OEM Application Programming Guide and Customer Support Information

For use with MFCBR_OEM version 26.0

Revised on June 3, 2015

Table of Contents

Introducing the CopySense OEM API

Technology Overview

Audible Magic's CopySense content identification technology is available for integration with your own client application. The CopySense content identification OEM API library makes it easy for developers to incorporate content identification into any application or device. The basic content identification system consists of a Client Fingerprinter, an Identification Engine (also known as an ID Server), a Reference Database and an optional Media Information Database.

- The Client Fingerprinter converts the unknown audio into an Audible Magic fingerprint.
- The Reference Database contains fingerprints that have been created from known audio segments.
- The Media Information Database contains information about each of the reference fingerprints.
- The Lookup Engine does the work to compare the unknown fingerprint against the reference database and to either declare that there has been a match, or that no match has been made.

These pieces of the overall solution can be deployed in a number of ways.

The Client Fingerprinter is deployed on a system where the unknown audio is available. This can be a mobile device using a microphone, a client system with access to media files, or perhaps a client system with access to media streams.

The Reference Database can contain a specified set of references selected from Audible Magic's master database of over 13MM song performances, movies, episodic television, and 100+ live TV broadcast channels. Or the Reference Database can be created from custom content supplied by you.

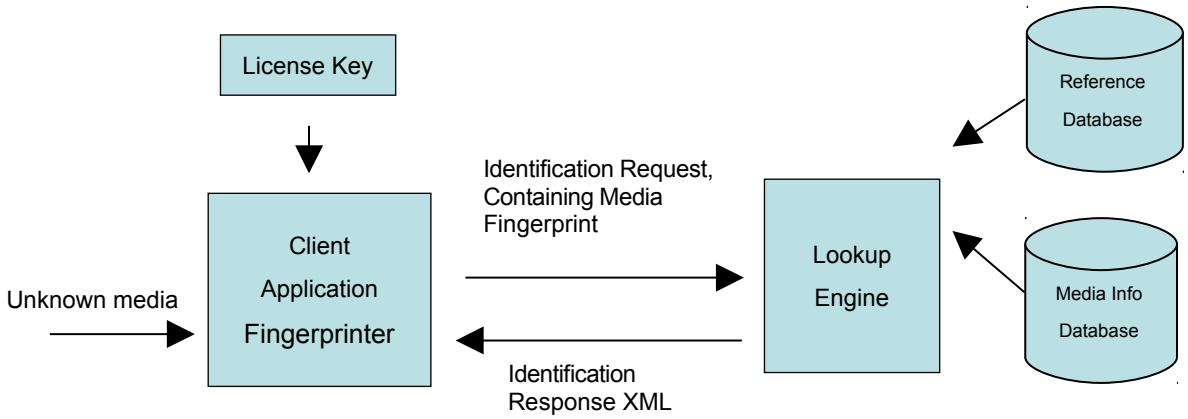
If the Reference Database is supplied by Audible Magic, then we also access our own Media Information Database and your application will receive this information when a match is made. If you create a custom Reference Database then you must supply the corresponding Media Information for each reference fingerprint.

The Lookup Engine can be deployed either remotely or locally on the client device. For instance, a mobile phone, tablet, television, or set top box can host a Lookup Engine and Reference Database suitable for many applications. This "local lookup" configuration is useful when a client application wants to remain in very tight synchronization with a media stream—such as when frame accurate positioning is needed. Local Lookup is also useful when an application needs to constantly scan a media stream for some particular content—such as when an application wants to quickly know when one of a set of commercials begins to play.

Remote Identification Servers are hosted by Audible Magic in a class A datacenter with high availability features. A remote Identification Engine is useful when the size of the Reference Database is extremely large. For instance, it would be impractical to do local lookup with a database of 1,000,000 songs or thousands of movies. Using a remote ID Engine (or ID Server) allows the back end to make use of more powerful computer servers to process each request. It also makes updating the Reference Database a concern of Audible Magic Operations and not of the customer.

LiveTV identification is a special case where the remote identification engines are fed from 100 live TV network feeds in real time. The client can use the match data to retrieve electronic program guide (EPG) data from their preferred EPG vendor. Alternatively, Audible Magic can return EPG data from Rovi or TMS at an additional charge.

In all of these cases the Client Application still uses the same Audible Magic client library. The library is supported on all major computer and mobile platforms.



Additional Tools

1. `amDbBuilder` is a command-line tool from Audible Magic you use to create your own Local Lookup Reference Database (AMDB file) from a list of your own media files. Your application is responsible for deploying the reference database to your client(s). When your application initializes the local lookup it will pass the local filename of the AMDB file to the client library. Each reference fingerprint retains the media filename used to create the fingerprint. When a match is made the local Identification Engine will alert your application, returning this media filename and information you can use to precisely synchronize your application behavior to the media playing. (See the section of this document titled “Application Synchronization to Media”). AmDbBuilder is available in a separate package; please contact AM Support if you need to use this tool.

2. `amSigGen` is a command-line tool from Audible Magic that creates single reference fingerprints from a media file, suitable for use in remote identification. Your company puts in place a process to create reference fingerprint files and metadata files and to FTP

these files to Audible Magic, where they are automatically ingested and deployed to our Remote Identification Servers. One field in the metadata files is a digital asset identifier that will be returned to your application as part of an identification response XML.

Please note that the reference fingerprints you submit to Audible Magic are deployed only for use by your own applications. Unless you have specifically made arrangements with Audible Magic, your reference fingerprints will not be used by any other customer.

OEM API Functionality

Below are outlined two typical implementation scenarios. Other deployments are possible and our client support team would be happy to discuss other options with your team.

For Remote Identifications your client application uses the OEM API as follows:

1. The application creates an MFMediaID object.
2. The application registers an error callback with the MFMediaID object.
3. An API call generates a fingerprint from a portion of the unknown audio or video. (Video files can be identified based on their soundtrack and optionally their video track.)
4. The library produces a small XML package, known as an ID request. The XML contains the media fingerprint and associated information, all encrypted.
5. The library sends the ID request XML to Audible Magic's ID server farm via HTTP.
6. When the library receives a response from the remote Identification Engine, the results are returned to your client application as clear-text XML.
7. Your application parses the returned XML text as needed.

For Local Lookup your client application uses the OEM API as follows:

1. The application creates an MFMediaID object.
2. The application registers an error callback with the MFMediaID object.
3. The application supplies the MFMediaID object with a Reference Database
4. The application calls StartListening on the MFMediaID object and the library begins monitoring the device microphone and continuously generates a fingerprint from the unknown audio stream.
5. When your application wants to identify the most recent audio, it calls Identify Now. This is a blocking call so your application should not call this from the thread that handles your user interface activity.
6. When IdentifyNow completes it returns a Id Response object. You can use various API calls to examine the Id Response object. Your application will check whether the result is a positive id or no-id and to retrieve information associated with a positive id.

Note: The routines described in this manual were developed primarily by an Audible Magic subsidiary called Muscle Fish, so you will find the prefix “MF” on many names of objects, functions, error codes, etc.

Accuracy

There are two key measures of accuracy: False Positives and False Negatives.

False Positives occur when the technology incorrectly identifies a media stream as one thing when it is really something else. In all applications, False Positives are very, very bad. In consumer applications, a False Positive will cause the consumer to see information or actions that are inappropriate to their expectation at the time. In royalty-tracking applications, False Positives will cause the wrong person or organization to be paid. In copyright compliance applications, False Positives will cause the wrong usage license to be applied.

Audible Magic services are all tuned to have essentially zero False Positives. We believe that application developers have enough to do without having to second-guess an identification. We test our services in-house to prove a False Positive rate of less than 1:1,000,000 and, practically speaking, our customers report that our services have zero False Positives in audio identification.

False Negatives occur when the technology fails to make an identification when one should be made. Because it is based on perceptual characteristics, Audible Magic's technology has a very, very low False Negative rate. In environments where the media stream is clean—such as embedded digital stream capture or media file identification—the technology will correctly identify 100% of the time. When the media signal is degraded by transmission artifacts—such as AM/FM radio noise or extremely high lossy compression—then our positive ID rate may drop to 99%. In analog microphone capture where the audio can be highly contaminated by other noise, our positive ID rate will be lower. In a typical home living-room TV-watching environment with a microphone capture from 10 feet away, we will still have a positive match more than 98% of the time. However, if in that same living room setup a vacuum cleaner is running next to the TV, then our ID rate will be lower.

False Negatives might also occur when attempting to identify audio from a microphone when the audio source is a speaker *on the same machine*. On many devices, this scenario works well, but some computers have an echo-cancellation feature that removes the sound being played through the speaker from the sound that the microphone is capturing. This echo cancellation results in False Negatives. If end users will be running the application on such a device, the application developer should either disable the echo cancellation if possible, or else avoid this embedded-speaker-to-embedded-microphone scenario.

In all cases, however, our False Positive rate remains at zero.

The good news with False Negatives is that your application can try the identification again. While a wrong identification (a False Positive) is difficult to recover from, a False Negative usually just means that your application has to try again.

Lookup Type

Identification can be done in many different ways. Some applications have only a short snippet of media to use, others have long samples. Some applications want to identify a media file of a complete work. Choosing a Lookup type is a matter of making tradeoffs between cost and ID rate. The difference between each type is the intensity of the scan performed on the unknown fingerprint. The least expensive lookup type requires that the application know precisely where the work begins. Certainly the most intensive lookup type can be used for any application, but the cost may be prohibitive. Audible Magic offers several less intensive searches that offer our customers the ability to select the right price / performance trade off for their particular situation.

You must work with Audible Magic Client Services to select the appropriate identification type for your particular application. The particular identification type is set by Audible Magic, and cannot be changed by the client application.

File Identification

These options are useful for applications that have a file containing audio to be identified. The file might commonly contain an entire work (such as an entire movie or song), be part of a work (such as a short video uploaded to a social media site), or be a capture of a longer stream (such as five or ten minutes of a radio or TV broadcast).

1. Type File ID (audio fingerprints only): A “Basic Lookup” mode, where the match is made at an offset near the start of the unknown media sample. Only a small portion of the unknown fingerprint is tested for a match. In this mode, an application will typically fingerprint the first 60 seconds of a media file for lookup. The Lookup Engine selects one segment within that fingerprint and examines it for a match with the same relative location in each reference fingerprint.

   For File ID identification to work, it is critical that the beginning of the unknown media sample correspond within a few seconds to the beginning of the original song or video.

   Type File ID is appropriate for applications that need to identify audio from a CD, DVD, or from a disk file ripped from a CD, DVD, or DRM-protected file, or from an Internet stream that signals the start of each new play.

2. Type Scan Mode (audio fingerprints only): “Scan” mode will try to match all segments in an unknown fingerprint to a single small portion from the beginning of each reference fingerprint. The client often uses the entire media sample for lookup.

   For Scan Mode to work it is critical that the unknown sample contain at least the first 60 seconds of the master song or video somewhere within it.

   Scan Mode is appropriate for applications that need to identify a set of works used in a long portion of a continuous media stream, such as a five minute capture of radio or television airplay.

3. Type Clip Lookup (audio or video fingerprints): “Clip Lookup” is conceptually the reverse of Type Scan Mode. Clip Lookup will examine one small segment within the unknown

fingerprint and determine if it matches ANY portion of any reference fingerprint.

A fingerprint of a sample of unknown media is sent to the lookup engine. From this fingerprint the lookup engine will select an arbitrary segment to use for the search. The selected segment is scanned against all segments of every reference fingerprint to determine if it will match anywhere. The response will specify the location and duration of each match within the tested media sample.

Clip lookup is appropriate for applications that only have an arbitrary piece of the work, such as User Generated Content web sites or customers that have just a portion of audio to identify. Depending upon the service the unknown sample will contain more than 21 seconds of audio, or more than 5.2 seconds.

4. Type Embedded Clip Lookup (audio only): This is the most intensive search offered by Audible Magic. In this type the ID server will compare ALL segments within the unknown fingerprint to determine if ANY segment matches ANY segment of a reference fingerprint.

   This type of lookup is expensive and is often used in legal investigations where cost is not an issue. Of course you may have an application that requires this kind of intensive scrutiny and Audible Magic is happy to offer such a service.

Live Capture Identification

These options are useful for applications that capture live audio via a microphone or a line-in source, or that intercept and render a digital media stream. In all these cases the audio is typically captured and fingerprinted continuously, and periodically a lookup is done to identify the most recent audio.

Please also read the section “Application Synchronization to Media.”

The AM library has the capability to capture audio from a device microphone or line-in source. If your application needs to monitor live audio from another source then your application will have to take responsibility for the intercept and rendering to PCM. See the API call `MFListenToSamples`.

5. Clip Lookup – In this mode a small sample of the current audio capture buffer is compared against all parts of a reference database to determine if it matches any part of any reference fingerprint.

   Clip lookup is appropriate for applications that need to synchronize actions to particular places in a work. Reference fingerprints of the works to be synchronized are typically computed days or hours before they are needed by the application.

6. Live TV – Audible Magic offers a remote identification service that can provide EPG data for any audio sample. Typically an application on a mobile device, set top box, or television will use this remote identification service to determine what a user is currently watching. As of this writing the Live TV service supports over 100 broadcast and cable channels in the United States.

LiveTV is appropriate for applications that want to trigger actions based on EPG data about the television show being watched. If you need to identify broadcast channels outside of the U.S. contact Audible Magic Support.

Application Synchronization to Media

Applications may want to have certain actions happen at certain points in the media stream. For instance, an application may want to present a quiz to a user when a certain point in a television show is reached. The application does not want to present the quiz too early or too late. The application also wants to correctly present the quiz even when the viewer makes use of special functions such as fast forward, skip back, and other time shifting features. These applications want to synchronize their actions to the media stream.

Applications that track media streams need to calculate "where in the media stream am I right now." Of course, this is not a static answer, because time continues to march on while the actual identification is being done. To achieve these objectives, the application needs to synchronize its actions to the media stream by calculating the current location in the reference fingerprint. In other words, the application needs to know where it is in the show right now. To determine the current location in the reference fingerprint, an application must use the following formula:

Current Location in reference fingerprint = Now – MatchOffset + SigOffset where:

- **Now** = Current Epoch Time
- **MatchOffset** = Epoch time when lookup is performed
- **SigOffset** = Offset in seconds where match starts in reference fingerprint.

Note: These time values may be Unix epoch time and *must be stored as doubles, not floats*.

Diagram: Application to Display Quiz at 216 seconds into Show

ID Server Reference Databases

Audible Magic maintains reference databases, with a high level of data integrity, and provides positive identification information for audio and video samples from copyrighted music performances, TV shows, and movies. Remote ID servers can use the entire Audible Magic catalog, or just a subset. Using a subset requires a special set up known as a "custom database."

It is also possible for you to create a private database that the Audible Magic servers will use. This is known as a "Submitted Content Database." Audible Magic provides a utility, `AMSigGen`, to create these reference fingerprints.

For Local Lookup, Audible Magic provides a utility, `amDbBuilder`, which allows you to build your own Audible Magic database containing reference fingerprints for content that is relevant to your particular application. This database can then be used on the device itself.

Scalability

The Remote Identification service architecture is modularized and highly scalable. The application is stateless, so it can easily scale horizontally to handle a large number of concurrent identification requests. Our proprietary identification algorithms allow us to split the database into any number of shards without performance penalty, so we can meet whatever response time you need on any size reference database.

The response time for your particular application will depend upon factors specific to your business needs; please consult with Audible Magic to discuss how we can meet your requirements.

For a Local Lookup configuration the maximum size of the database is set by Audible Magic in the configuration information accessed by your license key, but this size can be as large as your application requires. Large local databases require more RAM and more CPU from the local client.

Using Audible Magic's OEM API SDK

Baseline Recommended Configuration

Non-mobile platforms

Client computers are often out of the control of the application developer. However, any current PC on the market will meet the needs of a remote lookup. We recommend a computer with a minimum configuration of:

Pentium IV CPU

- 2 GB RAM
- 2 GB free space on hard disk (or 75 MB if not dealing with video files)
- A working Internet connection (DSL or better)

In a local lookup configuration, the performance characteristics of the device will dictate the maximum size of the local database and the frequency of lookup attempts. Please consult with Audible Magic.

For application software development, we recommend:

- The OEM API has also been used successfully with other programming languages, including Microsoft Visual Basic, C#, and Perl.
- Note for Flash developers: To our knowledge, it is not possible to call a function in a C library from the Adobe Flex SDK. However, it is apparently possible for a Flash application to communicate via a socket with a C (or Java) application. It should therefore be possible for you to modify the programming examples supplied with the Audible Magic OEM SDK in order to support such communication, sending the necessary parameters from your Flash application to the modified C or Java program, which would invoke the Audible Magic API and then return the result to the Flash application as needed.
- A note about HTML5 – The HTML 5 specification contains APIs to access client devices such as microphones. Using these APIs it could be possible for your HTML5 application to gather samples from the user microphone and pass those samples to a central processor in the cloud where the actual identification would be performed.

Mobile platforms

For iOS devices an iPhone 3GS or later is recommended. All iOS platforms capable of running iOS 4 or higher, through iOS 7, are supported. We assume the use of Apple's Xcode, version 4.0 or later, for iOS development. For further information, see the section "Installation Guide and Programming Examples."

Android devices come in variety of models and it is harder to specify a minimum supported configuration. In general, all Android tablets and phones work well except for the very-low-end models. Note that some Android tablets do not offer a local microphone.

For Android platforms, Android 2.3 or later is required.

In a local lookup configuration the performance characteristics of the device will dictate the maximum size of the local database and the frequency of lookup attempts. Please consult with Audible Magic as needed.

Embedded platforms

The Audible Magic SDK is written in standard C code and can be ported to any platform that has a suitable tool chain. Contact Audible Magic business development to discuss a port to your platform.

API specifications

▪ Program development tools capable of using a C library. The OEM API includes interfaces for C and Java. (The latter is currently intended for Android developers only.) The SDK contains several C programming examples that were compiled with Microsoft's Visual Studio 2010 or later on Windows, or gcc on Linux and Mac OS X.

