OpenML 1.1 SDK Release Notes

These release notes provide information for the OpenML 1.1 SDK release.

Summary:

About OpenML 1.1 SDK

The OpenML 1.1 SDK implements the OpenML 1.0 Specification.

OpenML 1.0 is a standardized cross-platform programming environment used for capturing, transporting, processing, displaying, and synchronizing digital media. Any developer or company can freely download the OpenML Specification & SDK and implement and ship products using OpenML completely free of charge and royalty.

OpenML enables multimedia authoring application developers to efficiently develop and deploy cross-platform applications that interoperate with a wide selection of systems and peripherals, enables hardware vendors to leverage the efforts of a larger base of application developers, and offers end-users increased functionality across a wider range of platforms.

About This Release

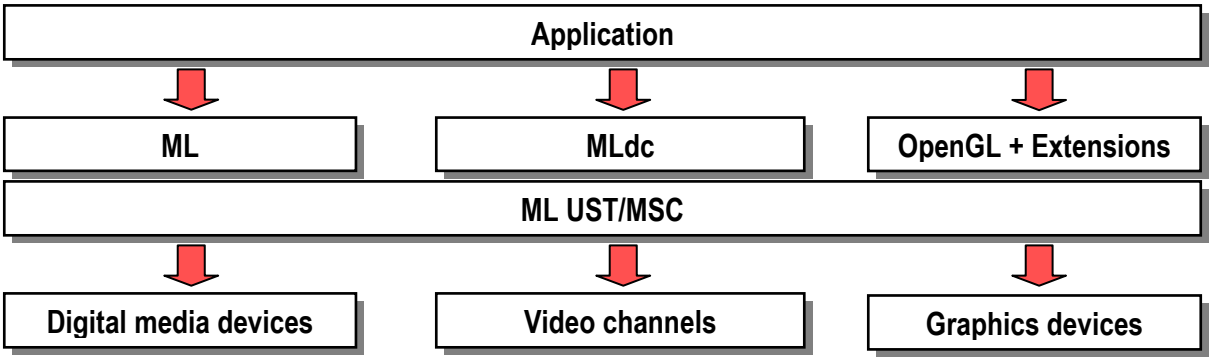
This release delivers the media library component of the specification, providing a low level API for configuring and controlling video/audio device input, output and processing features. The SDK also delivers a UST source module used to synchronize multiple media devices to a common clock source.

The OpenML 1.1 SDK contains: executable libraries and utilities to enable OpenML applications to run on Linux, IRIX® and Windows systems; source and header files for the libraries to enable new OpenML applications to be built; source code for example OpenML applications and audio and video device modules which can be used as a starting point for building commercial applications; and full documentation.

The Khronos Group is a member-funded industry consortium focused on the creation of open standard APIs to enable the authoring and playback of dynamic media on a wide variety of platforms and devices.

SDK Overview

The OpenML SDK is defined by a set of API's that are described in the following paragraphs.



ML

The OpenML Media Library API (ML) provides an application programming interface to the digital media I/O, digital media data conversion, and digital media synchronization facilities in an OpenML programming environment.

The primary functions of ML are to:

- Support asynchronous communication between an application and media devices such as video input/output, audio input/output, and graphics.
- Provide synchronization primitives that give applications the ability to correlate multiple digital media streams and coordinate their presentation to an end user
- Provide processing capabilities (transcoders) for digital media streams
- Provide device control and device capability queries
- Provide buffering mechanisms that support the smooth delivery of digital media and obtain the best possible performance on a given system

Notice: ML system no longer relies on the "mldaemon" tool, and the daemon is no longer distributed.

MLdc

The OpenML Display Control API (MLdc) provides an interface to initialize, set and change parameters to control video output devices and to set up channels in the display screen(s). The display may be a desktop screen or another device such as a special studio monitor.

This component will provide control for:

- Screens and channels
- Monitor information
- Video output channel position
- Video format queries and settings
- Vertical retrace rate
- Interlaced vs. progressive scanning
- Gamma ramps control
- Video synch sources
- Genlock state notification

Notice: The current release of the SDK does not include MLdc libraries.

