Adobe® XMP Toolkit SDK Overview

This overview introduces the Toolkit SDK for the Extensible Metadata Platform (XMP).

The Adobe XMP Toolkit SDK provides documentation and libraries for working with the XMP data model; for reading, writing and manipulating XMP metadata in various file formats. Refer to the XMP Specification (included in this SDK) for detailed descriptions of the XMP data model, file type support, and schemas supported in Adobe products. The XMP Toolkit SDK is published under a BSD License.

SDK components

The XMP Toolkit SDK contains two libraries, XMPCore and XMPFiles.

- XMPCore

  This library supplies an API for parsing, manipulating, and serializing metadata, according to the XMP data model and regardless of the file format of the data it describes. The XMPCore API is provided by the classes XMPMeta, XMPIterator, and XMPUtils; a complete API Reference is available in both HTML and Javadoc.

  XMPCore is provided as a C++ and Java implementation with project files for:

  - Windows XP and above (32 and 64 bit) using Visual C++ 2008 (Visual Studio Version 9)
  - Mac OS X 10.4 and above using Xcode 3.x for Intel processors.
  - Makefiles for the GNU C Compiler (gcc) Version 4.x to compile under Linux.
    How to compile under Linux is explained in <xmpsdk>\build\gcc4\usage_examples.txt
  - A *Java implementation* of XMPCore is also provided, to be used with J2SE Version 1.5/5.0 or higher. Project files for Eclipse 3 and above and an Ant build file are included.

- XMPFiles

  This library supplies an API for locating, adding, or updating the XMP metadata in a file. The API allows you to retrieve the entire XMP Packet, which you can then pass to the XMPCore component in order to manipulate the individual XMP properties.

  XMPFiles contains a number of "smart" file handlers that know how to efficiently access the XMP in specific file formats. See the XMP Specification Part 3, Storage in Files for details of how XMP is embedded in the supported file formats.

XMPFiles is provided as a C++ implementation with project files for:

- Windows XP and above (32 and 64 bit) using Visual C++ 2008 (Visual Studio Version 9)
- Mac OS X 10.4 and above using Xcode 3.x for Intel processors.
- Makefiles for the GNU C Compiler (gcc) Version 4.x to compile under Linux.

Dependencies

These publicly available components are needed to build the C/C++ libraries:

- The Expat XML parser is needed for XMPCore on all platforms.
- The zlib compression library is needed for XMPFiles on all platforms.

See instructions on obtaining and installing these tools in the ReadMe.txt files in the placeholder folders for each tool, and in the XMP Toolkit SDK Programmer's Guide. There are no dependencies for the Java version of XMPCore.

Downloading the XMP Toolkit SDK

This single zip file contains everything needed under Mac OS®, Windows®, and UNIX®/Linux®. Note that all source and text files have UNIX-style line endings with the exception of Visual Studio project files, which require Windows-style (CRLF) line endings.

XMP Toolkit SDK contents

The downloadable ZIP file contains the following folders under the root folder XMP-Toolkit-SDK-5.x.x:

/	At the root level, the license agreement (BSD_License.txt) and this overview (XMP-Toolkit-SDK-Overview.pdf).
build/	Projects for building the C++ version of the XMP Toolkit SDK on Intel Macs, Windows. Makefiles for building under Linux using gcc4. How to compile under Linux is explained in build\gcc4\usage_examples.txt
docs/	The three-part XMP Specification, the XMP Toolkit SDK Programmer's Guide, and the API reference documentation (API/index.html).
java/	The Java implementation of XMPCore, with the Javadoc documentation, project files for Eclipse 3.x, and sample code. The readme.txt file describes how to set up projects in Eclipse.
public/include/	The header files and glue code that clients of the XMP Toolkit SDK must include.

samples/	Sample and tutorial source code and build projects, with the necessary resources to run the sample code. See “Sample code and tools” below.
source/	The source code that implements the XMP Toolkit SDK libraries.
third-party/ expat/ zlib/ MD5/	Place holders for third party source files which are needed for the XMP Toolkit SDK, including ReadMe.txt files with information on how to obtain and install the tools. MD5 source code, needed by both components for MD5 hash computation, is included.

Sample code and tools

The SDK provides a set of samples that illustrate coding techniques for various tasks. In addition to the source code for each sample, there is a project file for use with a platform-specific IDE.

- Project files for MS Visual C++ 2008 are in the folder <*xmpsdk*>\samples\build\vc9.
- Project files for Xcode 3 are in the folder <*xmpsdk*>/samples/build/xcode3
- Makefiles for GCC 4 are in the folder <*xmpsdk*>/samples/build/gcc4

The source code for the samples is in <*xmpsdk*>/samples/source. When you build them, the compiled code is written to <*xmpsdk*>/samples/target/, to a platform-specific folder with debug and release subfolders.

These command-line sample applications are provided:

`ReadingXMP` *`filename`*	Demonstrates the basic use of the XMPFiles and XMPCore components, obtaining read-only XMP from a file and examining it through the XMP object.
`ModifyingXMP` *`filename`*	Demonstrates how to open a file for update, and modifying the contained XMP before writing it back to the file.
`CustomSchema`	Demonstrates how to work with a custom schema that has complex properties. It shows how to access and modify properties with complex paths using the path composition utilities from the XMP API.
`XMPCoreCoverage` `XMPFilesCoverage`	These demonstrate syntax and usage by exercising most of the API functions of each XMP Toolkit SDK component, using a sample XMP Packet that contains all of the different property and attribute types.
`XMPIterations`	Demonstrates how to use the iteration utility in the XMPCore component to walk through property trees.
`DumpMainXMP` *`filename`*	Uses the XMPFiles component API to find the main XMP Packet for a data file, serialize the XMP, and display it.
`DumpScannedXMP` *`filename`*	Scans a data file to find all embedded XMP Packets, without using the XMPFiles API or smart handlers. If a packet is found, serializes the XMP and displays it.

In addition, these command-line development tools are provided:

`Dumpfile`	Parses the structure of the given file and dumps a view of the file structure to standard out. This tool is a developer tool, not intended for any use production. This tool helps you determine whether a file is well-formed and to understand its structure.
`xmpcommand`	Performs basic XMP actions such as get, put, and dump. Can be used for testing and scripting automation.

For additional information about how to use the samples and tools, and for tutorial walkthroughs of the basic samples, see the XMP Toolkit SDK Programmer's Guide.

XMP Toolkit SDK changes

VERSION 5.1.2

- Compiler Upgrade: Xcode 3.x, Visual Studio 2008 (VC9)
- New Makefiles for gcc version 4 and also for XMPFiles.
- 64-Bit Support on Windows (as before), Mac and Linux/Unix.

- XMPCore and XMPFiles are more thread-friendly now by using a multiple reader/single writer locking mechanism. Please refer to the XMP Toolkit SDK Programmer's Guide for further details.
- The XMPUtils method *AppendProperties* has been deprecated and replaced by the method *ApplyTemplate.* Please refer to the XMP Toolkit SDK Programmer's Guide for further details.
- Added support for date/time values with no timezone

The string form of date/time values and the XMP_DateTime binary type now formally support date/time values that are in no timezone. The ISO 8601 date/time format that underlies the XMP usage requires a timezone for all but date-only values. This clashes with Exif usage, where there is no timezone for the DateTime, DateTimeOriginal, or DateTimeDigitized tags. This change allows importation from Exif without forcing an often incorrect timezone to be applied.

- The functionality to create custom aliases has been removed from XMPCore.
- Handler for RIFF based formats (AVI and WAV) has been rewritten.

The AVI and WAV handlers have been combined in a new RIFF handler. The placement logic for metadata blocks has changed some, with more appropriate support for AVI files over 2 GB. Some additional non-XMP metadata is supported; details are in the XMP Specification Part 3. The LIST/INFO chunk in WAV is now written using UTF-8 text, it was formerly written using local text. When reading a check is made for valid UTF-8, if not found a local to Unicode conversion is done. This is similar to the long standing practice for Exif metadata in digital photos.

- The support for MOV (QuickTime) files has been integrated into the MPEG-4 handler. The main impact of this is the removal of dependence on Apple's QuickTime SDK, which also means that MOV file support is now available in Linux builds of XMPFiles.
- MP3 handler has been rewritten. More stability and support for XMP reconciliation of ID3v2 Tags. Details are in the XMP Specification Part 3.
- XMPFiles now begins compliance with the Metadata Working Group (MWG) Guidelines: http://www.metadataworkinggroup.com/pdf/mwg_guidance.pdf

A variety of changes have been made in the JPEG, TIFF, and PSD file handlers to begin compliance with the Metadata Working Group (MWG) Image Metadata Guidelines. Highlights of the changes are:

- Greatly simplified selection of Exif and IIM blocks.
- Removal of tiff: and exif: namespaces from stored XMP.
- Removal of digests for the native TIFF and Exif metadata.
- Generally prefer Exif over IIM and XMP.
- Always write IIM as UTF-8, including 1:90 DataSet.
- Allow multiple values for Creator in IIM (ByLine, 2:80).
- Allow xmp:Rating to be a floating point value.
- Additional mappings for date/time items in Exif, IIM, and XMP.

The phrase "begin compliance" was used because the XMP SDK is not yet fully MWG-compliant. A few issues were found during CS5 pre-release testing. Some decisions were made to have more transition time in these cases:

Instead of always preferring Exif when reading, an existing IIM or XMP value is preferred over Exif for ImageDescription, Artist, Copyright, DateTime, and DateTimeDigitized. This enhances compatibility with existing applications that write only the IIM or XMP forms of these, leaving existing Exif unchanged. Those applications should themselves become MWG compliant by ensuring that all forms in a file are consistent.

The mapping of Exif DateTimeOriginal to photoshop:DateCreated and IIM DateCreated is not done. Exif DateTimeOriginal is still mapped to exif: DateTimeOriginal in XMP. This improves compatibility with existing applications that already copy the date portion, and with possible existing user practice. More sophisticated compatibility heuristics are under investigation.

- The file handlers for the folder based formats P2 and AVCHD have been updated.
- Added "server mode" on Linux that disables local text encoding

A "server mode" option has been added for SXMPFiles::Initialize, kXMPFiles_ServerMode. The only effect of this currently is to cause "local" text encodings to be ignored when reading and writing. Local text is defined relative to a user's local O/S settings. It has little meaning on servers where files can come from anywhere, and in addition Linux has no notion of a current text encoding. Server mode should be used for all server software, even on Macintosh and Windows. Server mode is required for Linux builds of XMPFiles.

- Dropped support for PowerPC architecture on Mac.
- Dropped support for gcc 3.2.
- Several Bugfixes. One important one is:

The AVI and WAV file handlers would sometimes write a RIFF file chunk of size 0. This would trigger a bug in Windows 7, causing an infinite loop in the O/S code. Microsoft has patched this bug in December 2009. The XMPFiles code has been changed to never generate 0 length RIFF chunks.

VERSION 4.4.2:

- Additional smart Handlers for additional file formats, including ASF (WMA, WMV), FLV; MPEG4; SWF; folder-based video formats AVCHD, P2, SonyHDV, and XDCAM; UCF (see XMP Specification Part 3, Storage in Files).
- Additional schemas to support document histories, composed documents, and temporal metadata (see XMP Specification Part 2, Standard Schemas).
- Xcode projects work in Xcode 3
- VS8 projects for Windows now include 64-bit build targets for Windows.
- Expanded, updated, and reorganized documentation. The XMP Toolkit SDK Programmer's Guide has been renamed and updated for new features. The XMP Specification has been split into three parts; Part 1, Data and Serialization Models Part 2, Standard Schemas, and Part 3, Storage in Files.

- Additional and updated sample code. See "Sample code and tools" on page 3.

VERSION 4.1.1: Added the XMPFiles library and the Java version of XMPCore. Visual Studio 2005 (VC8) projects replaced Visual Studio .Net 2003 (VC7) projects. Code Warrior projects were removed.

VERSION 3.5: Added Xcode projects for building universal binaries in Mac OS. Added functions to the SXMPUtils class to support the latest Adobe DNG SDK.

VERSION 3.2: A complete rewrite of the XMPCore, for a more convenient API, and a smaller, faster, more robust implementation.