Client OS Version	The current version of the SDK has been developed and/or tested on the following OS versions, but this is not a complete list of versions that work. Windows 7; Mac OS X 10.9, 10.10; Linux kernel 2.6.15, 3.2.0; iOS 8; Android 2.3.
Libraries	**Audible Magic C library:** • Windows*:* mfcbr_oem • Linux*:* libmfcbroem.so or libmfcbroem.a, each in a regular version and a smaller version that doesn't use libstdc++ • Mac OS X*:* libmfcbroem.dylib or libmfcbroem.a • iOS*:* libmfcbroem_ios.a • Android: libmfcbr.so (including Java [JNI] wrappers) **Video fingerprinting library:** Not included with the SDK. Contact Audible Magic if you require fingerprints of video image data.
Executables	The following executables are packaged with, and required for, the SDK: • Windows: `ffmpeg.exe` • Linux : `ffmpeg` • Mac OS X: `ffmpeg` • iOS: N/A • Android: N/A
Data Format	Encrypted XML in request sent to ID Server Clear XML in result returned to application
Service Delivery Model	Application Service Provider, Software Provider, Technology Licensor

Size of ID request XML package	Audio-only requests (including soundtracks in video): approximately 800 bytes per second of audio. Audio-plus-video requests (from video files): Variable, but typically between 800 and 1500 bytes per second of media data. Microphone captured audio requests are 1600 bytes per second. Most applications use 6 seconds of audio for a total of about 20KB per request.
Client Resource Load	**Fingerprint Generation** The client device library will consume CPU in creating the unknown fingerprint. The CPU for this activity is relatively small on modern devices. For example, on some MIPS based embedded device chips we have measured the continuous fingerprinting load as less than 4% of the CPU dedicated to application processing. For file based applications the CPU to fingerprint a file will be consumed as fast as possible, limited by the resource sharing algorithms of the underlying OS. Our customers have had no problems with this to date. On a mobile device battery life is always a concern. Our testing has shown that continuous fingerprinting from microphone input has a minimal affect on battery life. In one test, over an 8 hour period our library consumed less than two additional percentage points of battery life. The bottom line: for mobile devices, do not call MFMediaID_StopListening unless your application will not do content identification for at least 15 minutes. **Lookup Engine** If the lookup engine is remote then the only impact on local resources is the transmission of data to / from that server. In using our library each identification request is about the same load as uploading a 20KB file to a server and receiving a small web page in response. When local device lookup is used then the amount of CPU used will depend upon the size of the reference database. The CPU will be consumed as quickly as possible each time an identification is attempted, relying on the underlying OS to share resources equitably. To date this has not been a problem for our customers.

Finger-printing Speed	Audio: On a 2-GHz CPU, unknown audio in PCM format is fingerprinted approximately 100x as fast as real time. A fingerprint for a typical lookup request is created in a fraction of a second. Mobile and embedded applications will normally fingerprint an audio stream continuously. For typical hardware the continuous fingerprint generation will consume less than 4% of an application CPU. Video: Not included with the SDK. Contact Audible Magic if you require fingerprints of video image data.
Lookup Speed	For remote lookup applications the response time for the lookup is set as part of the Service Level Agreement in your contract. For local database lookups the response time will depend upon the speed of the local device CPU and the size of the reference database you deploy. Audible Magic Support is happy to consult with you on this.

Supported Media Formats on Server / Desktop Deployment

The desktop / server version of the library can support a wide variety of audio and video file formats and codecs. Please contact us if you have questions or difficulty with a specific format.

The desktop / server library uses, among other things, open-source software from the FFmpeg project to handle many audio and video file types. **In general, it will handle any kind of file that the FFmpeg executable supports. Running the command line "ffmpeg --formats" will display a very long list.**

A media file typically has both a file format (like AVI) and one or more codec data formats (like WMV3 and MP3). Some file types, like AVI, are containers; they can specify a large number of different codec data formats within them. It is important to understand that supporting a container does not necessarily mean supporting any conceivable codec whose data can go in the container, and likewise supporting a codec does not necessarily mean supporting every container that the codec's data can appear in.

Files protected by Digital Rights Management (DRM) cannot be fingerprinted.

Audible Magic has done extensive testing with the following common combinations of file type and codec (but this is not a complete list of supported types).

Video

Type	Video Codec	Audio Codec
.avi	mpeg4	mp3
.avi	h264	mp3
.avi		ac3
.avi		aac
.flv	vp6f	mp3
.flv	vp61	mp3
.mov	h264	aac
.mov	h264	pcm
.mov	svq3	qdm2
.mp4	h264	aac
.mpg	h264	mp2
.mpg	mpeg2	mp2
.vob	mpeg2	mp2
.vob	mpeg2	ac3
.vob	mpeg2	pcm
.wmv	mpeg4	wmav2
.wmv	vp6f	mp3
.wmv	wmv1	wmav2
.wmv	wmv2	wmav2
.wmv	wmv3	wmav2
.3gp		amrnb
.3gp		amrwb

Audio

Type	Codec
.mp3	mp3
.wma	wma
.wma	lossless
.wav	pcm
.m4a	aac

(For simplicity, the above list uses filename suffixes to denote the file type. However, the software does not use suffixes to ascertain the file type, because in the real world files are often incorrectly named.)

Decoded audio should, at a minimum, be 22050 Hz, 16-bit samples, little-endian, mono. If lossy compressed, it should be at least 56kbps for mono audio.

The desktop SDKs also support audio input from a microphone or line-in.

Supported Formats on Mobile Platforms

Live audio (e.g., microphone) input can be automatically captured, analyzed, and identified through, for example, the `MFMediaID_IdentifyNow` API.

On iOS and Android we support identification of media files via the `MFMediaID_IdentifyFileFromSamples` API. The input must be PCM 16-bit, little-endian, monophonic data. If an audio media file is not in this format, then your application will need to render the files to this format. No other formats are supported on iOS or Android.

The License Key

Each time your application uses the Audible Magic library, it must present a license key that was provided to you by Audible Magic. The license key is a string containing a globally unique identifier (GUID) and supplementary descriptive text. The library sends the license key to a remote authorization server, which in turn returns to the library the configuration information corresponding to your customer contract with Audible Magic.

The configuration information contains a number of settings that determine how the library will collect and analyze the data. Some features of the API must be explicitly enabled in this configuration information.

When your application creates an MFMediaID object, it must provide that object with the license key. Your application may create many MFMediaID objects with the same license key, but each MFMediaID object will use only one license key.

Some customer applications use several different configuration settings. For instance, one license key may specify that your application will be using a local reference database. Another license key may specify a remote database that is accessed via an Audible Magic identification server. If your application has a need for both local and remote databases, you will be provided with two license keys.

Applications that must run without Internet access can instead use a configuration file containing the same information that the authorization server returns in response to a license key. This configuration file must be obtained from Audible Magic. The application can tell the library the path to the configuration file, or the file's contents can be stored in an in-memory XML string and presented to the library in that form.

How to perform an ID — Overview

The Audible Magic OEM API provides a variety of styles of identification. For example, identification may be performed on either input media files or on live audio input. Identifications may occur in either an on-demand or a continuous periodic basis. Identifications may be sent to and performed on remote Audible Magic identification servers or they may be performed against a database that resides locally on the same device that is running your application.

Naturally, each of these identification styles is accomplished by using specific functions of the Audible Magic OEM API. The following describes the most basic calling sequences required to perform several of the identification styles described above. Sample applications, included with the OEM SDK release, provide more specific and detailed examples of how to use our API to meet your needs. We strongly recommend that you study these samples before embarking on the development of your application.

At the most general level all identification styles share the following steps:

1. Call the SDK's initialization function (`MFGlobalInit`).

> **Note:** Your application must call `MFGlobalInit` once, and only once, before calling *any other* Audible Magic API. In rare cases where you need to specify a file folder that is different than the current working directory, you should call `MFGlobalInitWithPrivateDataFolder)`.

2. Initialize an identification object (`MFMediaID`) using your credentials. Your credentials will be supplied to you by Audible Magic in the form of a license key.

3. Either

3.1 Pass the `MFMediaID` object a path to the media file

3.2 Instruct the MFMediaID object to start capturing the live audio input that you wish to identify and then call MFMediaID_IdentifyNow to trigger an identification attempt.

4. Examine the ID server response: XML that contains metadata identifying your audio or video.

Example 1: Identifying a Media File

1. Initialize the SDK – All applications must call this function once, and only once, before calling *any other* Audible Magic API.

```
MFGlobalInit
```

2.Initialize an `MFMediaID` object with your credentials (i.e., license key).

```
MFMediaID_CreateUsingLicenseKey
```

3. Identify a file and receive an `MFMediaIDResponse` object.

```
MFMediaID_IdentifyFile
```

4. Retrieve a copy of the XML response string from the `MFMediaIDResponse` object.

```
MFMediaIDResponse_GetStringLength
MFMediaIDResponse_GetAsString
```

As an alternative, or in addition to these functions, if one simply wants to know whether input media was identified, one may call

```
MFMediaIDResponse_GetIDStatus
```

which returns, by reference, an enum (integer) interpreted as follows:

```
typedef enum
    {
        MF_MEDIAID_RESPONSE_UNKNOWN,    /* server returned error */
        MF_MEDIAID_RESPONSE_NOT_FOUND,  /* match not found in DB */
        MF_MEDIAID_RESPONSE_FOUND,      /* match found in DB*/
    }
```

5.Destroy the `MFMediaIDResponse` object.

```
MFMediaIDResponse_Destroy
```

6. Parse the XML string, called an ID response. (The returned XML string is in clear text, and parsing it is the responsibility of the client application. See the "Sample Response XML" section of this document.)

7. Steps 2-5 can be called multiple times in succession for different media files.

8. Every OEM API call produces a return code. Be sure your application tests the return code after each function call, and if it is anything other than 0 (`MF_SUCCESS`), take the appropriate action. (See section “Error Codes” in the “Detailed API Reference” below.)

9. Destroy the `MFMediaID` object when your application is completely finished.


```
MFMediaID_Destroy
```

Example 2: Identification of Live Audio Input

1. Initialize the SDK – All applications must call this function once, and only once, before calling *any other* Audible Magic API.

```
MFGlobalInit
```

2. Using your platform's mechanism (e.g., audio control panel), select the desired audio input device (e.g., microphone or line-input).

3. Initialize an `MFMediaID` object with your credentials (i.e., license key).

```
MFMediaID_CreateUsingLicenseKey
```

4. Call MFMediaID_StartListening to tell the MFMediaID object to begin capturing audio in its circular buffer. This is typically done once when your application starts up.

5. Call `MFMediaID_IdentifyNow` to gather a fingerprint, do the identification, and return an `MFMediaIDResponse` object. This can be called as often as necessary.

Typically, the audio capture duration per each call to `MFMediaID_IdentifyNow` is between 6 to 25 seconds. The actual duration is defined in your configuration, as controlled by the license key. (See the discussion below of Pre Roll.)

6. Retrieve a copy of the XML response string from the `MFMediaIDResponse` object.

```
MFMediaIDResponse_GetStringLength
MFMediaIDResponse_GetAsString
```

As an alternative, or in addition to these functions, if one simply wants to know whether input media was identified, one may call

```
MFMediaIDResponse_GetIDStatus
```

which returns, by reference, an enum (integer) interpreted as follows:

```
typedef enum
  {
      MF_MEDIAID_RESPONSE_UNKNOWN,     /* server returned error */
      MF_MEDIAID_RESPONSE_NOT_FOUND,   /* match not found in DB */
      MF_MEDIAID_RESPONSE_FOUND,       /* match found in DB*/
  }
```

7. Destroy the `MFMediaIDResponse` object.

```
MFMediaIDResponse_Destroy
```

8. Parse the XML string, called an ID Response. (The returned XML string is in clear text, and parsing it is the responsibility of the client application. See the "Sample Response XML" section of this document.)

9. Steps - 5-8 can be called multiple times as needed.

10. Every OEM API call produces a return code. Be sure your application tests the return code after each function call, and if it is anything other than 0 (`MF_SUCCESS`), take the appropriate action. (See section "Error Codes" in the "Detailed API Reference" below.)

11. Destroy the `MFMediaID` object when your application is completely finished.

```
MFMediaID_Destroy
```

Pre-Roll

Customers often want to know how long after their application calls `MFMediaID_IdentifyNow` will the library capture audio. The answer is, it depends on the “Preroll” and “Duration” specified in your configuration (which is controlled by the license key).

If pre-roll is set to 0, then when `MFMediaID_IdentifyNow` is called the library will begin gathering audio until Duration seconds have been captured. At that point the identification will begin.

More typically, pre-roll is set to a non-zero value. In this scenario, instead of the gathering of audio starting when `MFMediaID_IdentifyNow` is called, the audio that is to be identified starts some number of seconds in the past, relative to the time that `MFMediaID_IdentifyNow` is called. If pre-roll is set to be equal to Duration, then the identification can begin as soon as `MFMediaID_IdentifyNow` is called. This makes for very fast identification response time.

Note that the gathering of audio for pre-roll cannot begin until `MFMediaID_StartListening` has been called.

Example 3: Automatic Periodic Lookups in a Local Database Using Live Audio Input

This example path is only suitable for local database implementations.

1. Initialize the SDK – All applications must call this function once, and only once, before calling *any other* Audible Magic API.

```
MFGlobalInit
```

2. Register your one and only Audible Magic SDK error callback function. This should be called after `MFGlobalInit` but before anything else. If you do not call this, you won't get error callbacks, meaning that errors returned by the SDK's internal threads will never be reported to your application.

```
MFRegisterErrorCallback
```

3. Using your platform's mechanism (e.g., audio control panel), select the desired audio input device (e.g., microphone or line-input).

4. Initialize an `MFMediaID` object with your credentials (i.e., license key).

```
MFMediaID_CreateUsingLicenseKey
```

5. Specify the callback function which will automatically be invoked when identifications occur.

```
MFMediaID_RegisterIDResponseCallback
```

6. Begin continuous audio capture, analysis and lookups.

```
MFMediaID_StartListening
```

7. The library will now attempt a lookup at a frequency which is set by your license key. Whenever a positive identification occurs, the callback function specified in `MFMediaID_RegisterIDResponseCallback` will be invoked. It will receive an `MFMediaIDResponse` object. Your callback function can retrieve a copy of the XML response string from the `MFMediaIDResponse` object.

```
MFMediaIDResponse_GetStringLength
MFMediaIDResponse_GetAsString
```

8. Inside the callback, destroy the `MFMediaIDResponse` object.

```
MFMediaIDResponse_Destroy
```

9. Inside the callback store the XML and exit the callback.

10. Your code notes that a new XML response has been obtained and it will parse the XML string, called an ID Response. (The returned XML string is in clear text, and parsing it is the responsibility of the client application. See the "Sample Response XML" section of this document.)

11. Every OEM API call produces a return code. Be sure your application tests the return code after each function call, and if it is anything other than 0 (`MF_SUCCESS`), take the appropriate action. (See section "Error Codes" in the "Detailed API Reference" below.)

12. When ready, stop the continuous audio capture, analysis, and lookup process.

    `MFMediaID_StopListening`

13. Destroy the `MFMediaID` object when your application is completely finished.

    `MFMediaID_Destroy`

Platform Implementation Notes

Windows

Starting with version 26.0, we ship the x64 build of our Windows release only. If you need to run on a 32 bit machine, or a 32 bit version of Windows, please contact support about a special build.

Linux

No specific notes.

Macintosh OS X

No specific notes.

Android

The audio capture from a microphone can be interrupted by other applications which take over the microphone. If your application needs to handle this case, you can modify the audio capture Java class function which captures audio from the microphone and passes it to the library.

Some Android devices do not have microphones.

iOS

When your application is put into the background, you should, upon retaking the foreground, destroy all your MFMediaID objects and create them again. Note that the notification happens asynchronously, thus you should be careful about any threading issues in your application code. The Audible Magic SDK is thread-safe, but if you destroy an object and then try to access it through an API call, we will return error 20187, `MF_MEDIAID_ALREADY_DESTROYED`.

Installation Guide and Programming Examples

This section explains how to install the SDK in order to run the programming examples and to develop your own application.

Note: Audible Magic License Keys

You will not be able to get any of the programming examples to run without error until you have obtained suitable license keys from Audible Magic. See the section "The License Key" earlier in this guide. The license key controls configuration parameters such as whether identification is done remotely (in which case ID requests are sent to a specified URL) or locally (in which case matches are sought in a specified database file), and whether identifications are done on an on-demand, one-shot basis (IdentifyNow mode) versus periodically at a specified frequency (continuous lookup mode). To obtain the license keys for the example applications, contact Audible Magic Support at support@audiblemagic.com.

Installation for Non-Mobile Platforms

All the necessary libraries (including the third-party, non-Audible Magic libraries) and executables are included in the `lib` folder of the release package.

On all non-mobile platforms:

Before running the programming examples, or when developing your own application using the Audible Magic OEM SDK, copy the *contents* of the `lib` folder (not the `lib` folder itself) into the folder containing the application's executable (whether that executable is one of the programming examples or your own program). You also need to set environment variables—`PATH` on Windows, `LD_LIBRARY_PATH` on Linux, and `DYLD_LIBRARY_PATH` on Mac OS X—to point to the current working directory ("."). Detailed per-platform instructions are given below.

The SDK assumes that the FFmpeg executable is in the current working directory. If you must instead run the application while cd'd to a directory other than the directory containing the application executable, you will need to have a copy of the FFmpeg executable, or a link to it, in the current working directory. If the FFmpeg executable (or a link to it) is not in the current working directory, and you are trying to identify a media file, you will likely see something like the following run-time error:

```
Error 48 (3760586800) Didn't find FFMPEG command-line program.
```

Caveat: If other versions of the FFmpeg executable might exist on the system, be sure that the search path for your application will find the Audible Magic-supplied version first. Do *not* substitute a different version of the FFmpeg executable! If you do, Audible Magic cannot guarantee proper operation of our library, and problems might well arise that do not cause any error to be reported.

Per-platform installation tips are given next.