MLust

This component provides precise timing services that allow synchronizing video, audio and graphics streams making sure that all media streams start at the same time and stay in sync. The Unadjusted System Time (or UST) is a high resolution 64-bit monotonically increasing counter that is available throughout the system. In addition, each media device in the OpenML environment is expected to maintain a Media Stream Counter (or MSC) that is incremented for each media event that occurs for that device.

This component provides:
- A system-wide high resolution time base called Unadjusted System Time (UST);
- A media stream counter (MSC) for each media device; and
- A graphics swap buffer counter (SBC) specific to graphics channels.

OpenGL extensions

Extensions to OpenGL add capabilities for synchronization with other parts of the system, for improved rendering performance and quality, and for improved treatment of video formats. OpenML proposes a set of extensions to strengthen OpenGL's rendering and video integration capabilities. These extensions are not part of the SDK and are provided by independent software and hardware vendors.

The recommended extensions are:
- Synchronization using UST/MSC information
- Reading and writing of interlaced video images
- Direct support for video pixel formats such as YcrCb
- Extended texturing functionality for compositing support
- Access to programmable geometry and shading hardware
- Asynchronous processing of lengthy pixel operations

Software Requirements

The ML SDK provides a common building environment on both Windows and Linux. This environment is command-line based. The specific requirements for each platform are listed below. For more information on building the SDK, refer to the "OpenML ML 1.0 SDK Developer's Manual" (unchanged for this 1.1 release). This document is available in the SDK, in the file *oss/doc/ml-dev-manual.pdf*.

Linux Requirements

The ML SDK supports the following operating system and compiler versions:

OS	RedHat Linux version 8.0
Compiler	GNU gcc version 3.2
Other development tools	Other compiler versions may also work, but have not been tested

Windows Requirements

The ML SDK supports the following operating system and compiler versions:

OS	Windows XP Professional
Compiler	Microsoft Visual Studio (C++), version 6.0, service pack 5
Other development tools	cygwin development environment, version 1.3 or later

SDK Content

ML

Windows Contents

The Windows SDK is contained in the "OpenML" directory, which is located in the top-level directory chosen during installation – by default, this is

C:/Program Files/OpenML.

The "OpenML" directory contains the following sub-directories:

bin — pre-compiled versions of the "mlquery" utility and the "mlAudioCapture" sample program.

include/ML — header files for the "ml" and "mlu" libraries.

lib — pre-compiled versions of the "ml" and "mlu" libraries. These are the "IMPLIB" libraries only, i.e.: the libraries required for linking an ML application program. The DLLs are described below.

examples — source code and Microsoft Visual Studio project files for ML sample programs.

man — documentation, in the Unix standard "man page" format, for functions provided by the "ml" and "mlu" libraries (this information is also provided as a PDF document, in the "docs" directory).

src — contains a zip file of the ML SDK source tree. See "SDK Source Tree Contents" on page 5 for details.

docs — the ML User Guide and an errata to that guide, and PDF versions of the "man page" documentation for the "mlu" utility library and the DI/DD API (for ML module developers). Note that Chapter 1 of the ML User Guide is a good "Beginner's Guide" to programming with ML.

In addition to these directories, the SDK installation process places files in the following system directories:

windows/system32 — contains the DLLs for the "ml" and "mlu" libraries, required to run any ML program (including the "mlquery" utility). This directory also contains the `audiofile` DLL, which is required for certain sample programs. Since this system directory is automatically searched for DLLs, no extra configuration is required before using ML programs.

windows/system32/openml/mlmodule — contains pre-compiled versions of the sample modules. This directory is searched by default by the ML libraries, so no extra configuration is required before using ML on your system.

Linux Contents

The Linux SDK is contained in the "ml" sub-directory where the package was installed. This directory contains the following files and directories:

oss — source tree for the ML SDK, as described below.

linux — pre-compiled versions of the libraries, tools and modules. Note that if the SDK is re-compiled, the compiled components will be placed in this same directory structure, thus over-writing the pre-compiled versions distributed with the SDK.

The SDK does not contain a copy of the libaudiofile header files and shared library. This is generally pre-installed on Linux systems; however, if your system does not have the audiofile and audiofile-devel rpm files, you may find them at: `http://oss.sgi.com/project/audiofile`.

