Adding Intelligence to Media

XMP Toolkit SDK Overview

February 2022

Adobe

Adobe XMP Toolkit SDK Overview

Contents

Overview 4
SDK components 4
XMPCore 4
XMPFiles 4
Dependencies 5
XMP Toolkit SDK 6
XMP Toolkit SDK contents 6
Sample code and tools 7
XMP Toolkit SDK changes 9

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 HTML.

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 on how XMP is embedded in the supported file formats.

With the XMPFiles plug-in SDK, it is possible to add custom file handlers or replace existing file handlers with custom ones in XMPFiles.

The XMP toolkit implementation is provided for:

- Windows 10 (64 bit) using Visual C++ 2019 (Visual Studio Version 16)
- Mac OS X 10.15.4 and above using Xcode 12.4 for universal binaries
- GNU C Compiler (gcc) Version 4.8.2 to compile under Linux (32 and 64 bit).
- How to compile under Linux is explained in <xmpsdk>\build\README.txt
- iOS and Android, for XMPCore and XMPFiles component
- Like the previous release, instead of providing Project files for different platforms, CMake script files are provided which are capable of generating project files for Windows, iOS, Mac and Linux Platform. For using this, the CMake tool needs to be downloaded and placed at <xmpsdk>\tools\cmake folder.

Dependencies

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

- The Expat XML parser is needed for XMPCore and XMP Files on all platforms.
- The CMake tool for creating projects on all platforms.
- The zlib compression library is needed for XMPFiles on all platforms.
- The ndk-bundle , ninja, and cmake are needed for XMPCore and XMPFiles on Android.

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*.

XMP Toolkit SDK

The repository contains everything needed under Mac OS®, Windows®, UNIX®/Linux®, iOS, and Android.
Note that all source and text files have UNIX-style line endings.

XMP Toolkit SDK contents

The GitHub repository contains the following folders:

`/`	At the root level, the license agreement (BSD_License.txt) and this overview (XMP-Toolkit-SDK-Overview.pdf).
`build/`	Contains Batch file, shell scripts, Makefile and CMake scripts to be used to create XMP Toolkit SDK project files on the supported platforms. Follow the instructions in README.txt to create the projects.
`docs/`	The three-part XMP Specification, the XMP Toolkit SDK Programmer's Guide, the API reference documentation (`API/index.html`) and the XMPFiles plug-in SDK documentation.
`public/include/`	The header files and glue code that clients of the XMP Toolkit SDK must include.
`samples/`	Sample source code and CMake scripts for building sample projects, with the necessary resources to run the sample code. See "Sample code and tools" below.
`source/`	The common source code that is used by both components of the XMP Toolkit SDK.
`XMPCore/`	The source code for the XMPCore library.
`XMPFiles/`	The source code for the XMPFiles library.

`third-party/`
`expat/`
`zlib/`
`zuid/`
	Placeholders for third party source files that 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.

`tools/`	Placeholder for downloading and placing CMake tool and Android tools
`XMPFilesPlugins/`	The header files and glue code of the XMPFiles plug-in SDK and a sample plug-in.

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 are scripts to generate project files for different platform-specific IDE.

On running the scripts, which in turn call the CMake tool, the following project files will be created:

- Project files for MS Visual C++ 2019 will be created at *xmpsdk*>\samples\build\vc16\.
- Project files for Xcode 12.4 will be created at <*xmpsdk*>/samples/build/xcode
- Makefiles for GCC 4 will be created at <*xmpsdk*>/samples/build/gcc

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 of production. This tool helps you determine whether a file is well-formed and to understand its structure.

`Xmpcommand`	Performs basic XMP actions such as to get, put, and dump. It 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

From **VERSION 2022.02** onwards this document is being deprecated. For the list of changes in a version please refer to releases page of XMP Toolkit SDK GitHub repository.

VERSION 2021.10

- Security Fixes

VERSION 2021.08

- Security Fixes
- Few syntax corrections- like missing semicolons at few places (that code was mostly unreachable)
- Copyright note changes to comply with BSD License terms & conditions- across the codebase
- Broken license link fix in the Readme.md file
- Implementation of kXMPFiles_OpenOnlyXMP flag for MPEG4

VERSION 2021.07

- Security Fixes
- Write Exif 2.3.1 Time Zone Metadata - XMPFiles Should Read/Write
- Removal of words Master/Slave/Blacklist/Whitelist from codebase
- cmake scripts use --version instead of -dumpversion to get complete gcc version string
- Updating ReadMe.txt

VERSION 2020.01

- CMake Upgrade: Version 3.15.5.
- Compiler Upgrade: Xcode 10.2, Visual Studio 2017 (VC 15) and Linux gcc 4.8.2.
- Added iOS build support for dynamic libraries, in addition to the previously supported static libraries on iOS.
- Android build support.
- Dropping support of Windows 32 bit.
- Added support of compressed SVG file in XMPFiles.
- Added "http://www.day.com/jcr/cq/1.0" and "http://www.day.com/dam/1.0" as registered namespace with "cq" and "dam" prefix respectively in XMPCore.
- Added an interface in XMPCore IMetadataConverterUtils, for converting SXMPMeta object to new XMPCore (IMetadata and vice versa).