Windows

If you have not installed the libraries correctly, you will probably get a run-time dialog saying something like this:

```
The application has failed to start because libcurl.dll was not found.
Reinstalling the application may fix this problem.
```

cURL DLLs

The Audible Magic library (mfcbr_oem.dll) communicates over the Internet with the Audible Magic ID servers by using the cURL libraries. Any application that uses this feature requires that the appropriate DLLs be placed in the folder containing the application's executable. These DLLS include the following:

libcurl.dll
libeay32.dll
ssleay32.dll

Be sure you have copied them from the OEM SDK release's `lib` folder into the folder containing your application's (or the programming example's) executable. Otherwise, an error dialog may appear at run time.

Microsoft Visual C/C++ Runtime DLLs

We redistribute the Microsoft runtime DLLs msvcr110.dll and msvcp110.dll, in case you do not have them on your system already.

If you are developing on Windows, and you are using Windows 2000 or XP (and have not installed XP Service Pack 2), and you are using a version of Microsoft's Visual Studio prior to VS 2010, you may be missing the particular version of the Microsoft C/C++ runtime DLLs that are required for the Audible Magic OEM SDK.

If you see the following error

```
(48) File not found: mfcbr_oem.dll [<program name>]
```

when running any of the prebuilt example applications that ship with the Audible Magic OEM SDK, or when running your own application built with our SDK, and you've verified that mfcbr_oem.dll is in fact in the application's executable folder, then your system is missing the appropriate C/C++ runtime libraries.

The surest way to resolve this problem is to upgrade to Visual Studio 2012 or later. If you'd rather not upgrade your Visual Studio, then you must copy the redistributable DLLs, mentioned above, into the same folder that contains the executable for your Audible Magic-based application.

Windows Media

The SDK assumes that Windows Media Player is installed on Windows platforms. If it is not, the library (and therefore the programming examples) might not function as expected when presented with WMA or WMV input files.

Note that Windows Media Player is *not* installed by default on some systems, such as Windows Server 2008. On Windows Server 2008, you must enable the "Desktop Experience" by the selecting the following Start menu sequence:

Control Panel -> Programs and Features -> Add Feature(s) -> Desktop Experience

This change requires a reboot.

Linux and Mac OS X

As on Windows, the shared libraries and the FFmpeg executable are found in the `lib` directory of the release package and should be copied into the directory containing the application's executable.

You must also set the environment variable `LD_LIBRARY_PATH` (on Linux) or `DYLD_LIBRARY_PATH` (on the Mac) to include the directory containing the shared libraries. The easiest way is to set the variable to the current working directory, "."; this works if you have copied the contents of the `lib` folder to the application directory and are cd'd into that directory. Otherwise, the environment variable will need to include the path to the `lib` folder (or to the application directory, if you have copied or linked everything into it), either as an absolute path or as a path relative to the working directory.

In the csh (or tcsh) shell, this is done with the `setenv` command. For example (on Linux):

```
setenv LD_LIBRARY_PATH .:../../lib:$LD_LIBRARY_PATH
```

In the bash or Bourne shells, use the `export` command. For example (on Linux):

```
export LD_LIBRARY_PATH=.:../../lib:$LD_LIBRARY_PATH
```

On the Mac, do exactly the same thing, except that the environment variable is called `DYLD_LIBRARY_PATH`.

In this particular example, the library search path is set to the current directory ("."), followed by the `lib` folder (specified as a relative path that assumes that your working directory is `examples/c_c++`), followed by the previously set value of the environment variable. In many cases, however, simply setting the library search path to "." will work.

If you have not installed the libraries and set the library search path correctly, you will probably get a run-time error message, such as the following on Linux:

```
error while loading shared libraries: libavcore.so.0: cannot open
shared object file: No such file or directory
```

and on the Mac:

```
dyld: Library not loaded: @executable_path/libmfcbroem.dylib
```

C Programming Examples for Non-Mobile Platforms

The OEM SDK release packages for non-mobile platforms include sample programs for using the Audible Magic API with a C/C++ interface. See the folder `examples/c_c++`. (A future release might include Java example programs.)

The `examples/c_c++` folder contains pre-built executables as well as source files. You should be able to run the binaries "out of the box," as long as you follow the steps described under "Installation for Non-Mobile Platforms" above, to make sure your OS loader knows where to find the needed libraries. As mentioned above, you also need to obtain license key(s) from Audible Magic, and specify the appropriate license key on the command line, else the program will exit with an error message.

Depending on the platform for which the release is packaged, the `examples/c_c++` folder also includes a Makefile (for Linux and OS X) or the appropriate Microsoft project files (for Windows) so that you can build these programs yourself. On Linux or OS X, `cd` into the `examples/c_c++` directory and type `make`.

On Windows, the pre-built executables for the example files listed below are in the `examples/c_c++/Release_x64` folder. If you want to re-build these programs just double-click the `examples-VS2010.sln` file to launch Visual Studio, and select Build Solution from the Build menu.

Programming Example Files:

identifyFile.c

This command-line program will take an input media file, a user-supplied asset identifier, and an output filename, and, in one step, fingerprint the input file and contact an Audible Magic ID server (or look in a local database) to try to identify it. The resulting XML response is stored in the output file.

continuousLookup.c

This command-line program does continuous fingerprinting of the platform's default audio input (typically a microphone), doing lookups against a local database and/or a remote ID server, and displaying the hits (positive identifications) on the console. The local database (if any) is specified by the user and must have been created using the amDbGen program (supplied separately from the SDK).

The license key must allow "continuous lookup" mode. The lookup frequency is specified by a configuration parameter controlled by the license key.

identifyNow.c

This command-line program is similar to continuousLookup.c, except that it makes on-demand requests to identify the default audio input, instead of continuous requests. In its interactive mode, which is the default, it does an ID request every time the user types "i" followed by <RETURN>. In its noninteractive mode, it does a single ID request and exits.

The license key must allow "IdentifyNow" mode.

listenToSamples.c

This is an advanced programming example provided for developers who need to supply their own real-time audio feed. See the reference documentation for the MFListenToSamples() function. The license key must allow this feature.

generateRequest.c

This is the program allows you to separate the *creation* of an ID Request from the *submission* of that request to our servers. You will need to have a local config file or license key in order to create an ID request output file (xml format) using this program.

postRequest.c

This is the program allows you to *submit* an ID request that was previously created with the **generateRequest** program. You will need to have a local config file or license key in order to submit your ID request file. The output file from this program is an xml document containing the response from our servers about the unknown audio.

Mobile Platforms

As with the non-mobile platforms, suitable license keys must be obtained from Audible Magic in order for the programming examples to work correctly.

Android

Android 2.3 or later is required. The Audible Magic SDK release package for Android includes two programming examples: “AMContinuousSampleApp” and “AMIdentifyNowSampleApp.” These applications illustrate how to do remote and local lookups using microphone input and either continuous monitoring or one-shot (IdentifyNow) identification. For instructions on how to configure and run the examples, see the README file in the release package. The instructions assume you are using the Eclipse integrated development environment.

iOS

The release package includes three programming examples, called MediaID_ios, SmartSyncSample, and SmartSyncEventsSample. The last of these is an advanced example whose directory includes a README file explaining its use. Each of the programming examples includes an Xcode project that will build the sample application.

All iOS platforms capable of running iOS 4 or higher, through iOS 7, are supported. We assume the use of Apple's Xcode, version 4.0 or later, for iOS development. We developed and tested the iOS programming examples using Xcode version 4.5.2 with iOS SDK 7.

Audible Magic-based iOS applications require your project to include all of the following Apple frameworks:

AudioToolbox
AVFoundation
CoreAudio
Foundation
MessageUI
Security
SystemConfiguration
UIKit

as well as:

libz.1.2.5.dylib (or libz.1.2.3.dylib if using Xcode version < 4.2)

which, depending on the iOS version, may found in a system path such as `/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS[x.y].sdk/usr/lib`

or `/Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS[x.y].sdk/usr/lib.`

Your project must also include all of the static libraries found in the Audible Magic release package's `lib` directory (i.e., libcurl.a and libmfcbroem_ios.a).

In order to build and execute the application on your iPhone/iPad/iPod-Touch device, you must be an Apple Developer and have created the appropriate provisioning files and certificates.

Detailed API Reference

Note: C programmers need to be aware of several conventions the API follows. (These conventions were originally designed for compatibility with Visual Basic.)

1. Space for returned strings is always allocated and released by the *caller*.

2. Space for structures is allocated and released within the library. Only a pointer to the structure is returned. The *caller* must make an explicit call to the library to destroy these structures. This is detailed in the reference documentation.

3. On Windows, all functions are declared with the `__stdcall` calling convention (double underscore).

4.