SDK Source Tree Contents

The source for the SDK, in the "oss" directory, consists of the following:

build, make — makefiles and scripts for the multi-platform building environment. The build environment is command-line based (ie: Unix-like) on all platforms. For more information, refer to the "OpenML 1.0 ML SDK Manual" PDF file.

lib/ml, lib/mlu — source for the ML SDK libraries "ml" and "mlu". The "ml" library implements the core of the ML API, as defined in the 1.0 specification. The "mlu" library implements additional utility functions to ease the process of developing an OpenML ML application.

tools/mlquery — source for the ML query utility. This utility is a command-line program that uses ML services to report on the devices available on the system; it can also report on the devices' properties.

devices/nullxcode, devices/ossaudio, devices/v4l, devices/winaudio devices/ustsource — source for the sample devices ("mlmodules") distributed with the SDK. "nullxcode" is a sample transcoder, and compiles on Linux and Windows. "ossaudio" and "v4l" are sample audio and video modules respectively; they are built on top of Linux APIs, and can be built on Linux only. "winaudio" is a sample audio module built on top of the Windows WAVE API, and can be built on Windows only. "ustsource" is an example of a module that provides a system-wide UST source; it can be built on all platforms (note however that on Windows, it is not pre-installed in the system directory. In order to use this module, it is necessary to compile it) Note that the modules included with the SDK are intended as examples only, and are neither fully functional nor fully compliant with the 1.0 specification.

man/man3dm — man pages for the API implemented by the "ml" and "mlu" libraries, as well as man pages on the Device-Independent/Device-Dependent (DI/DD) API, used by module developers.

doc — documentation, in PostScript and PDF formats. The user guide (especially Chapter 1) serves as a good introduction to ML programming; it should be used in conjunction with the OpenML specification document (available from the Khronos web site: http://www.khronos.org/openml/spec.html), which serves as a reference guide.

examples — sample command-line programs that use ML to perform audio or video I/O. In order to use these programs, ML modules must be available (either sample modules distributed with the SDK, or production modules distributed with OpenML-compliant I/O hardware).

The `examples/mlAudioCapture` directory contains a sample audio capture program that uses a graphical user interface; it allows capturing audio at various sample rates and writing the result to a ".wav" file. This sample compiles on Windows-based platforms only.

In addition to these directories, the SDK contains a few files at the same level as the "oss" directory. Of particular interest are:

LICENSES — license information for the OpenML ML SDK source code.

SET_ENV — script (for tcsh shells) to set environment variables required to build the SDK.

GNUmakefile, BUILD_LINUX — top-level makefile and build script, to build the entire SDK. See the section "Building the SDK".

Finally, the SDK contains a local installation directory, which is where header files and binaries are installed when you re-build the SDK source tree. This directory – which is created at build-time if it does not yet exist – is at the same level as the "oss" directory, and is named either "linux" or "win32", depending on the platform. Its contents are somewhat different depending on the platform, but are generally as follows:

- an "include" directory, for all "ml" and "mlu" header files;
- a "lib" directory, for the "ml" and "mlu" libraries ("IMP" libraries only, on Windows);
- a "bin" directory, for the ML tools (and DLLs, on Windows);

- a "man" directory, in which all man pages are installed;
- an "examples" directory, in which the source code for the sample programs is installed – note that this does *not* contain pre-compiled binaries.

Changes from the 1.0 SDK

The changes from the 1.0 SDK release are as follows:

- New constants and types added for video device capabilities.
- A few small bug fixes to the SDK code.
- Some additions to the Specification Errata document.
- All copyright notices changed to allow the code to be Open-Sourced.

References

OpenML 1.0 white paper — Document providing an overview of the OpenML digital content and playback environment.

OpenML 1.0 Specification — Specification document providing the OpenML 1.0 architectural overview, specifying the various programming interfaces that comprise the OpenML programming environment, and enumerating and defining each of the function calls that comprise those interfaces.

ML User Guide and errata — Complete ML SDK programmers guide.

ML Beginners Guide — Compact document providing relevant information to quickly get started in developing ML applications and tools.

mlu man pages — Introduction to the ML "mlu" utility library.

DI/DD API man pages — Introduction to the interface between the device independent (DI) and device dependent (DD) portions of the ML component.

OpenML 1.0 ML SDK Manual — Developer's manual describing ML SDK installation, content and build information.

Release Notes — Document providing the OpenML 1.1 SDK release description and content.

All documents are available from the Khronos website or from the OpenML 1.0 SDK distribution.

NOTES: