http://code.google.com/p/pugixml/

"Light-weight, simple and fast XML parser for C++." Pros and Cons (as advertised):

- Low memory consumption and fragmentation (compared to other DOM style parsers).
- Extremely high parsing speed.
- Standard-conformant (with the exception of DTD related issues).
- Pretty much error-ignorant.
- Assumes UTF-8 encoding of the input data. UTF-16/32 not supported.
- Fully standard compliant C++ code. Multiplatform.
- High flexibility. You can control many aspects of file parsing and DOM tree building via parsing options.
- Lacks validation, DTD processing, XML namespaces, proper handling of encoding.
- Lacks UTF-16/32 parsing.

The website includes a short documentation page including some code samples that illustrate the use of the library. Not tested myself.

libroxml

http://www.libroxml.net/

This library is a minimum, easy-to-use, C implementation for xml file parsing. Libroxml targets mainly embedded software and environments, but you can use it whenever you need to deal with XML since libroxml is ligth and fast.

Info from the developer: The packaging is tar.gz or debian package (not yet in debian repo). It compiles under Linux, MacOSX and Windows (using cmake). No dependencies are used except one mutex (nativ criticalsection under windows or pthread for Linux and osx).

The target is embedded software so it is a DOM like API but with a very low footprint as it only indexes the XML content inside the file instead of loading it all.

API supports XML parsing of file or buffers, navigation inside the tree, a large subset of xpath is handled, tree can be created/modified in RAM and then commited (serialized) to a file or buffer. Namespace is about to be fully supported in next release.

Not tested myself.

Links

- XML article at Wikipedia
- XML Benchmarking - A benchmarking toolset for all available multiplatform C/C++ XML libraries.
- C/C++ developers: Fill your XML toolbox - Tools & advice for C and C++ programmers ramping up on XML
- Comparison of fast, multiplatform XML libraries
- CodeSynthesis XSD: An open-source, cross-platform W3C XML Schema to C++ data binding compiler
- UTF8-CPP: UTF-8 with C++ in a Portable Way