Error Codes

Nearly all the API calls in the SDK return an `MFError` error value (or, in the case of the Java interface, an `int` that should be interpreted as an `MFError`.) The error value returned from an API call contains both an error code and additional information that tells Audible Magic where in the code the error occurred. To translate an API error return value into one of the codes below your application will need to call `MFError_GetCode`. The value returned from this API call will map into the list below. A human-readable error string can always be returned from the call `MFError_GetDescription`, whether or not the argument has already been translated via `MFError_GetCode`.

```
/*
 * This list is not complete.  Some errors are generated automatically, for example the
 * errors returned from libcurl, which are merely added to MF_CURL_ERROR_BASE, HTTP
 * status codes reported by libcurl, which are added to MF_HTTP_STATUS_CODE_BASE, etc.
 */

#define MF_SUCCESS                                          (0)
#define MF_FAILURE                                          (1)
#define MF_OUT_OF_MEMORY                                    (2)
#define MF_INCORRECT_COMMAND_LINE                           (3)
#define MF_CALLING_PROGRAM_ERROR                            (4)
#define MF_INTERNAL_PROGRAM_ERROR                           (5)
#define MF_FILE_IO_ERROR                                    (6)
#define MF_MALFORMED_INPUT_DATA                             (7)
#define MF_NOT_IMPLEMENTED_YET                              (8)
#define MF_UNSUPPORTED_SOUNDFILE_FORMAT                     (9)
#define MF_NO_SOUND_DATA_IN_FILE                            (10)
#define MF_HARDWARE_ERROR                                   (11)
#define MF_END_OF_FILE                                      (12)
#define MF_DROPPED_SAMPLES_ERROR                            (13)
#define MF_NULL_POINTER                                     (14)
#define MF_PARAMETER_OUT_OF_RANGE                           (15)
#define MF_COULDNT_CREATE_FILE                              (16)
#define MF_BAD_HEADER                                       (17)
#define MF_COULDNT_OPEN_URL                                 (18)
#define MF_COULDNT_LOAD_REALAUDIO_DLL                       (19)
#define MF_REALAUDIO_NO_MORE_PLAYERS                        (20)
#define MF_REALAUDIO_SDK_ERROR                              (21)
#define MF_WINDOWS_REGISTRY_ERROR                           (22)
#define MF_WRONG_CODEC_VERSION                              (23)
#define MF_CODEC_OPEN_ERROR                                 (24)
#define MF_CODEC_READ_ERROR                                 (25)
#define MF_CODEC_NO_MORE_PLAYERS                            (26)
#define MF_STRING_TOO_SHORT                                 (27)
#define MF_THREAD_ERROR                                     (28)
#define MF_MUTEX_ERROR                                      (29)
#define MF_TIMEOUT                                          (30)
#define MF_KEY_ALREADY_EXISTS                               (31)
#define MF_NO_SUCH_KEY                                      (32)
#define MF_COULDNT_EXPAND_ARCHIVE                           (33)
#define MF_COULDNT_DELETE_FILE                              (34)
#define MF_COULDNT_CREATE_DIRECTORY                         (35)
#define MF_COULDNT_DELETE_DIRECTORY                         (36)
#define MF_FORMAT_UNSUPPORTED_BY_SOUND_DEVICE               (37)
#define MF_DUPLICATE_AUDIO_BUFFER_ERROR                     (38)
#define MF_STRING_TOO_LONG                                  (39)
#define MF_FILE_ALREADY_EXISTS                              (40)
#define MF_DLL_HACKED                                       (41)
#define MF_NO_VIDEO_AVAILABLE                               (42)
#define MF_DISALLOWED_FORMAT                                (43)
#define MF_OPERATION_DISABLED                               (44)
#define MF_FILE_TOO_BIG                                     (45)
#define MF_NO_STREAMS_FOUND                                 (46)
#define MF_FILE_IS_PROTECTED_BY_DRM                         (47)
```

```
#define MF_COULDNT_OPEN_FFMPEG_EXECUTABLE                          (48)
#define MF_FILE_NOT_FOUND                                          (49)
#define MF_FFMPEG_OUTPUT_FILE_NOT_FOUND                            (50)
#define MF_NETWORK_ERROR                                           (51)
#define MF_AUDIO_CODEC_READ_ERROR                                  (52)
#define MF_VIDEO_CODEC_READ_ERROR                                  (53)
#define MF_NEEDS_ENTIRE_FILE                                       (54)
#define MF_NO_COMMAND_SERVER                                       (55)
#define MF_COULDNT_OPEN_SOCKET                                     (56)
#define MF_WRONG_CODEC                                             (57)
#define MF_RHOZET_MUST_USE_AUDIO_OR_VIDEO                          (58)
#define MF_SQL_NOT_AVAILABLE                                       (59)
#define MF_MS_RUNTIME_LIB_FOR_VSS1_NOT_FOUND                       (60)
#define MF_MS_RUNTIME_LIB_WRONG_VERSION_LINKED                     (61)
#define MF_MS_RUNTIME_LIB_MISSING_S_BY_S_FOLDER                    (62)
#define MF_MS_RUNTIME_LIB_WRONG_S_BY_S_FOLDER                      (63)
#define MF_SQL_UNABLE_TO_OPEN                                      (64)
#define MF_SQL_COMMAND_FAILED                                      (65)
#define MF_SQL_PREPARE_FAILED                                      (66)
#define MF_SQL_ESCAPE_STRING_FAILED                                (67)
#define MF_DIRECTORY_NOT_FOUND                                     (68)
#define MF_SPECIFIED_METADATA_FILE_NOT_FOUND                       (69)
#define MF_INVALID_CLIENT_SUFFIX                                   (70)
#define MF_NOT_A_MEDIA_FILE                                        (71)
#define MF_UNEXPECTED_SERVER_RESPONSE                              (72)
#define MF_CURL_INIT_ERROR                                         (73)
#define MF_CODEC_CLOSE_ERROR                                       (74)
#define MF_MORE_THAN_ONE_FILE_FOUND                                (75)
#define MF_INVALID_CHARACTER                                       (76)
#define MF_MISSING_MF_LIBRARY                                      (77)
#define MF_COULDNT_GET_IF_ADDRESSES                                (78)
#define MF_IP_CONVERSION_FAILED                                    (79)
#define MF_REFCNT_DEBUG_AUDIO_LOST_SYNC                            (80)
#define MF_ERROR_END                                               (81)
#define MF_ID_SERVER_ERROR_BASE                                    (0)
#define MF_ID_SERVER_ERROR_BEGIN                                   (2000)
#define MF_ID_SERVER_ERROR_END                                     (2999)
#define MF_ID_RESPONSE_STATUS_REQUEST_DENIED                       (2001)
#define MF_ID_RESPONSE_STATUS_CLIENT_OBSOLETE                      (2002)
#define MF_ID_RESPONSE_STATUS_DO_HANDSHAKE                         (2003)
#define MF_ID_RESPONSE_STATUS_ERROR                                (2004)
#define MF_ID_RESPONSE_STATUS_SONG_NOT_FOUND                       (2005)
#define MF_ID_RESPONSE_STATUS_MATCH                                (2006)
#define MF_ID_RESPONSE_STATUS_BAD_REQ_TYPE                         (2007)
#define MF_ID_RESPONSE_STATUS_BAD_TRAN_TYPE                        (2008)
#define MF_ID_RESPONSE_STATUS_NO_MEDIA_SIG                         (2009)
#define MF_ID_RESPONSE_STATUS_SERVER_BUSY                          (2010)
#define MF_ID_RESPONSE_STATUS_UPGRADE_AVAIL                        (2011)
#define MF_ID_RESPONSE_STATUS_OVER_CONCURRENCY_LIMIT               (2012)
#define MF_ID_RESPONSE_STATUS_REJECTED                             (2013)
#define MF_ID_RESPONSE_STATUS_OVER_PER_DAY_LIMIT                   (2014)
#define MF_ID_RESPONSE_STATUS_THREAD_ABORT                         (2027)
#define MF_ID_RESPONSE_STATUS_THREAD_ERROR                         (2028)
#define MF_ID_RESPONSE_STATUS_CANT_CONNECT_TO_ID_ENGINE            (2029)
#define MF_ID_RESPONSE_STATUS_TIMEOUT_CONNECTING_TO_ID_ENGINE      (2030)
#define MF_ID_RESPONSE_STATUS_ERROR_IN_ID_ENGINE                   (2031)
#define MF_ID_RESPONSE_STATUS_ERROR_IN_LIVEBROADCAST_MANAGER       (2032)
#define MF_ID_RESPONSE_STATUS_NOT_USING_HTTP_POST                  (2033)
#define MF_ID_RESPONSE_STATUS_ERROR_IN_PROXY_CODE                  (2034)
#define MF_ID_RESPONSE_STATUS_ID_REQUEST_EMPTY                     (2035)
#define MF_ID_RESPONSE_STATUS_ID_REQUEST_TOO_LARGE                 (2036)
#define MF_ID_RESPONSE_STATUS_ID_REQUEST_MISSING_VALUES            (2037)
#define MF_ID_RESPONSE_STATUS_ID_REQUEST_INVALID                   (2038)
#define MF_ID_RESPONSE_STATUS_CANT_DECRYPT                         (2039)
#define MF_ID_RESPONSE_STATUS_APP_NAME_NOT_CONFIGURED              (2040)
#define MF_ID_RESPONSE_STATUS_ID_SVR_TIERS_NOT_CONFIGURED          (2041)
#define MF_ID_RESPONSE_STATUS_INVALID_APP_OWNER                    (2042)
#define MF_ID_RESPONSE_STATUS_CONCURRENCY_ERROR                    (2043)
#define MF_ID_RESPONSE_STATUS_MESSAGE_QUEUE_CONNECTION_ERROR       (2044)
#define MF_ID_RESPONSE_STATUS_NO_RESPONSE_FROM_MESSAGE_QUEUE       (2045)
#define MF_CURL_ERROR_BASE                                         (3000)
#define MF_CURL_ERROR_END                                          (3999)
```

```
#define MF_RSA_ERROR_BASE                                (4000)
#define MF_RSA_SIGN_ERROR_SHA                            (MF_RSA_ERROR_BASE + 0)
#define MF_RSA_SIGN_ERROR_SIG_BUFFER_OVERFLOW            (MF_RSA_ERROR_BASE + 1)
#define MF_RSA_VERIFY_ERROR_BN_CTX                       (MF_RSA_ERROR_BASE + 2)
#define MF_RSA_VERIFY_ERROR_SHA                          (MF_RSA_ERROR_BASE + 3)
#define MF_RSA_VERIFY_ERROR_BLINDING                     (MF_RSA_ERROR_BASE + 4)
#define MF_RSA_READPRIVATEKEY_ERROR                      (MF_RSA_ERROR_BASE + 5)
#define MF_RSA_READPUBLICKEY_ERROR                       (MF_RSA_ERROR_BASE + 6)
#define MF_RSA_SIGN_ERROR_SIGN_FAILED                    (MF_RSA_ERROR_BASE + 7)
#define MF_RSA_VERIFY_FAILED                             (MF_RSA_ERROR_BASE + 8)
#define MF_RSA_VERIFY_ERROR_B64_PADDING                  (MF_RSA_ERROR_BASE + 9)
#define MF_RSA_VERIFY_ERROR_B64_STRICT                   (MF_RSA_ERROR_BASE + 10)
#define MF_RSA_VERIFY_ERROR_B64_MALLOC                   (MF_RSA_ERROR_BASE + 11)
#define MF_RSA_PUBKEY_SET_FROM_HEX_ERROR                 (MF_RSA_ERROR_BASE + 12)
#define MF_RSA_PUBKEY_GET_HEX_ERROR                      (MF_RSA_ERROR_BASE + 13)
#define MF_RSA_HACK_SUSPECTED                            (MF_RSA_ERROR_BASE + 14)
#define MF_RSA_ERROR_END                                 (MF_RSA_ERROR_BASE + 14)
#define MF_HTTP_STATUS_CODE_BASE                         (5000)
#define MF_HTTP_STATUS_CODE_END                          (5999)
#define MF_TRID_ERROR_BASE                               (6000)
#define MF_TRID_ERROR_END                                (6999)
#define MF_PA_ERROR_BASE                                 (10000)
#define MF_PA_ERROR_ADDEND                               (20000)
#define MF_NOT_A_SIGNATURE                               (20000)
#define MF_UNSUPPORTED_SIG_VERSION                       (20001)
#define MF_INVALID_SIGNATURE                             (20002)
#define MF_INVALID_OPTIONS                               (20003)
#define MF_HOP_SIZE_DOESNT_MATCH                         (20004)
#define MF_SEGMENT_SIZE_DOESNT_MATCH                     (20005)
#define MF_MINFREQ_DOESNT_MATCH                          (20006)
#define MF_MAXFREQ_DOESNT_MATCH                          (20007)
#define MF_SAMPLE_RATE_TOO_LOW                           (20008)
#define MF_SIGNATURE_TOO_SHORT                           (20009)
#define MF_DATABASE_EMPTY                                (20010)
#define MF_INVALID_OBJECT                                (20011)
#define MF_COULDNT_OPEN_DATABASE                         (20012)
#define MF_CRC_ERROR                                     (20013)
#define MF_BUFFER_TOO_SMALL                              (20014)
#define MF_UNSUPPORTED_DATABASE_VERSION                  (20015)
#define MF_BAD_AUDIO_CAPTURE_STATE                       (20016)
#define MF_OUT_OF_DISK_SPACE                             (20017)
#define MF_AUDIO_CAPTURE_ERROR                           (20018)
#define MF_AUDIO_HARDWARE_ERROR                          (20019)
#define MF_SIGNATURE_TOO_BIG                             (20020)
#define MF_UNKNOWN_XML_CUSTOMER                          (20021)
#define MF_MUTEX_FAILURE                                 (20022)
#define MF_CANT_WRITE_NORMALIZED_SIG                     (20023)
#define MF_SIGNATURE_SILENT                              (20024)
#define MF_MALFORMED_XML                                 (20025)
#define MF_INVALID_CHECKSUM                              (20026)
#define MF_INVALID_VERSION                               (20027)
#define MF_INVALID_USERGUID                              (20028)
#define MF_THREAD_FAILURE                                (20029)
#define MF_ATA_FAILURE                                   (20030)
#define MF_NUM_MFCCS_DOESNT_MATCH                        (20031)
#define MF_NO_OVERLAP                                    (20032)
#define MF_BAD_START_TIME                                (20033)
#define MF_ALL_OVERLAP                                   (20034)
#define MF_INPUT_AUDIO_FORMATS_DIFFER                    (20035)
#define MF_SAMPLE_RATE_CHANGED                           (20036)
#define MF_LOOKUP_STOPPED                                (20037)
#define MF_ANALYSIS_STOPPED                              (20038)
#define MF_SIGNATURE_DOESNT_MATCH                        (20039)
#define MF_DATABASE_NOT_FOUND                            (20040)
#define MF_THREAD_WAIT_FAILURE                           (20041)
#define MF_BAD_CONFIGURATION                             (20042)
#define MF_BUFFER_EMPTY                                  (20043)
#define MF_BAD_RESPONSE_FROM_SERVER                      (20044)
#define MF_BUFFER_TOO_BIG                                (20045)
#define MF_MP3_DECODER_FAILURE                           (20046)
#define MF_NOT_IN_DATABASE                               (20047)
#define MF_NOT_ON_SERVER                                 (20048)
```

```
#define MF_NOT_IN_METADATA                                      (20049)
#define MF_INSUFFICIENT_DATA                                    (20050)
#define MF_TOO_MANY_THREADS                                     (20051)
#define MF_TEXTFILTER_NOT_TRAINABLE                             (20052)
#define MF_GUIDS_DONT_MATCH                                     (20053)
#define MF_ID_SERVER_UP_BUT_NOT_RESPONDING                      (20054)
#define MF_ID_SERVER_DOWN                                       (20055)
#define MF_ID_SERVER_HEALTH_UNKNOWN                             (20056)
#define MF_CANT_REACH_HEALTH_CHECK_SERVER                       (20057)
#define MF_HEALTH_CHECK_BAD_FORMAT                              (20058)
#define MF_HEALTH_CHECK_BAD_GMT                                 (20059)
#define MF_ID_SERVER_DOS_ATTACK                                 (20060)
#define MF_INVALID_AMITEMID                                     (20061)
#define MF_TOO_MANY_SIGNATURES_TO_INDEX                         (20062)
#define MF_TOO_MANY_SEGMENTS_TO_INDEX                           (20063)
#define MF_SIGNATURE_NOT_RICH_ENOUGH                            (20064)
#define MF_VIDEO_SIGNATURE_NON_MONOTONIC                        (20065)
#define MF_VIDEO_SIGNATURE_MAX_SCENE_LOCATION_TOO_BIG           (20066)
#define MF_VIDEO_SIGNATURE_FEATURE_SET_LIBRARY_NOT_FOUND        (20067)
#define MF_VIDEO_SIGNATURE_FEATURE_SET_FUNCTION_NOT_FOUND       (20068)
#define MF_VIDEO_SIGNATURE_STRIPPED                             (20069)
#define MF_INCONSISTENT_SIGNATURE_NORMALIZATION                 (20070)
#define MF_NORMALIZED_SEARCH_NOT_SPECIFIED                      (20071)
#define MF_UNSUPPORTED_MEDIA_FILE                               (20072)
#define MF_CONFIGURATION_ERROR                                  (20073)
#define MF_VIDEO_SIGNATURE_FEATURE_SET_UNKNOWN                  (20074)
#define MF_UNDESIRED_FILE_FORMAT                                (20075)
#define MF_VIDEO_SIGNATURE_BAD_FRAME_TIMES                      (20076)
#define MF_VIDEO_SIGNATURE_FEATURE_SET_SIG_GEN_FAILED           (20077)
#define MF_COULDNT_WRITE_TO_DATABASE                            (20078)
#define MF_TOO_MANY_ATTEMPTS                                    (20079)
#define MF_NOT_IN_CACHE                                         (20080)
#define MF_INVALID_CUSTOMER_PREFIX                              (20081)
#define MF_UNSUPPORTED_VARIABLE_OR_FLEXIBLE_FRAME_RATE          (20082)
#define MF_AUDIO_SIGNATURE_ALL_SILENCE                          (20083)
#define MF_LOOKUP_RUNNING                                       (20084)
#define MF_LOOKUP_RESULTS_UNAVAILABLE                           (20085)
#define MF_CHUNKLIST_OUT_OF_SYNC                                (20086)
#define MF_INVALID_DATA                                         (20087)
#define MF_PARAMETER_NOT_INTEGRAL_MULTIPLE_OF_HOP_SIZE          (20088)
#define MF_CONFIG_FILE_NOT_FOUND                                (20089)
#define MF_DENIED_BY_AUTHORIZATION_SERVER                       (20090)
#define MF_MESSAGE_TOO_LONG                                     (20091)
#define MF_CANT_START_SERVER                                    (20092)
#define MF_MEDIA_ID_ALREADY_LISTENING                           (20093)
#define MF_MEDIA_ID_NOT_LISTENING                               (20094)
#define MF_NO_DATABASE_SPECIFIED                                (20095)
#define MF_TWO_DATABASES_SPECIFIED                              (20096)
#define MF_NO_SEGMENTS_TO_FINGERPRINT                           (20097)
#define MF_BAD_REQUEST                                          (20098)
#define MF_SERVER_ERROR                                         (20099)
#define MF_SERVER_BAD_REQ_TYPE                                  (20102)
#define MF_SERVER_NO_MEDIA_SIG                                  (20103)
#define MF_SERVER_SERVER_BUSY                                   (20104)
#define MF_SERVER_OVER_CONCURRENCY_LIMIT                        (20105)
#define MF_SERVER_REQUEST_REJECTED                              (20106)
#define MF_INVALID_CHARACTER_IN_AMITEMID                        (20107)
#define MF_INCOMPATIBLE_CONFIGURATION_FILE_VERSION              (20108)
#define MF_DATABASE_TOO_LARGE                                   (20109)
#define MF_AMITEMID_TOO_LONG                                    (20110)
#define MF_MALFORMED_CONFIG_FILE                                (20111)
#define MF_INVALID_BOOLEAN_VALUE_IN_CONFIG_FILE                 (20112)
#define MF_INVALID_INTEGER_VALUE_IN_CONFIG_FILE                 (20113)
#define MF_INVALID_DOUBLE_VALUE_IN_CONFIG_FILE                  (20114)
#define MF_INVALID_STRING_VALUE_IN_CONFIG_FILE                  (20115)
#define MF_UNINITIALIZED_CONFIG_FILE_PARAMETER                  (20116)
#define MF_CONFIG_FILE_SIGNING_ERROR                            (20117)
#define MF_CONFIG_FILE_DUPLICATE_TAG                            (20118)
#define MF_XML_UNKNOWN_TAG                                      (20119)
#define MF_FEATURES_ON_DISK_UNSUPPORTED                         (20120)
#define MF_SIGNATURE_TYPE_UNSUPPORTED                           (20121)
#define MF_SEARCH_TYPE_UNSUPPORTED                              (20122)
```

```
#define MF_SIGNATURE_NOT_FOUND                                    (20123)
#define MF_NOT_AN_XML_REQUEST                                     (20124)
#define MF_LOCAL_LOOKUPS_DISALLOWED                               (20125)
#define MF_REMOTE_LOOKUPS_DISALLOWED                              (20126)
#define MF_CONFLICTING_LOOKUP_TYPES                               (20127)
#define MF_UNSUPPORTED_DATABASE_TYPE                              (20128)
#define MF_DATABASE_INDEX_OUT_OF_RANGE                            (20129)
#define MF_OPERATION_UNSUPPORTED_BY_CONFIG_FILE__NO_SERVER_URL    (20130)
#define MF_IDENTIFY_NOW_DISALLOWED_DURING_CONTINUOUS_LOOKUP       (20131)
#define MF_MEDIA_ID_NOT_DEBUGGING_AUDIO                           (20132)
#define MF_SPLICING_ENDED_TOO_EARLY                               (20133)
#define MF_SPLICING_NO_MATCH                                      (20134)
#define MF_HTTP_DISABLED                                          (20135)
#define MF_NO_DATABASES                                           (20136)
#define MF_FINGERPRINT_SERVER_SOCKET_CONNECT_ERROR                (20137)
#define MF_FINGERPRINT_SERVER_SOCKET_READ_ERROR                   (20138)
#define MF_FINGERPRINT_SERVER_SOCKET_WRITE_ERROR                  (20139)
#define MF_NO_FEATURES                                            (20140)
#define MF_BAD_SEGMENT_OR_HOP_SIZE                                (20141)
#define MF_COULDNT_GET_THREAD_PRIORITY                            (20142)
#define MF_COULDNT_SET_THREAD_PRIORITY                            (20143)
#define MF_AUTHORIZATION_SERVER_UNREACHABLE                       (20144)
#define MF_BAD_RESPONSE_FROM_AUTHORIZATION_SERVER                 (20145)
#define MF_EMPTY_RESPONSE_FROM_CONFIGURATION_SERVER               (20146)
#define MF_NO_IP_ADDRESS_FOUND                                    (20147)
#define MF_CUSTOM_ERROR_MESSAGE                                   (20148)
#define MF_COULDNT_OPEN_CONFIGURATION_NAME_FILE                   (20149)
#define MF_AUTHORIZATION_SERVER_UNTRIED                           (20150)
#define MF_LISTEN_TO_SAMPLES_API_DISALLOWED                       (20151)
#define MF_AUTHORIZATION_ATTEMPT_FAILED                           (20152)
#define MF_NO_PARTITIONS                                          (20153)
#define MF_NO_SUCH_PARTITION                                      (20154)
#define MF_SAMPLE_BUFFER_FAILURE                                  (20155)
#define MF_SHUTTING_DOWN                                          (20156)
#define MF_COULDNT_INITIALIZE_CURL                                (20157)
#define MF_GLOBAL_INIT_HASNT_FINISHED                             (20158)
#define MF_GLOBAL_INIT_HAS_ALREADY_BEEN_CALLED                    (20159)
#define MF_AUTHORIZATION_PENDING                                  (20160)
#define MF_THRIFT_ERROR                                           (20161)
#define MF_WEB_SITE_DISALLOWED                                    (20162)
#define MF_IDENTIFY_NOW_LOOKUPS_NOT_ALLOWED                       (20163)
#define MF_NO_ID_RESPONSE_CALLBACK_REGISTERED                     (20164)
#define MF_EMPTY_RESPONSE_FROM_DATABASE_SERVER                    (20165)
#define MF_MEDIA_ID_RESPONSE_CALLBACK_ALREADY_REGISTERED          (20166)
#define MF_MEDIA_ID_NOT_INITIALIZED                               (20167)
#define MF_UNSUPPORTED_AUDIO_FORMAT                               (20168)
#define MF_LISTENING_SOURCE_MISMATCH                              (20170)
#define MF_LISTENING_SOURCE_NONE                                  (20171)
#define MF_INVALID_ID_REQUEST_VERSION                             (20172)
#define MF_NO_AUDIO_INPUT_DEVICE                                  (20173)
#define MF_PREROLL_DURATION_GREATER_THAN_DURATION                 (20174)
#define MF_CONFIG_FILE_DEPRECATED_TAG                             (20175)
#define MF_LICENSE_STRING_TOO_LONG                                (20176)
#define MF_ILLEGAL_CHARACTERS_IN_LICENSE_STRING                   (20177)
#define MF_LICENSE_STRING_TOO_SHORT                               (20178)
#define MF_BAD_DATABASE                                           (20179)
#define MF_EVENT_COULDNT_BE_INITIALIZED                           (20180)
#define MF_THREAD_CREATE_FAILED                                   (20181)
#define MF_THREAD_DETACH_FAILED                                   (20182)
#define MF_BAD_CONFIGURATION_NO_LOOKUP_SPECIFIED                  (20183)
#define MF_BAD_CONFIGURATION_REMOTE_AND_LOCAL_SPECIFIED           (20184)
#define MF_NEGATIVE_DURATION_INTERNAL_ERROR                       (20185)
#define MF_FINGERPRINTER_TIMEDOUT                                 (20186)
#define MF_MEDIAID_ALREADY_DESTROYED                              (20187)
#define MF_MEDIAID_NO_SUCH_OBJECT                                 (20188)
#define MF_MEDIAID_CREATE_INTERNAL_ERROR                          (20189)
#define MF_NULL_MEDIAID_POINTER                                   (20190)
#define MF_INVALID_OFFSET                                         (20191)
#define MF_INVALID_DURATION                                       (20192)
#define MF_AUTHORIZATION_INFO_NOT_CREATED                         (20193)
#define MF_CALLING_SAFETHREAD_DESTROY_FROM_EXTERNAL_THREAD        (20194)
#define MF_CALLING_SAFETHREAD_DESTROY_WITH_PENDING_MESSAGES       (20195)
```

```
#define MF_SIGNATURE_NOT_NORMALIZED                              (20196)
#define MF_NO_DATABASE_LOADED                                    (20197)
#define MF_CANT_NORMALIZE_RAW_SIGNATURE                          (20198)
#define MF_SIGNATURE_DOESNT_CONTAIN_DIRECTIONS                (20199)
#define MF_COULDNT_STAT_FILE                                  (20200)
#define MF_CONFIG_FILE_UNSUPPORTED_IDREQUEST_VERSION          (20201)
#define MF_VIDEO_SIGNATURE_PARAMETERS_DO_NOT_MATCH            (20202)
#define MF_UNKNOWN_TRANSACTION_TYPE                           (20203)
#define MF_UNKNOWN_MESSAGE_FIELD_NAME                         (20204)
#define MF_CANNOT_PROCESS_MEDIA_FILES                         (20205)
#define MF_COULDNT_OPEN_FILE                                  (20206)
#define MF_UNKNOWN_MFCAF_OPCODE                               (20207)
#define MF_UNEXPECTED_MFCAF_OPCODE                            (20208)
#define MF_AUDIO_FINGERPRINTER_UNINITIALIZED                  (20209)
#define MF_AUDIO_FINGERPRINTER_THREAD_ISNT_RUNNING            (20210)
#define MF_FAILED_TO_ALLOCATE_DEBUG_AUDIO_BUFFER              (20211)
#define MF_FAILED_TO_RETRIEVE_DEBUG_AUDIO_BUFFER              (20212)
#define MF_FAILED_TO_RETRIEVE_DEBUG_SIGNATURE_BUFFER          (20213)
#define MF_CONFIG_FILE_TAG_OUTSIDE_OF_SIGNED_SECTION          (20214)
#define MF_BAD_XML_COMMENT                                    (20215)
#define MF_UNNORMALIZED_SEARCH_WITHOUT_UNNORMALIZED_FEATURES (20216)
#define MF_UNNORMALIZED_REMOVAL_STATE_DOESNT_MATCH            (20217)
#define MF_BAD_NUM_FEATURES_TO_COMPARE                        (20218)
#define MF_METADATA_SERVICE_EXCEPTION                         (20219)
#define MF_THRIFT_EXCEPTION                                   (20220)
#define MF_BAD_FEATURE_TYPE                                   (20221)
#define MF_COULDNT_ADD_CUSTOM_HTTP_HEADER                     (20222)
#define MF_INVALID_LOOKUP_PARAM                               (20223)
#define MF_EMPTY_SIGNATURE                                    (20224)
#define MF_LISTENING_SOURCE_UNKNOWN                           (20225)
#define MF_NO_SIGNATURE_FILE_OR_PATH                             (20226)
#define MF_COULDNT_CREATE_XML_NODE                               (20227)
#define MF_AMITEMIDS_DIFFER                                      (20228)
#define MF_NOT_A_SIGNATURE_NOR_A_DATABASE_FILE                   (20229)
#define MF_FSEEK_FAILED                                          (20230)
#define MF_REFERENCE_PATH_NOT_ACCESSIBLE                         (20231)
#define MF_INVALID_WHILE_CONTINUOUS_LOOKUPS_RUNNING              (20232)
#define MF_PARAMETER_CANT_BE_CHANGED                             (20233)
#define MF_UNKNOWN_RESPONSE_TYPE                                 (20234)
#define MF_CANT_FIND_MATCHES_IF_LOCAL_DATABASE_EXISTS            (20235)
#define MF_READING_PAST_END_OF_BUFFER                            (20236)
#define MF_INVALID_LOOKUP_INDEX_VERSION                          (20237)
#define MF_REFERENCE_HOP_NOT_INTEGRAL_MULTIPLE_OF_UNKNOWN_HOP    (20238)
#define MF_FINGERPRINTER_NOT_READY                               (20239)
#define MF_COULDNT_STAT_FFMPEG_LAST_RESORT_EXECUTABLE            (20240)
#define MF_FINGERPRINT_ALREADY_IN_DATABASE                       (20241)
#define MF_FINGERPRINT_NOT_IN_DATABASE                           (20242)
#define MF_IDENGINE_NOT_IN_SERVER_MODE                           (20243)
#define MF_AMITEMID_NOT_UNIQUE                                   (20244)
#define MF_OFFSET_TOO_LARGE                                      (20245)
#define MF_NONINTEGRAL_NUMBER_OF_FRAMES                          (20246)
#define MF_CANT_CHANGE_SEGMENT_SIZE_NOR_HOP_SIZE              (20247)
#define MF_DEPRECATED_FUNCTION                                (20248)
#define MF_FEATURES_ON_DISK_DONT_MATCH                        (20249)
#define MF_NO_CONFIGURATION                                   (20250)
#define MF_COULDNT_RENAME_FILE                                (20251)
#define MF_ERROR_PARSING_MEDIA_FILE_INFO                      (20252)
#define MF_COULDNT_DECODE_MEDIA_FILE                          (20253)
```