VERSION 2016.07

- CMake Upgrade: Version 3.5.2

- Compiler Upgrade: Xcode 7.2.1, Visual Studio 2015 (VC14) and linux gcc 4.8.2
- iOS build support for XMPFiles component.
- New DOM-based APIs are added in XMP Core to access metadata tree hierarchy. See *Addendum for the XMP Toolkit SDK Programmer's Guide for details.*
- Added support XMP/metadata in utf-8 encoded uncompressed SVG files in XMPFiles. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Added support of GIF handler in XMPFiles. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Added support for the reconciliation of iXML's TRACK_LIST and its components to get microphone data for the customers.
- Added support for IFDs of type 13 in TIFF.
- Added support for MPEG4 videos shot from Google Nexus 5 camera.
- Restructured the implementation of XDCAM Handler to support File Access Mode (FAM) and Simple Access Mode (SAM)
- Added inbuilt support for iXML namespace in XMPCore.
- Modified the alias tiff:ImageDescription for dc:description as language alternative array instead of simple property.
- Bugs Fixes in XMPFiles and XMPCore.

VERSION 2014.12

- Compiler Upgrade: Xcode 5.0.2, Visual Studio 2012 (VC11)
- Added support for video files shot using GoPro cameras in the MPEG4 handler in XMPFiles.
- Optimize File Layout
- XMP Files now provides a flag *"kXMPFiles_OptimizeFileLayout*" that can be provided with *OpenFile()* API to ensure certain optimizations while updating the file. This is currently supported only in MPEG4 Handler. See *XMP Toolkit SDK Programmer's Guide* for details.
- Added support in the MPEG4 handler in XMPFiles for repairing the ill-formed structure in all MPEG4 files. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Enabled the import of Timecode information for all ISO-based MP4 files. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Added support for PSIR marker larger than 64K in JPG files. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Enabled read/write support in XMPFiles for JPEG files with multiple EXIF APP1 markers. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Added support for spanned clips in the P2 handler. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Added support for parsing and reconciling the iXML chunk in Wave files in the WAVE Handler. See *XMP Toolkit SDK Programmer's Guide and XMP Specification Part 3* for details.
- Bugs Fixes in XMPFiles.
- Fixed memory leaks in several file handlers and native support files in XMPFiles
- Fixed security issues in both XMPCore and XMPFiles.

VERSION 2013.06

- Upgraded Expat lib to 2.1.0 (from 2.0)
- Instead of Providing Projects for different platforms, the projects are now created through the Cmake tool.
- Support of long file paths on windows
- Postscript handler enhanced, now it supports packet expansion and new packet insertion
- New API GetAssociatedResources() in XMPFiles to retrieve the list of resources associated with the open file.
- New API IsMetaDataWritable() to report if the metadata can be written to the open file, especially useful for folder-based handlers.
- Progress Notification feature to provide periodic progress-report notifications for write and update operation.
- Error Notification feature which allows the client to register an error-notification callback function, which can be used to suggest an intention for recovery from the error.
- iOS build support for XMPCore component.
- Added versioning concept in XMP Plugins, now the same plugin can be made to work for previous and future XMP Hosts.
- New APIs added to Plugin SDK – importToXMP(), exportToXMP(), getXMPStandard(), checkFormatStandard(). Refer *XMPPlugin* doc for more details.
- Bugs Fixes in XMPFiles.
- Multithreading bug fixes in Plugin SDK.

VERSION CS6

- Compiler Upgrade: Xcode 4.x, Visual Studio 2010 (VC10)
- A plug-in SDK for XMPFiles that allows you to add custom file handlers, or replace existing file handlers with custom ones in XMPFiles
- A new version of the file handler for RIFF-based formats that also supports AIFF. The AVI handler still uses the old code, but WAVE and AIFF are based on the new file handler.
- New functions in XMPFiles that allow you to use your own I/O code in XMPFiles
- Support for Exif 2.3 properties
- A new XMPFiles function, `GetFileModDate()`, that retrieves the most recent OS file modification date for the associated metadata of a file
- Many bug fixes and minor improvements in XMPFiles

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 more thread-friendly, using a multiple reader/single-writer locking mechanisms. See *XMP Toolkit SDK Programmer's Guide* for details.
- The XMPUtils method *AppendProperties* has been deprecated and replaced by the method *ApplyTemplate.* See *XMP Toolkit SDK Programmer's Guide* for details.
- 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 have no timezone value. 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 ability 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 more stability and support for the XMP reconciliation of ID3v2 Tags. See *XMP Specification Part 3*.

  XMPFiles begins to support 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 support 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 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.
- Bug fix: 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.