C typedefs

Note that the authoritative source for these typedefs is the header file that ships with your particular version of the SDK.

```
typedef MFMediaIDOpaque* MFMediaID;
```

This is the main object; it holds configuration information for the specific client application and is passed as an argument to many of the SDK calls. It can be created once and used multiple times.

```
typedef void (*MFErrorCallbackFunction)(int mfErrorNumber, MFMediaID mediaID,
void* userData);
```

This defines a callback function that the library will call to report errors, primarily ones that occur in separate threads. See the description of the function `MFRegisterErrorCallback`, later in this document.

```
typedef MFMediaIDRequestOpaque* MFMediaIDRequest;
```

This object represents the identification request that will be sent to the ID server.

```
typedef MFMediaIDResponseOpaque* MFMediaIDResponse;
```

This object represents the response from the ID server. There are a few routines to help extract the nature of the response.

```
typedef void (*MFMediaIDCallback) (void* userData, MFMediaIDResponse*
response);
```

This defines a callback function that receives the result of a positive identification ("hit"). See the description of the function `MFMediaID_RegisterIDResponseCallback`, later in this document.

```
typedef enum
{
    MF_MEDIAID_RESPONSE_UNKNOWN = 0,
    MF_MEDIAID_RESPONSE_NOT_FOUND,
    MF_MEDIAID_RESPONSE_FOUND
} MFResponseStatus;
```

See the description of the MFResponseStatus enum under MFMediaIDResponse_GetIDStatus().

```
typedef enum
{
    MF_16BIT_SIGNED_LINEAR_PCM,
} MFSampleFormat;
```

Some functions accept audio samples directly rather than reading them from files. Currently only one value of MFSampleFormat is defined (zero, meaning that a per-channel sample is in linear pulse-code-modulation format and occupies16 bits, little-endian, with a signed value).

```
typedef struct MFMediaIDDebugDataOpaque* MFMediaIDDebugData2;
```

This object is created by `MFMediaID_GetDebugData()` and is used for debugging. See the description of `MFMediaID_GetDebugData`.

```
typedef const char* MFFilePath;
typedef char* MFString;
typedef const char* MFConstString;
```

These data types are used for some function arguments in order to be more self-documenting.

Other C Definitions

This section lists some C definitions that are referred to in the “Functions” section of this document. The authoritative source for these definitions is the header file that ships with your particular version of the SDK.

```
#define MF_ERROR_STRING_LENGTH      (100)
#define MF_VERSION_STRING_LENGTH    (20)
```

Utility Functions

This section describes the OEM API's C functions, and their Java equivalents, that initialize the library, handle errors, and retrieve version information.

MFGlobalInit

This routine must be called before any other functions in the SDK. It must be called only once, and it must return without error. It allocates a global mutex and initializes cURL, among other things. (For this reason, it should not be called at the same time that the cURL function curl_global_init is called on any other thread.) Neglecting to call MFGlobalInit prior to invoking any other functions in the SDK will lead to erroneous and unpredictable behavior.

C

```
MFError MFGlobalInit();
```

Java on Android

```
public long MFMediaIDJNI.GlobalInit();
```

> In previous versions, MFGlobalInit was called from the Java OnLoad method and was not exposed to the JNI interface. Starting with version 25.12, applications must explicitly call `MFMediaIDJNI.GlobalInit()` or `MFGlobalInitWithPrivateFolder()` after creating the JNI object.

Remarks

Introduced in SDK version 20.7f, modified in 25.12.

See Also: `MFGlobalInitWithPrivateFolder`

MFGlobalInit_WithPrivateDataFolder

This routine must be called before any other functions in the SDK. It must be called only once, and it must return without error. It allocates a global mutex and initializes cURL, among other things. (For this reason, it should not be called at the same time that the cURL function curl_global_init is called on any other thread.) This version of MFGlobalInit will let you pass in a relative or absolute path to a file system directory when the SDK can write the userGUID and temp files. You should only use this routine if you do not have write access to the directory where you are running your application.

C

```
MFError MFGlobalInit_WithPrivateDataFolder(MFFilePath
strPrivateDataFolder);
```

Java on Android

```
public long MFMediaIDJNI.GlobalInit_WithPrivateDataFolder(String
strPrivateDataFolder);
```

In previous versions, MFGlobalInit was called from the Java OnLoad method and was not exposed to the JNI interface. Starting with version 25.12, applications must explicitly call `MFMediaIDJNI.GlobalInit()` or `MFGlobalInitWithPrivateFolder()` after creating the JNI object.

MFRegisterErrorCallback

This routine registers the one and only error callback function. If the function is not invoked the application will not receive error callbacks, meaning that errors returned by the SDK's internal threads will never be reported. The `errorCallback` argument specifies a pointer to an application-defined **`MFErrorCallbackFunction`** (see `C Typedefs` section, above), and the specified `userData` (or `NULL`) will be passed to that function.

C

```
MFError MFRegisterErrorCallback(
            MFErrorCallbackFunction errorCallback,
            void* userData);
```

Java on Android

`The MFMediaIDJNI` java class automatically registers an internally defined error callback function. When you create your instance of an `MFMediaIDJNI` class, you must do so with the constructor that takes a java Handler object, for example:

```
myMFMediaIDJNI = new MFMediaIDJNI(myHandler);.
```

If the object is created in this way, then asynchronous error codes will be sent to the Handler for processing by your Handler.

Remarks

`MFRegisterErrorCallback` should be called after `MFGlobalInit` but before any other functions in the SDK. See an example of an application-defined error callback function in `mediaIDUtils.c`, which is used by the sample programs distributed with this release.

For the Error Callback method signature you must implement, see the “C Typedefs” section and the discussion immediately after these remarks.

Since this routine is called from an internal SDK thread, you should perform as little computation as possible. Typically you should send a message to your main thread so it can take care of any necessary operations. Do not call any Audible Magic API functions from within the callback.

void (*MFErrorCallbackFunction) (int mfErrorNumber, MFMediaID mediaID, void* userData);

mfErrorNumber is suitable for passing to the MFError routines to get more detailed information.

mediaID is the pointer to the MFMediaID object that is reporting the error. Note that when an error is not associated with any MFMediaID object then mediaID will be NULL.

userData is the value that was passed in to MFRegisterErrorCallback. The library ignores this value but returns it (by reference) so that the application can refer to its own data after receiving this callback.

This is the method definition of the error callback function which is passed to MFRegisterErrorCallback. When an error occurs in the MFCBR OEM library, this routine will be invoked.

When this routine is invoked, your application should pass the mfErrorNumber, mediaID pointer, and userData pointer back to the main thread in a thread-safe manner (such as a thread-safe queue). Then exit this routine. We recommend that you place very little application logic inside this callback routine.

If you receive errors during application development, please don't try to work around them. Instead contact Audible Magic support.

Error handling in the production version of an application is always a complex issue. If you receive an error callback with a specific MFMediaID object, your application could try to destroy the object and then create a new one. If that also fails it might be best to disable the ACR part of your application until you can have the application shut down and restart.

Remarks

If MFRegisterErrorCallback is called more than once, the most recent call is the effective one.

Your application should not make any API calls to MFMediaID objects when in this callback routine.

See Also: `MFError_GetCode` and `MFError_GetDescription`

MFError_GetDescription

Returns, by reference, a human-readable description string for any `MFError`.

C

```
MFError  MFError_GetDescription(
            MFError err,
            MFString strDescription,
            int strLen);
```

Java on Android

```
public String MFMediaIDJNI.GetDescription(
            long errorNum);
```

If the `MFMediaIDJNI` java class is created with the Handler argument, then asynchronous error codes will be passed back via the `ErrorCallback` mechanism to that Handler. You can use the `GetDescription` method to pass in an error number and retrieve the same string that is generated by exceptions.

Remarks

For the C API, caller allocates space for description string. `strLen` is the length in bytes of the char array. The error description string will be copied into description up to `strLen`-1 and null-terminated. A value of at least `MF_ERROR_STRING_LENGTH` is recommended, to avoid retrieving a truncated string.

Returns `MF _NULL_POINTER` if `strDescription` is zero, MF_PARAMETER_OUT_OF_RANGE if `strLen` is not positive or if `err` is not a valid Audible Magic error code, and MF_ SUCCESS otherwise.

MFError_GetDescription handles both complex and simple error codes, so you do not need to call MFError_GetCode first.

Replaces `MFGetErrorDescription`, which was deprecated in SDK version 20.29.

See Also: `MFError_GetCode` and the “Error Codes” section of this document.

MFError_GetCode

Convert err and return an `MFError` code that can then be looked up in the list contained in the `mfErrors_oem.h` file.

C

```
MFError  MFError_GetCode(
            MFError err);
```

Java on Android

```
N/A
```

Remarks

The MFError value returned by most of the functions is a complex value that helps Audible Magic determine where the error occurred. For customers who want to take action based on certain specific errors, they should call this routine to extract the simple error code from the complex value.

See Also: `MFError_GetDescription` and the “Error Codes” section of this document.

MFGetLibraryVersion

Returns, by reference, the current implementation version string of the MF API. The version string is changed each time a new version of the OEM library is released by Audible Magic. Normally, the client application will not care about the version string because implementations are backwards compatible. However, in troubleshooting a problem, Audible Magic Support Engineers may request the version string. An application with a well-designed user interface will provide some mechanism to report this string when requested.

C

```
MFError MFGetLibraryVersion(
            MFString* strVersion,
            int strLen);
```

Java on Android

```
public String MFMediaIDJNI.Version();
```

Remarks

For the C API, the caller allocates space for the version string. `strLen` is the length in bytes of the version char array; it should be MF_VERSION_STRING_LENGTH bytes. If less, the string will be truncated. The library's version string will be copied into `strVersion` up to `strLen`-1 and null-terminated.

For the Java API, the function returns a Java String containing the version string.

Configuration Functions

This section describes the OEM API's C functions, and their Java equivalents, for creating MFMedia objects, which are initialized with specific configuration information. It also describes API for destroying these objects and for creating local databases of reference fingerprints.

MFMediaID_CreateUsingLicenseKey

This routine is called before any of the other identification API calls. The MFMediaID object is the workhorse of the Audible Magic library. Once you create a MFMediaID object, you will then pass it to other API calls.

The license key is a string provided to you by Audible Magic. The license key is tied to a specific application and the license key controls much of the library behavior.

This call blocks until the library is able to contact the Audible Magic cloud service to validate the license key. Once the license key is validated, this call will return and your application can continue using the library.

C

```
MFError MFMediaID_CreateUsingLicenseKey(
            MFMediaID* mediaID,
            const char* strLicenseKey);
```

Java on Android

```
public long MFMediaIDJNI.MediaIDCreateUsingLicenseKey(
            String strLicenseKey);
```

Remarks

Each `MFMediaID` object can use only one license key. If your application uses more than one license key then you will need to create and manage one `MFMediaID` object for each license key.

`MFMediaID` objects are intended to be long-lived. Your application can do many, many media identifications with one `MFMediaID` object.

Note for iOS: when your application is put into the background, you should, upon retaking the foreground, destroy all your `MFMediaID` objects and create them again. Note that the notification happens asynchronously, thus you should be careful about any threading issues in your application code. The Audible Magic SDK is thread-safe, but if you destroy an object and then try to access it through an API call, we will return error 20187, `MF_MEDIAID_ALREADY_DESTROYED`.

MFMediaID_CreateUsingConfigFile

Similar to `MFMediaID_CreateUsingLicenseKey` except that this method allows the application to initialize with a configuration file provided by Audible Magic. License keys are the preferred method of initialization. However, in some cases Audible Magic may require your application to use a configuration file.

With this call, the caller specifies the path to the configuration file.

C

```
MFError MFMediaID_CreateUsingConfigFile (
            MFMediaID* mediaID,
            MFFilePath strPathToConfigFile);
```

Java on Android

```
public long MFMediaIDJNI.MediaIDCreate(
            String strPathToConfigFile);
```

Throws java Exception on error.

Remarks

Each `MFMediaID` object can use only one configuration file. If your application uses more than one configuration file then you will need to create and manage one `MFMediaID` object for each configuration file.

See Also: `MFMediaID_CreateUsingLicenseKey and MFMediaID_CreateUsingXMLString`

MFMediaID_CreateUsingXMLString

Similar to `MFMediaID_CreateUsingConfigFile` except that this method takes a string that contains the entire contents of the configuration file. In other words, the caller must first read the full contents of the desired configuration file, including all its whitespace characters (newlines, carriage returns, tabs, spaces, etc., without modification), into `strXML` which will then be passed to this method.

C

```
MFError MFMediaID_CreateUsingXMLString (
            MFMediaID* mediaID,
            const char* strXML);
```

Java on Android

```
public long MFMediaIDJNI.MediaIDCreateUsingXMLString(
            String strXML);
```

Throws java Exception on error.

Remarks

This method is useful when Audible Magic requires your application to use a configuration file and the specifics of your implementation prevent the library from easily gaining access to the file system. Note that you must copy all of the contents, verbatim, from the configuration file into the target string or this method will return an error.

See Also: `MFMediaID_CreateUsingLicenseKey and MFMediaID_CreateUsingConfigFile`

MFMediaID_Destroy

This routine must be invoked to free memory and release other resources associated with an `MFMediaID` object. It should be called only when the caller is finished with all identification work or the application is about to exit.

C

```
MFError MFMediaID_Destroy(
            MFMediaID* mediaID);
```

Java on Android

```
public long MFMediaIDJNI.MediaIDDestroy(
            long mediaID);
```

Throws java Exception on error.

Remarks

Notice that the argument to the C function has an asterisk, meaning that the argument is the *address* of the MFMediaID object (which is itself defined to be a pointer). This is so that MFMediaID_Destroy() can set your local variable to NULL after the memory is freed.

Note for iOS: when your application is put into the background, you should, upon retaking the foreground, destroy all your MFMediaID objects and create them again. Note that the notification happens asynchronously, thus you should be careful about any threading issues in your application code. The Audible Magic SDK is thread-safe, but if you destroy an object and then try to access it through an API call, we will return error 20187, `MF_MEDIAID_ALREADY_DESTROYED`.

MFMediaID_AddFileToLocalDatabase

If your license key specifies a local database lookup, use this routine to specify the AMDB file to be used for subsequent identification attempts. The `mediaID` object must have already been created using one of the `MFMediaID_CreateUsingLicenseKey`, `MFMediaID_CreateUsingConfigFile`, or `MFMediaID_CreateUsingXMLString` calls. `strPathToDatabase` is the path to the local database. Each time this function is called, the specified database is added to the database for local lookups. There is no way to remove a database after it is loaded. If you need to take a database out of use, destroy the mediaID object and create a new one, adding only the database files you wish to use.

C

```
MFError MFMediaID_AddFileToLocalDatabase(
            MFMediaID mediaID,
            MFFilePath strPathToDatabase);
```

Java on Android

```
public long MFMediaIDJNI.AddFileToLocalDatabase(
            long mediaID,
            String strDBFilename);
```

Remarks

Replaces the deprecated function, `MFMediaID_SetDatabase` in SDK version 20.29. Replaces the deprecated function `MFMediaID_LoadDatabaseFromFile` in SDK version 21.11.

Note that you must use the amDbBuilder application, available from Audible Magic in a separate package, to generate your databases beforehand.

Your license key must allow for local lookups or this call will return an error.

If your mediaID object is currently listening, you must call MFMediaID_StopListening before calling this routine, or MFMediaID_AddFileToLocalDatabase will return an error.

Functions for Identifying Stored Media

This section describes the OEM API's C functions, and their Java equivalents, for identifying "stored" media: either media files or in-memory buffers of media data. ("Stored" media is distinguished from "real-time" media; the latter is captured from the microphone or from another source of streaming data. See "Functions for Identifying Real-Time Media.")

The two recommended functions for identifying stored media are `MFMediaID_IdentifyFile` and `MFMediaID_IdentifySamples`. However, this section also includes some functions that can be used when the application wants to do fingerprinting first, and a remote identification at some later time.

MFMediaID_IdentifyFile

This routine is called to identify a media file. The `mediaID` object must have already been created using one of the `MFMediaID_Create...` calls. `strMediaFile` is the path to the media file. The duration of the media content and the location of that content within the media file is specified by Audible Magic in the configuration information controlled by the license key.

`strAssetID` is an arbitrary caller-supplied string. It is preferable that it be uniquely associated with the media file, e.g., a catalog number, for tracking purposes, but that is not necessary for the identification process to work correctly. (A response from a remote ID server will include, within its XML, the asset ID that was specified in this call.)

Handling of the `response` argument is the same as described for `MFMediaID_IdentifyNow()`.

Note: Not supported on iOS or Android; passing the return value to `MFError_GetCode()` will yield MF_CANNOT_PROCESS_MEDIA_FILES. Instead, use `MFMediaID_IdentifySamples`.

C

```
MFError MFMediaID_IdentifyFile(
            MFMediaID mediaID,
            MFFilePath strMediaFile,
            MFConstString strAssetID,
            MFMediaIDResponse* response);
```

Java on Android

```
N/A
```

Remarks

This replaces the deprecated call `MFMediaID_GenerateAndPostRequest`.

Prior to 2013, this function was not supported for local database lookup, only for posting a request to a remote ID server. That restriction has been removed.

In general, this call will handle any file type that can be decoded by FFmpeg. (Within the call to `MFMediaID_IdentifyFile`, the library calls out to the FFmpeg executable for conversion to a form that the library can handle..)

If this function, or any other function that sends ID requests, is called too frequently by simultaneous processes, it may return error 5429 (“HTTP status code 429”). This indicates you have exceeded the number of concurrent transactions allowed in your contract with Audible Magic. When this occurs, retry the transaction after waiting, say, ten seconds, and/or reduce the number of processes that are concurrently sending requests to Audible Magic's servers.

See Also: `MFMediaID_IdentifySamples`, `MFMediaID_IdentifyNow`

MFMediaID_IdentifySamples

This is analogous to the `MFMediaID_IdentifyFile` call except that the media to be identified is captured and held by the application in a sample buffer.

`samples` is a pointer to a region of memory containing `numFrames` (mono or stereo) samples, sampled at rate `sampleRate`, of `numChannels`, and sample format `format`. Your application must ensure that the audio sample format is 16-bit, signed, little-endian, linear PCM. (See the description of MFSampleFormat in the "C Typedefs" section of this document.)

It is crucially important that the parameters describing the contents of the sample buffer be accurate. If they are not accurate you will not receive an error, but you will not have any positive IDs. Handling of the `response` argument is the same as described for `MFMediaID_IdentifyNow()`.

C

```
MFError MFMediaID_IdentifySamples(
            MFMediaID mediaID,
            const void* samples,
            int numFrames,
            float sampleRate,
            int numChannels,
            MFSampleFormat format,
            MFConstString strAssetID,
            MFMediaIDResponse* response);
```

Java on Android

```
static public int MF_16BIT_SIGNED_LINEAR_PCM    = 0;
/* use MF_16BIT_SIGNED_LINEAR_PCM for sampleFormat, below */

public long MFMediaIDJNI.IdentifySamples(
            long mediaID,
            byte samples[],
            int numFrames,
            float sampleRate,
            int numChannels,
            int sampleFormat,
            String strAssetID);
```

Throws java Exception on error.

Remarks

For example, a buffer of 6 seconds of stereo PCM data might be:

sampleRate = 22050
numChannels = 2
numFrames = 132300

You would expect the sample buffer in this case to be 529200 bytes long (6 seconds × 22050 samples per second × 2 bytes per sample × 2 channels).

This call replaces the deprecated call `MFMediaID_GenerateAndPostRequestFromSamples`.

Prior to 2013, this function was not supported for local database lookup, only for posting a request to a remote ID server. That restriction has been removed.

The `strAssetID` argument is currently (2013) not put into the ID request. In a future release, it will be, just as it currently is when passed to `MFMediaID_IdentifyFile`.

The buffer passed to `MFMediaID_IdentifySamples` must contain all the audio necessary to get an identification. The library does not "remember" a previously passed buffer; in other words, an identification cannot span successive buffers. If your application needs that functionality and is processing streaming audio—that is, a real-time signal such as microphone input rather than content stored in memory—see `MFMediaID_IdentifyNow` or the continuous lookup scenario that uses `MFMediaID_RegisterIDResponseCallback`. (However, if your application is processing streaming audio from a source other than the microphone or default sound input, see the advanced API `MFListenToSamples`.) In any of these cases of real-time input, you will need a license key or configuration file from Audible Magic that allows that functionality.

See Also: `MFMediaID_IdentifyFile, MFMediaID_IdentifySamplesWithTimestamp, MFMediaID_IdentifyNow, MFListenToSamples`

MFMediaID_GenerateRequest

This routine allows an application to keep the fingerprint generation activity separate from the identification activity. This routine is only supported for remote database lookup. It creates an `MFMediaIDRequest` object which can then be sent to the remote identification service using the `MFMediaID_PostRequest` call.

Note: Not supported on iOS and Android; passing the return value to `MFError_GetCode()` will yield MF_CANNOT_PROCESS_MEDIA_FILES.

C

```
MFError MFMediaID_GenerateRequest(
            MFMediaID mediaID,
            MFFilePath strMediaFile,
            MFConstString strAssetID,
            MFMediaIDRequest* request);
```

Java on Android

```
public long MFMediaIDJNI.GenerateRequest(
            long mediaID,
            String strMediaFile,
            String strAssetID);
```

Throws java Exception on error.

Remarks

An application would typically use `MFMediaID_IdentifyFile` instead of this function. `MFMediaID_IdentifyFile` is essentially equivalent to `MFMediaID_GenerateRequest` followed by `MFMediaID_PostRequest` and `MFMediaIDRequest_Destroy`. However, `MFMediaID_IdentifyFile` can be used for either remote or local lookups, whereas `MFMediaID_GenerateRequest` is used only for remote lookups.

See Also: `MFMediaID_IdentifyFile`, `MFMediaID_PostRequest`, `MFMediaIDRequest_Destroy`

MFMediaID_GenerateRequestFromSamples

This is analogous to `MFMediaID_GenerateRequest` except that the audio to be identified is held by the application in a sample buffer instead of in a file.
Please review the important notes under `MFMediaID_IdentifySamples`.

C

```
MFError MFMediaID_GenerateRequestFromSamples(
            MFMediaID mediaID,
            const void* samples,
            int numFrames,
            float sampleRate,
            int numChannels,
            MFSampleFormat format,
            MFConstString strAssetID,
            MFMediaIDRequest* request);
```

Java on Android

```
static public int MF_16BIT_SIGNED_LINEAR_PCM    = 0;
/* use MF_16BIT_SIGNED_LINEAR_PCM for sampleFormat, below */

public long MFMediaIDJNI.GenerateRequestFromSamples(
            long mediaID,
            byte samples[],
            int numFrames,
            float sampleRate,
            int numChannels,
            int sampleFormat,
            String strAssetID);
```

Throws java Exception on error.

Remarks

An application would typically use `MFMediaID_IdentifySamples` instead of this function. `MFMediaID_IdentifySamples` is essentially equivalent to `MFMediaID_GenerateRequestFromSamples` followed by `MFMediaID_PostRequest` and `MFMediaIDRequest_Destroy`. However, `MFMediaID_IdentifySamples` can be used for either remote or local lookups, whereas `MFMediaID_GenerateRequestFromSamples` is used only for remote lookups.

See Also: `MFMediaID_IdentifySamples`, `MFMediaID_PostRequest`, `MFMediaIDRequest_Destroy`

MFMediaIDRequest_GetStringLength

This routine is provided for applications that need to keep the fingerprint generation activity separate from the identification activity. `MFMediaIDRequest_GetAsString` requires a pre-allocated buffer to hold the returned request string. Before calling that function, call `MFMediaIDRequest_GetStringLength` to get the required size of the allocation (including the null terminator). The `request` argument is the `MFMediaIDRequest` object returned by the call to functions such as `MFMediaID_GenerateRequestFromSamples`.

C

```
MFError MFMediaIDRequest_GetStringLength(
            MFMediaIDRequest request,
            int* length);
```

Java on Android

No need for this API. Just call `MFMediaIDJNI.RequestGetAsString`.

See Also: `MFMediaIDRequest_GetAsString`

MFMediaIDRequest_GetAsString

This routine is provided for applications that need to keep the fingerprint generation activity separate from the identification activity. This routine is used to retrieve the ID request as an XML string from the `MFMediaIDRequest` object. `request` is an `MFMediaIDRequest` object returned by one of the request-generation API calls (e.g., `MFMediaID_GenerateRequestFromSamples`). `strRequest` is a caller-allocated buffer that will hold the string on return. `strLength` is the allocated size of `strRequest`. (An error is returned if `strLength` is too small for the string. Use `MFMediaIDRequest_GetStringLength` to obtain the right value.)

C

```
MFError MFMediaIDRequest_GetAsString(
            MFMediaIDRequest request,
            MFString strRequest,
            int strLength);
```

Java on Android

```
public String MFMediaJNI.RequestGetAsString(
      long request);
```

Throws java Exception on error.

Note: On the Android platform, the API programmer does not need to call `MFMediaIDRequest_GetStringLength` as the internal Java code will properly allocate a string of the correct size.

Remarks

In some error conditions the returned string may not be well-formed XML.

See Also: `MFMediaIDRequest_GetStringLength`

MFMediaIDRequest_CreateUsingString

This routine is provided for applications that need to keep the fingerprint-generation activity separate from the identification activity. If the `MFMediaIDRequest` object is stored in a file or in memory using the `MFMediaIDRequest_GetAsString` function, then it must be converted back into an `MFMediaIDRequest` object before being posted to the ID server for identification. After calling this routine you may use the request object in routines such as `MFMediaID_PostRequest`.

C

```
MFError MFMediaIDRequest_CreateUsingString(
            MFString strRequest,
            MFMediaIDRequest* request);
```

Java on Android

```
public long MFMediaIDJNI.RequestCreateUsingString(
            String strRequest);
```

Throws java Exception on error.

MFMediaID_PostRequest

This routine is used by applications that need to keep the fingerprint generation activity separate from the remote identification activity. This routine is used to send a previously generated `MFMediaIDRequest` to the remote Audible Magic ID servers and receive an identification response.

Your application passes to this routine an `MFMediaIDRequest` object that has either been generated in this run of the application by `MFMediaID_GenerateRequest()` or `MFMediaID_GenerateRequestFromSamples()`, or else generated in a previous run (possibly of another application), stored somewhere, and then recreated by this application via `MFMediaIDRequest_CreateUsingString()`..

Handling of the `response` argument is the same as described for `MFMediaID_IdentifyNow()`. It is the caller's responsibility to deallocate the request and the response by calling `MFMediaIDRequest_Destroy()` and `MF_MediaIDResponse_Destroy()` when done with them.

C

```
MFError MFMediaID_PostRequest(
            MFMediaID mediaID,
            MFMediaIDRequest request,
            MFMediaIDResponse* response);
```

Java on Android

```
public long MFMediaIDJNI.PostRequest(
            long mediaID,
            long request);
```

Throws java Exception on error.

Remarks

An application that would typically use `MFMediaID_IdentifyFile` could, alternatively, use `MFMediaID_GenerateRequest` and store the request for later use by `MFMediaID_PostRequest`.

See Also: `MFMediaID_IdentifyFile`, `MFMediaIDRequest_CreateUsingString`, `MFMediaID_GenerateRequest`, `MFMediaIDRequest_Destroy`

MFMediaIDRequest_Destroy

This routine must be called to free the memory associated with the `MFMediaIDRequest` object that was created by `MFMediaID_GenerateRequest` or `MFMediaID_GenerateRequestFromSamples`.

C

```
MFError MFMediaIDRequest_Destroy(
            MFMediaIDRequest* request);
```

Java on Android

```
public int MFMediaIDJNI.RequestDestroy(
            long request);
```

Remarks

Notice that the argument to the C function has an asterisk, meaning that the argument is the *address* of the `MFMediaIDRequest` object (which is itself defined to be a pointer). This is so that `MFMediaIDRequest_Destroy` can set your local variable to NULL after freeing the memory.

Functions for Identifying Real-Time Media

This section describes the OEM API's C functions, and their Java equivalents, for identifying "real-time" media such as audio that is in the process of being captured from the default sound input, which is usually a microphone. (To instead identify media files or in-memory media content, see "Functions for Identifying Stored Media.")

MFMediaID_IdentifyNow

This routine will analyze microphone input (or the system's default audio input) for the Duration specified by the license key (or configuration file or XML sting) that was used in the Create call for this mediaID object. As soon as the analysis is complete, the library will attempt to identify the audio, using the process (local or remote lookups) specified in the configuration data controlled by the license key.

The opaque argument `response` will be allocated and returned by the call. The status of the identification attempt can subsequently be determined by calling `MFMediaIDResponse_GetIDStatus` on `response`, and one can convert `response` to a string, for additional parsing, by passing it to `MFMediaIDResponse_GetAsString`. Note that it is the caller's responsibility to free memory and other resources associated with `response` by calling `MFMediaIDResponse_Destroy`.

In the case of a non-zero "Preroll" configuration (see "On-Demand Identification of Live Audio Input" elsewhere in this document), you must call `MFMediaID_StartListening` before calling `MFMediaID_IdentifyNow`; if not, this method will return the error `MF_MEDIA_ID_NOT_LISTENING`.

If Preroll is 0 (the default), it is unnecessary to call `MFMediaID_StartListening` beforehand. In this case the call to `MFMediaID_IdentifyNow` will do an implicit `MFMediaID_StartListening` and `MFMediaID_StopListening` to gather the needed audio.

C

```
MFError MFMediaID_IdentifyNow(
            MFMediaID mediaID,
            MFMediaIDResponse* response);
```

Java on Android

```
long MFMediaIDJNI.IdentifyNow(
            long mediaID);
```

MFMediaID_RegisterIDResponseCallback

This routine is used only when your license key specifies AllowContinuousLookups to be TRUE; this routine is not to be used when you are calling `MFMediaID_IdentifyNow` (or `MFMediaID_IdentifyFile` or `MFMediaID_IdentifySamples)`. The `mediaID` object must have already been created using one of the `MFMediaID_Create...` calls. The `callback` argument specifies a pointer to an application-defined `MFMediaIDCallback` function (see the "C Typedefs" section in this document). The specified `userData` (which can be `NULL`) will be passed back to that function.

The callback function is invoked when a lookup attempt results in a positive identification. (The license key can override this default behavior in order to allow negative IDs to be reported for continuous periodic lookup as well.) The continuous periodic look up process itself commences as soon as the `MFMediaID_StartListening` routine is invoked by the application, and it ceases soon after the application invokes `MFMediaID_StopListening`.

C

```
MFError MFMediaID_RegisterIDResponseCallback(
            MFMediaID mediaID,
            MFMediaIDCallback callback,
            void* userData);
```

Java on Android

The `MFMediaIDJNI` Java class automatically registers an internally defined ID response callback function. When you create your instance of an `MFMediaIDJNI` class, you must do so with the constructor that takes a Java Handler object, for example:

```
myMFMediaIDJNI = new MFMediaIDJNI(myHandler);
```

If the object is created in this way, asynchronous error codes will be sent to the Handler for processing by your Handler.

Remarks

Replaces the deprecated function, `MFMediaID_RegisterCallback` in SDK version 20.29.See Also: The C Typedefs section of this document for a precise description of the callback function you must implement.

Since the callback is called from an internal SDK thread, the routine should not perform any time-consuming operations. In particular, it should not make any API calls to the library.

void (*MFMediaIDCallback) (void* userData, MFMediaIDResponse* response);

userData is the value that was passed in to MFMediaID_RegisterIDResponseCallback. The library ignores this value but returns it by reference here so that the application can refer to its own data after receiving this callback.

response is a pointer to an object that is suitable for using with the various MFMediaIDResponse API calls.

This is the method definition of the identification callback function which is passed to MFMediaID_RegisterIDResponseCallback. When a continuous lookup mode mediaID object has a positive identification, this function will be invoked.

When this routine is invoked, your application should pass the userData and response object pointer back to the main thread in a thread-safe manner (such as a thread-safe queue). Then exit this routine. We recommend that you place very little application logic inside this callback routine. In particular, you should not make any calls to the Audible Magic library while inside the callback routine. Calling the library from within the callback function can lead to deadlocks.

Remarks

If MFMediaID_RegisterIDResponseCallback is called more than once, the most recent call is the effective one.

Be aware that this routine will be called from a thread within an MFMediaID object. If your userData is a pointer to a shared data structure, you must be very careful to lock accesses to that structure to avoid data inconsistency errors. Also, we recommend you do as little processing as possible during the callback routine.

See Also: MFMediaIDResponse API calls

MFMediaID_StartListening

Used to tell the `mediaID` object to begin listening to the default audio stream (microphone or line-in). The `mediaID` object must have already been created using one of the `MFMediaID_Create...` calls.

If your license key specifies continuous lookup mode, then the continuous lookup process commences as soon as the `MFMediaID_StartListening` routine is invoked by the application, and it ceases soon after the application invokes `MFMediaID_StopListening`. Between these two points, the continuous lookup process reports positive IDs via the callback routine that you had specified in `MFMediaID_RegisterIDResponseCallback`.

C

```
MFError MFMediaID_StartListening(
            MFMediaID mediaID);
```

Java on Android

```
public long MFMediaIDJNI.StartListening(
                  long mediaID);
```

Remarks

In continuous lookup mode, the identification process will commence as soon as this call is made, but your application will not receive any identification results until sometime after the Duration of audio specified in your license key has been captured.

Typical usage is to start listening soon after the `mediaID` object is created and to continue listening for the life of the `mediaID` object.

In IdentifyNow mode, `MFMediaID_StartListening()` does not need to be called unless the PrerollDuration configuration parameter has a nonzero value. A nonzero value can result in a more immediate identification. See the discussion under “Pre-roll” elsewhere in this document.

Note well: In order to avoid conflicts and contention that can arise if multiple apps require access to the audio hardware simultaneously, it is best to insert the following statement in your Audible Magic app just prior to calling the `MFMediaID_StartListening()` function: `[AVAudioSession setCategory:AVAudioSessionCategoryPlayAndRecord error:nil];`

MFMediaID_StopListening

Used when an application must stop the library from listening. Since all MFMediaID objects share a common low-level listening and fingerprinting process, you must call this method on *all* MFMediaID objects to truly stop the low-level activity. The `mediaID` object must have already been created using one of the `MFMediaID_Create...` calls. The continuous lookup process commences as soon as the application invokes `MFMediaID_StartListening`, and ceases as soon as the application invokes `MFMediaID_StopListening`.

C

```
MFError MFMediaID_StopListening(
            MFMediaID mediaID);
```

Java on Android

```
public long MFMediaIDJNI.StopListening (
            long mediaID);
```

Remarks

Applications may call `MFMediaID_StopListening` if they wish, but this will increase the time required for the next identification to occur (See the discussion of Duration and Preroll under `MFMediaID_IdentifyNow`). A typical application will only call `MFMediaID_StopListening` when it truly wants to stop recognizing content for an extended period of time.

MFMediaID_IsListening

`pBoolIsListening` is passed by reference, and upon return will contain

- `TRUE`, if the caller has previously invoked `MFMediaID_StartListening` on the specified `mediaID` object, or
- `FALSE`, if the caller either has never called `MFMediaID_StartListening` or if the caller has previously invoked `MFMediaID_StopListening`.

C

```
MFError MFMediaID_IsListening(
            MFMediaID mediaID,
            MFBoolean* pBoolIsListening);
```

Returns an error which (after processing by `MFError_GetCode()`) will be `MF_NULL_POINTER` if `mediaID` or `pBoolIsListening` is `NULL (0)`.

Java on Android

N/A

Remarks

A long-lived monitoring application may make use of the `MFMediaID_StopListening` call. In that case `MFMediaID_IsListening` is provided so that the application does not have to maintain variables that hold the listening state of each `mediaID` object.

Functions for Handling ID Responses

This section describes the OEM API's C functions, and their Java equivalents, for examining the response to an identification request. The request could have been made for "stored media" or "real-time media." In the latter case the ID response could have been returned by `MFMedia_IdentifyNow` or (in the continuous lookup scenario) in the application-provided ID response callback routine.

MFMediaIDResponse_GetIDStatus

This routine parses `response` to determine if the media was successfully identified. `response` is an `MFMediaIDResponse` object returned by one of the identification API calls or in the ID callback routine. If `MFMediaIDResponse_GetIDStatus` returns something other than MF_SUCCESS, your application should use the MFError_ calls to get additional information.

If this routine returns MF_SUCCESS, `MFResponseStatus` is an enum with three possible values:

- `MF_MEDIAID_RESPONSE_FOUND` means that the media was successfully identified,
- `MF_MEDIAID_RESPONSE_NOT_FOUND` means that the media was not identified, and
- `MF_MEDIAID_RESPONSE_UNKNOWN` means that the status could not be determined from the server response. The caller would have to look at the XML in the `response` object more closely to determine the reason. The XML might contain a field called MiscInfo containing relevant information.

If the server response was not well-formed XML, due to server error, overload, or corruption over the network, the function that posted the request would have returned an applicable error and would not have created an `MFMediaIDResponse` object, meaning there would be no object to be queried by `MFMediaIDResponse_GetIDStatus`.

C

```
MFError MFMediaIDResponse_GetIDStatus(
            MFMediaIDResponse response,
            MFResponseStatus* status);
```

Java on Android

```
public long MFMediaIDJNI.ResponseGetIDStatus(
            long response);
```

Throws java Exception on error.

See Also: `MFMediaIDResponse_GetAsString`

MFMediaIDResponse_GetStringLength

`MFMediaIDResponse_GetAsString` requires a pre-allocated buffer to hold the returned response string. Before calling that function, call `MFMediaIDResponse_GetStringLength` to get the required size of the allocation (including the null terminator). The `response` argument is the `MFMediaIDResponse` object returned by the call to functions such as `MFMediaID_IdentifyNow` and `MFMediaID_IdentifyFile` (or, in the case of continuous lookup, by your function that you passed to `MFMediaID_RegisterIDResponseCallback`).

C

```
MFError MFMediaIDResponse_GetStringLength(
            MFMediaIDResponse response,
            int* length);
```

Java on Android

Note: On the Android platform, the API programmer does not need to call `MFMediaIDResponse_GetStringLength` as the internal Java code will properly allocate a string of the correct size.

See Also: `MFMediaIDResponse_GetAsString`

MFMediaIDResponse_GetAsString

This routine is used to retrieve the ID response as an XML string from the `MFMediaIDResponse` object. `response` is an `MFMediaIDResponse` object returned by one of the identification API calls or in the ID callback. `strResponse` is a caller-allocated buffer that will hold the string on return. `strLength` is the allocated size of `strResponse`. (An error is returned if `strLength` is too small for the string. Use `MFMediaIDResponse_GetStringLength` to obtain the right value.)

C

```
MFError MFMediaIDResponse_GetAsString(
            MFMediaIDResponse response,
            MFString strResponse,
            int strLength);
```

Java on Android

```
public String MFMediaIDJNI.ResponseGetAsString (
            long response);
```

Throws java Exception on error.

Remarks

If your application is attempting to synchronize actions to the media the user is listening to, then please see the section of this document titled "Application Synchronization to Media."

In some error conditions the returned string may not be well-formed XML.

The application is responsible for using an XML parser to extract needed information from `strResponse`. Because the XML structure might change in the future, the application code must be able to handle the presence of new fields and so on. For this reason, a standards-compliant XML parser should be used rather than a simple string parsing.

See Also: `MFMediaIDResponse_GetStringLength`

MFMediaIDResponse_Destroy

This routine must be called to free the memory associated with the `MFMediaIDResponse` object.

C

```
MFError MFMediaIDResponse_Destroy(
            MFMediaIDResponse* response);
```

Java on Android

```
public int MFMediaIDJNI.ResponseDestroy(
            long response);
```

Remarks

Notice that the argument to the C function has an asterisk, meaning that the argument is the *address* of the `MFMediaIDResponse` object (which is itself defined to be a pointer). This is so that `MFMediaIDResponse_Destroy` can set your local variable to NULL after the memory is freed.

Functions for Debugging

This section describes the OEM API's C functions for obtaining certain debugging information related to the "real-time media" scenario. (See "Functions for Identifying Real-Time Media.")

These debugging functions are currently available in C but not Java, and therefore not on Android.

MFMediaID_SetKeepDebugData

Used only when you want certain debugging information to be made available to the application. This debug data includes the audio being captured and fingerprinted. The `mediaID` object must have already been created using one of the `MFMediaID_Create...` calls. If `boolKeepDebugData` is `TRUE`, the debug data will be available when you later call `MFMediaID_GetDebugData;` otherwise it will not be available and calling that function will return an error. If `boolKeepDebugData` is `FALSE`, and you had previously called this function with `boolKeepDebugData` set to `TRUE`, the library will stop storing debugging information.

C

```
MFError  MFMediaID_SetKeepDebugData(
            MFMediaID* mediaID,
            MFBoolean boolKeepDebugData);
```

Java on Android

```
N/A
```

Remarks

Returns an error which (after processing by `MFError_GetCode()`) will be `MF _NULL_POINTER` if `mediaID` is `NULL (0).`

MFMediaID_GetDebugData

This function is used for debugging—for example, when you are not getting positive identifications you expect and you would like to hear the audio that was recently captured, to see whether it is what you expected. `MFMediaID_GetDebugData()` returns, by reference, an `MFMediaIDDebugData2` object that you can then pass to `MFMediaIDDebugData_FillBuffers()` to retrieve an audio file and/or an encrypted XML file. (The latter contains data besides the audio and is useful only to Audible Magic's Support department).

The function also returns, by reference, the required lengths of two buffers that you will need to allocate before passing the buffers to `MFMediaIDDebugData_FillBuffers()`.
The `MFMediaID_GetDebugData()` function must be invoked after `MFMediaID_SetKeepDebugData()` has been called with the `boolKeepDebugData` argument `TRUE` and while the associated `MFMediaID` object is listening (i.e., after `MFMediaID_StartListening()` has been called and before `MFMediaID_StopListening()` has been invoked). If the function is called while the `MFMediaID` object is not currently listening, `MFMediaID_GetDebugData()` will return the `MFError MF_MEDIA_ID_NOT_LISTENING`. This implies that for `MFMediaID` objects configured to do on-demand (i.e., `IdentifyNow`) lookups, one must first call `MFMediaID_StartListening()`, then `MFMediaID_IdentifyNow()`, then `MFMediaID_GetDebugData()`, and lastly `MFMediaID_StopListening()`.

C

```
MFError MFMediaID_GetDebugData(
            MFMediaID mediaID,
            MFMediaIDDebugData2* debugData,
            unsigned int* xmlLength,
            unsigned int* wavBufferLength);
```

Java on Android

```
N/A
```

Remarks

Returns an error which (after processing by `MFError_GetCode()`) will be `MF_NULL_POINTER` if any parameter is `NULL (0)`. If `MFMediaID_SetKeepDebugData` was not previously called with `boolOnOff` set to `TRUE`, then `MFMediaID_GetDebugData` returns an error which (after processing by `MFError_GetCode()`) will be `MF_MEDIA_ID_NOT_DEBUGGING_AUDIO`.

This function replaces the deprecated function `MFMediaID_RetrieveDebugData()`.

MFMediaIDDebugData_FillBuffers

This function is used to fill one or two buffers with debugging data. Before calling it, call `MFMediaID_SetKeepDebugData` (with `boolOnOff` set to `TRUE`) and then `MFMediaID_GetDebugData()`. The latter returns, by reference, an `MFMediaIDDebugData2 object` that you pass here as the `debugData` argument.

If you want to hear the audio that was recently captured, allocate a buffer of the size specified by the `wavBufferLength` argument that `MFMediaID_GetDebugData()` returned by reference, and then pass that buffer here as `wavBuffer`. Also pass its size here as the `wavBufferLength` argument. After this function returns, write `wavBuffer` to a file (presumably named with the suffix ".wav") and play it with a media player.

This audio will be 30 seconds long, and its ending corresponds roughly to the time that `MFMediaID_GetDebugData()` was called. If you did not call `MFMediaID_SetKeepDebugData` at least 30 seconds before calling `MFMediaID_GetDebugData()`, the audio will begin with silence. (The behavior described in this paragraph might change in a future release.)

If Audible Magic's Support department is helping you debug a problem and asks you to retrieve encrypted XML, allocate a buffer of the size specified by the `xmlLength` argument that `MFMediaID_GetDebugData()` returned by reference, and then pass that buffer here as `strXML`. Also pass the size here as the `xmlLength` argument. This encrypted XML will contain data besides the audio and is useful only to Audible Magic. Probably you will write it to a file named with the suffix ".xml" and send that file to Audible Magic Support.

C

```
MFError MFMediaIDDebugData_FillBuffers(
            MFMediaIDDebugData2 debugData,
            char* strXML,
            unsigned int xmlLength,
            char* wavBuffer,
            unsigned int wavBufferLength);
```

Java on Android

```
N/A
```

Remarks

Either, but not both, of the two buffers `strXML` and `wavBuffer` may be `NULL (0)`, in which case this function does not attempt to fill that buffer. `MF_NULL_POINTER` if is returned if they both are `NULL`, or if `debugData` is. `MF_BUFFER_TOO_SMALL` is returned if either length argument is too

short, but this will not happen if you use the values that `MFMediaID_GetDebugData()` returned by reference. If you pass a length argument that is longer than necessary, the buffer's corresponding additional bytes are filled with zeros. The length argument must reflect the number of bytes you actually allocated, or memory may be corrupted and your application might crash.

MFMediaIDDebugData_Destroy

`MFMediaIDDebugData_Destroy()` deallocates memory that the library allocated internally when `MFMediaID_GetDebugData()` was called. The parameter is the `MFMediaIDDebugData2` object that `MFMediaID_GetDebugData()` returned by reference.

(In contrast, you are responsible for freeing the buffers that you allocated and passed to `MFMediaIDDebugData_FillBuffers()`.)

C

```
MFError  MFMediaIDDebugData_Destroy(
            MFMediaIDDebugData2* debugData);
```

Java on Android

```
N/A
```

Remarks

Returns an error which (after processing by `MFError_GetCode()`) will be `MF_NULL_POINTER` if the parameter is `NULL` (0), or if it points to `NULL` (which it will not if `MFMediaID_GetDebugData()` returned without error). Before returning MF_SUCCESS, this function sets `*debugData` to NULL.

This function replaces the deprecated function `MFMediaID_ReleaseDebugData()`.

Advanced API Functions

The following API calls are part of the standard MFCBR OEM library. However, these APIs expose functionality that can be complex to use. We do not recommend using the following API calls unless your use case requires them.

MFMediaID_SetOffset

This is an advanced method that lets developers override the offset value (in seconds) that is normally specified in the config file or license key supplied by Audible Magic. It is intended to let developers fine-tune the location in the input files or sample buffers where the signatures will be generated. An error will be returned by this function if the offset value is negative.

C

```
MFError MFMediaID_SetOffset(double dNewOffset);
```

Java on Android

```
public int MFMediaIDJNI.SetOffset(double dNewOffset);
```

Remarks

MFListenToSamples

This is an advanced method that lets developers send audio samples directly to the continuous audio fingerprinter, bypassing the internal Audible Magic audio capture routines. It can only be used if it is turned on by your specific license key (or configuration file). This method is used in combination with the "Functions for Identifying Real-Time Media."

Note that *all* `MFMediaID` objects share the same common low-level fingerprinter (except for ones whose configurations don't allow identification of "real-time media," because these can only identify "stored media" and don't use a continuous fingerprinter).

To use this function, create an `MFMediaID` object using a license key (or configuration file) that enables the `MFListenToSamples` API call. Then call `MFMediaID_StartListening`. From this point onward, any call to `MFListenToSamples` will send a buffer of samples into the low-level finger-printing code. The audio samples must have the following characteristics for successful identification:

- 16-bit, signed, little endian, linear PCM samples,
- Stereo or mono (2 or 1 channels)
- At least 22050 sample rate
- The `MFSampleFormat` value should be set to 0 (zero).

The size of the buffers (represented by `inNumSamples`) will determine the latency of the system. `MFListenToSamples` will maintain a circular buffer of samples that are passed to it. Any `MFMediaID` object that attempts an ID will need to obtain a long enough media buffer to satisfy the Duration specified in its license key.

C

```
MFError MFListenToSamples(
      const void* inputSamples,
      unsigned long inNumSamples,
      float inSampleRate,
      int inNumChannels,
      MFSampleFormat inSampleFormat);
```

Java

```
public mfListenToSamples(samples, numFrames, sampleRate, numChannels,
sampleFormat);
```

Java on Android

The `MFMediaIDJNI` Java class uses a private version of `MFListenToSamples` to communicate with the Java AudioRecorder class. You should not try to use `MFListenToSamples` directly on this platform without specific needs to do so.

Remarks

Just to state this one more time, *all* `MFMediaID` objects share the same low-level fingerprinting process. Your entire application can only monitor *one* audio stream using `MFListenToSamples`. If you need to monitor multiple simultaneous streams, you will need to implement your application in multiple processes — not threads, processes.

A typical application will be gathering PCM samples from some source and passing them to the library on a frequent, near real-time basis. A typical call to `MFListenToSamples` will have an `inputSamples` buffer of less than a hundred samples.

`inNumberSamples` is calculated as the number of channels times the number of frames, where a frame will have PCM data for all channels (1 or 2). So `inNumberSamples` is *dependent* on the number of channels in the input data stream. For example, a stereo "frame" may contain 2 samples. If the application buffer has 100 stereo frames of information, then `inNumSamples` should be 200. If `inNumSamples` is not a multiple of `inNumChannels`, the error `MF_NONINTEGRAL_NUMBER_OF_FRAMES` is returned. (For example, a stereo buffer must contain an even number of samples.)

While it may be tempting to use this interface to identify a media file, the library will not work as you might expect. (Instead, use `MFMediaID_IdentifyFile`.) `MFListenToSamples` is intended to receive a real-time feed of audio samples. If your application needs to simulate a real-time audio stream from a media file (for example to effect a demonstration), then your application should read from your media file and use a timer to pass the PCM samples to `MFListenToSamples` in a simulation of a real-time stream. See the listenToSamples example C program in the SDK release.

If your application needs to pass a single buffer of samples on a one-shot basis, rather than sequential buffers on a streaming basis, and cannot use `MFMediaID_IdentifyFile`, see `MFMediaID_IdentifySamples`. Note, however, that the buffer passed to `MFMediaID_IdentifySamples` must contain all the audio necessary to get an identification. Unlike `MFListenToSamples`, `MFMediaID_IdentifySamples` does not let the library "remember" a previously passed buffer; in other words, an identification cannot span successive buffers.

See Also: `MFMediaID_IdentifySamples`, `MFMediaID_IdentifyFile`

MFMediaID_GenerateRequestFromSamplesWithTimestamp

This is similar to the `MFMediaID_GenerateRequestFromSamples` call with the addition of the `timestamp` argument. This is only useful in a "live content" capture environment and must be used in conjunction with special server settings that will help the searches be more efficient. Please contact support if you have questions about how to use this feature.

The `timestamp` value is a double precision floating point value that represents the time at which the samples were capture. A value of zero means that we are not using this feature. A value of `MF_TIME_CURRENT` means that the samples will be timestamped with the current time as gathered from the computer running the program. The current time is defined as the Unix Epoch Time (the number of seconds that has elapsed since Thursday, 1 January, 1970, Coordinated Universal Time).

C

```
MFError MFMediaID_GenerateRequestFromSamplesWithTimestamp(
            MFMediaID mediaID,
            const void* samples,
            int numFrames,
            float sampleRate,
            int numChannels,
            MFSampleFormat format,
            MFConstString strAssetID,
            double timestamp,
            MFMediaIDRequest* request);
```

Java on Android

```
public int MF_16BIT_SIGNED_LINEAR_PCM     = 0;
/* use MF_16BIT_SIGNED_LINEAR_PCM for sampleFormat, below */
public int MF_CURRENT_TIME                = -1;
/* use MF_CURRENT_TIME to indicate the Unix Time in seconds */

long MFMediaIDJNI.GenerateRequestFromSamplesWithTimestamp(
      long mediaID,
      byte samples[],
      int numFrames,
      float sampleRate,
      int numChannels,
```

```
int sampleFormat,
String strAssetID,
double timestamp)
```

Throws java Exception on error.

Remarks

This function is useful when an application needs to take "live content" sample of audio (timestamped with the current time), but submit the ID request at a later time.

Please refer to the documentation on `MFMediaID_GenerateRequestFromSamples` for a description of the other arguments.

See Also: `MFMediaID_GenerateRequestFromSamples, MFMediaID_IdentifySamples, MFMediaID_PostRequest, MFMediaIDRequest_Destroy`

MFMediaID_IdentifySamplesWithTimestamp

This is similar to the `MFMediaID_IdentifySamples` call with the addition of the `timestamp` argument. This is only useful in a "live content" capture environment and must be used in conjunction with special server settings that will help the searches be more efficient. Please contact support if you have questions about how to use this feature.

The `timestamp` value is a double precision floating point value that represents the time at which the samples were capture. A value of zero means that we are not using this feature. A value of `MF_TIME_CURRENT` means that the samples will be timestamped with the current time as gathered from the computer running the program. The current time is defined as the Unix Epoch Time (the number of seconds that has elapsed since Thursday, 1 January, 1970, Coordinated Universal Time).

C

```
MFError MFMediaID_IdentifySamples(
            MFMediaID mediaID,
            const void* samples,
            int numFrames,
            float sampleRate,
            int numChannels,
            MFSampleFormat format,
            MFConstString strAssetID,
            double timestamp,
            MFMediaIDResponse* response);
```

Java on Android

```
public int MF_16BIT_SIGNED_LINEAR_PCM     = 0;
public int MF_CURRENT_TIME = -1;
/* use MF_16BIT_SIGNED_LINEAR_PCM for sampleFormat, below */

public long MFMediaIDJNI.IdentifySamplesWithTimestamp(
            long mediaID,
            byte samples[],
            int numFrames,
            float sampleRate,
            int numChannels,
            int sampleFormat,
```

```
double timestamp,
String strAssetID);
```

Throws java Exception on error.

Remarks

Please refer to the documentation on `MFMediaID_IdentifySamples` for a description of the other arguments.

MFMediaID_FindSignatureMatches

This is an advanced method that lets developers find the areas of similarity between two fingerprints. You must create (and later destroy) an MFMediaID object for use with this function. The parameters `pathToReferenceSignature` and `pathToUnknownSignature` are the paths to the two signature files. For many applications, the "reference" and "unknown" designations are arbitrary, but internally the lookups are done from the unknown signature against the reference signature, and the response XML will refer to times in the unknown and the reference. The accuracy of the returned information will depend on the type of the fingerprint (i5, i20, etc.) and on the lookup settings controlled by the license key (or configuration file).

When the XML string is extracted from the response object, it will have the form of this example:

```
<?xml version="1.0" encoding="utf-8"?>
<SignatureMatches>
   <Version>1</Version>
   <MatchInfo>
      <MatchTimeInReference>62.34</MatchTimeInReference>
      <MatchTimeInUnknown>234.34</MatchTimeInUnknown>
      <MatchDurationInReference>42.5</MatchDurationInReference>
      <MatchDurationInUnknown>42.5</MatchDurationInUnknown>
   </MatchInfo>
   ... MatchInfo repeats if applicable ...
</SignatureMatches>
```

C

```
MFError MFMediaID_FindSignatureMatches(
      MFMediaID mediaID,
      MFFilePath pathToReferenceSignature,
      MFFilePath pathToUnknownSignature,
      MFMediaIDResponse* response);
```

Java on Android

N/A

See Also: `MFMediaIDResponse_GetAsString`

MFMediaID_SetLicenseInfo

This is an advanced method that lets developers specify licensor and license string pairs and to include these strings as additional fields in subsequent ID requests, either for the life of the `MFMediaID` objector until the `MFMediaID_ClearLicenseInfo` API has been called on that object. This function can be called multiple times on a given `MFMediaID` object. In that case all of the specified licensor and license string pairs will be included in subsequent ID requests.

C

```
MFError MFMediaID_SetLicenseInfo(
      MFMediaID mediaID,
      MFConstString strLicensor,
      MFConstString strLicenseString);
```

Java on Android

```
N/A
```

MFMediaID_ClearLicenseInfo

This is an advanced method that lets developers clear all licensor and license string pairs that were previously added to the specified `MFMediaID` object via the `MFMediaID_SetLicenseInfo` function.

C

```
MFError MFMediaID_ClearLicenseInfo(
      MFMediaID mediaID);
```

Java on Android

```
N/A
```

Reference Fingerprinter API

The Reference Fingerprinter API is an advanced set of calls used by content providers to create the high-resolution fingerprints that are used by our ID servers. These reference fingerprints must be submitted to Audible Magic in order to be added to our online databases. This particular API is designed for users who will be sending streams of audio samples (from an opened media file, for example). The end result is a binary fingerprint file (sometimes referred to as a "signature") that can be submitted to Audible Magic.

The general programming flow is described in the pseudo-code below. The individual API calls, along with any special notes, are described in the following section.

1) Create the ReferenceFingerprinter object using a license key supplied by Audible Magic. The license key must explicitly allow the use of these API calls. (Alternatively, you might call MFReferenceFingerprinter_CreateUsingConfigFile if you are given a local config file.)

```
MFReferenceFingerprinter rf = NULL;
err = MFReferenceFingerprinter_CreateUsingLicenseKey(&rf, licenseKey);
```

2) Sit in a loop, reading in buffers of audio samples and sending them to the ReferenceFingerprinter until you are done with the stream or file. We currently only support mono or stereo buffers, 16-bit PCM samples, at a minimum of 22050 samples per second.

```
err = MFReferenceFingerprinter_AddSamples(rf, (void*)buffer,
numSamples, sampleRate, numChannels, MF_16BIT_SIGNED_LINEAR_PCM);
```

3) If you want to know the AMItemID string that has been assigned to this reference fingerprint (for example, to use in the future file name), you can use calls similar to these:

```
char* strAMItemID = NULL;
int   itemIDLen = 0;
err = MFReferenceFingerprinter_GetAMItemIDMaxLen(rf, &itemIDLen);
strAMItemID = (char*)malloc(itemIDLen);
err = MFReferenceFingerprinter_GetAMItemID(rf, strAMItemID, itemIDLen);
```

4) When done adding all the sample buffers (see 2, above), you must write out the reference fingerprint to a local file. You may want to use the AMItemID string retrieved in Step 3 above as part of the `strOutputFilename`.

```
err = MFReferenceFingerprinter_WriteFingerprintToFile(rf,
strOutputFilename);
```

You will receive a separate set of instructions for how to submit this output file to Audible Magic via our content ingestion process.

MFReferenceFingerprinter_CreateUsingLicenseKey

This function creates a ReferenceFingerprinter object based on a license key. The license key must have the reference fingerprinter features enabled. Once the `MFReferenceFingerprinter` object is successfully created, the caller can send audio samples to it.

C

```
MFError MFReferenceFingerprinter_CreateUsingLicenseKey(
            MFReferenceFingerprinter* rf,
            char* licenseKey);
```

Java on Android

```
N/A
```

Remarks:

To avoid memory leaks you must call `MFReferenceFingerprinter_Destroy` when done using this object.

MFReferenceFingerprinter_CreateUsingConfigFile

This function creates a ReferenceFingerprinter object based on a local config file. The config file must have the reference fingerprinter features enabled.

C

```
MFError MFReferenceFingerprinter_CreateUsingConfigFile(
            MFReferenceFingerprinter* rf,
            MFFilePath strPathToConfigFile
);
```

Java on Android

```
N/A
```

Remarks:

To avoid memory leaks you must call `MFReferenceFingerprinter_Destroy` when done using this object.

MFReferenceFingerprinter_Destroy

After you are done with with the `MFReferenceFingerprinter` object (usually after your call to `MFReferenceFingerprinter_WriteFingerprintToFile`), you must destroy the object to prevent memory leaks. Pass the address of the reference fingerprinter (as declared in the `MFReferenceFingerprinter_CreateUsingLicenseKey` call). This ensures that the pointer will be set to NULL upon completion of this call.

C

```
MFError MFReferenceFingerprinter_Destroy(
            MFReferenceFingerprinter* rf);
```

Java on Android

```
N/A
```

Remarks:

MFReferenceFingerprinter_GetAMItemIDMaxLen

This function returns the maximum length of the globally unique identifier (the AMItemID) that will be associated with the current reference fingerprinter object. Call this function to make sure you allocate enough room to call `MFReferenceFingerprinter_GetAMItemID`.

C

```
MFError MFReferenceFingerprinter_GetAMItemIDMaxLen(
            MFReferenceFingerprinter* rf,
            int* maxLen);
```

Java on Android

```
N/A
```

Remarks:

The `MFReferenceFingerprinter` object will generate the globally unique identifier based on the settings in your license key. See `MFReferenceFingerprinter_GetAMItemID`.

MFReferenceFingerprinter_GetAMItemID

This function returns the globally unique identifier (the AMItemID) that will be associated with the current reference fingerprinter object. The caller must allocate a string that is large enough to contain the AMItemID string or an error will be returned. Adding the `len` argument helps assure that the string returned will be of the proper length. Most AMItemID strings will fit in a 50 character string, but this convention may change in future versions.

C

```
MFError MFReferenceFingerprinter_GetAMItemID(
            MFReferenceFingerprinter* rf,
            char* strAMItemID,
            int len);
```

Java on Android

```
N/A
```

Remarks:

Since this is sometimes the ONLY identifier that will be returned by our online servers upon making a positive identification of an unknown signature, it is crucial that you save this string and associate it with the audio that was used to produce the reference fingerprint. One method would be to maintain your own local database that stored other metadata about the original audio along with this AMItemID string.

MFReferenceFingerprinter_AddSamples

Adding samples to the reference fingerprinter object requires knowing the format of the audio samples. The sample rate (samples per second), the number of channels (mono or stereo), and the sample format (16-bit, PCM linear) are required for each call.

The audio samples must have the following characteristics for successful identification:

- 16-bit, signed, little endian, linear PCM samples,
- Stereo or mono (2 or 1 channels)
- At least 22050 sample rate
- The `MFSampleFormat` value should be set to `MF_16BIT_SIGNED_LINEAR_PCM (zero).`

C

```
MFError MFReferenceFingerprinter_AddSamples(
            MFReferenceFingerprinter* rf,
            const void* samples,
            int numSamples,
            float sampleRate,
            int numChannels,
            MFSampleFormat sampleFormat);
```

Java on Android

```
N/A
```

Remarks:

When sending buffers of stereo samples make sure that the `numSamples` value represents the real number of individual samples involved. For example, 1 second of stereo samples at 44100 samples per second would contain 88200 samples, since each stereo "frame" (each time instant) consists of two 16-bit samples. For stereo buffers, `numSamples` must be an even number, or the error code `MF_NONINTEGRAL_NUMBER_OF_FRAMES` will be returned.

While there is no limit on the size of the buffers being sent to this routine, we suggest that the buffers contain between 0.5 and 2.0 seconds of audio.

MFReferenceFingerprinter_WriteFingerprintToFile

When you are done adding samples to the `MFReferenceFingerprinter`, you must save the resulting binary fingerprint to a file. You may call this function incrementally as you continue to add samples to the fingerprinter. Each time the function is called it writes out the complete set of of data for all the samples it has ever received.

C

```
MFError MFReferenceFingerprinter_WriteFingerprintToFile(
            MFReferenceFingerprinter* rf,
            MFFilePath strPathToSignature);
```

Java on Android

```
N/A
```

Remarks:

Do not use the same `MFReferenceFingerprinter` to fingerprint multiple input streams unless you want all of those streams to share the same AMItemID.

You will receive specific instructions from Audible Magic about how to submit this file for inclusion in our server databases.

Local Database Functions

These functions allow the application to query the library for the current state of a local database. Depending on the configuration, “local database” may refer to a reference cache. The functions are supported only for configurations that enable a local database or a reference cache, and will return errors in other configurations. (For more information about configurations, see the section “The License Key” in this document.)

MFMediaID_GetLocalDatabaseState

This function returns a newly created opaque object that is passed to the subsequent functions documented in this section. This object is a "snapshot" of the database. The subsequent calls report on the state in that snapshot. The actual database could change in the meantime, but this situation will not arise often, assuming these calls are made in rapid succession.

C

```
MFError MFMediaID_GetLocalDatabaseState(
                    MFMediaID mediaID,
                    MFLocalDatabaseState** localDBState);
```

Java on Android

N/A

MFLocalDatabaseState_GetStringLength

This function reports how many bytes should be in a string that the user will then allocate and pass to `MFLocalDatabaseState_GetAsString()`.

C

```
MFError MFLocalDatabaseState_GetStringLength(
                    MFLocalDatabaseState* localDBState,
                    int length);
```

Java on Android

N/A

MFLocalDatabaseState_GetAsString

Fills `strXML` with an XML string that has a format like the following:

```
<?xml version="1.0" encoding="utf-8"?>
<LocalDatabaseState>
    <Version>1</Version>
    <NumSignatures>1</NumSignatures>
    <Signature>
        <AMItemID>3493b3e6d5c04da9afd487f9386d797b_XYZ</AMItemID>
        <StartTime>1497.4000</StartTime>
        <Duration>180.0000</Duration>
    </Signature>
</LocalDatabaseState>
```

The `AMItemID` is the unique identifier that was associated with the reference signature when it was generated. (See `MFReferenceFingerprinter_GetAMItemID()`.) `StartTime` and `Duration` are expressed in seconds. `StartTime` is normally an offset from the beginning of the media file from which the signature was generated. In some cases, such as a signature generated from a live broadcast, it may instead be an epoch time (the number of seconds since the beginning of the year 1970, UTC/GMT).

The function's `length` parameter informs the function how many bytes have been allocated for `strXML`. This length includes a byte for the null terminator, and should be the value returned by `MFLocalDatabaseState_GetStringLength()`.

C

```
MFError MFLocalDatabaseState_GetAsString(
                    MFLocalDatabaseState* localDBState,
                    char* strXML,
                    int length);
```

Java on Android

N/A

Remarks

The application is responsible for using an XML parser to extract needed information from `strXML`. Because the XML structure might change in the future, the application code must be able to handle the presence of new fields and so on. For this reason, a standards-compliant XML parser should be used rather than a simple string parsing that assumes the structure in the above example.

MFLocalDatabaseState_Destroy

Tells the library to free the object that it allocated in `MFMediaID_GetLocalDatabaseState()`. The function also sets the application's variable (which is passed by reference) to NULL.

C

```
MFError MFLocalDatabaseState_Destroy(
                    MFLocalDatabaseState** localDBState);
```

Java on Android

N/A

Audio Input Objects

In general, one application will monitor one media stream. This is true for 99% of the applications that use the Audible Magic library. However, in some particular use cases a single application may be monitoring more than one stream. These applications are typically deployed on streaming media servers, in MSO head ends, and the like.

These applications can be implemented with multiple program instances. This implementation provides a measure of isolation – if one program instance crashes the others continue to run.

We provide these advanced calls for applications that need to monitor more than one stream in one program instance. When using these calls you will no longer use `MFListenToSamples`. Instead you will use `MFAudioInput_ListenToSamples`.

The general architecture is straightforward. For each media stream your application will create one MFAudioInput object. Your application will pass the PCM audio samples from each stream to the appropriate MFAudioInputObject using `MFAudioInput_ListenToSamples`.

Use `MFMediaID_SetAudioInput` to associate an MFAudioInput object with an MFMediaID object. An MFAudioInput object can be associated with many MFMediaID objects, but each MFMediaID object is associated with only one MFAudioInput object.

MFAudioInput_Create(MFAudioInput* audioInput);

This call creates an MFAudioInput object and returns it (by reference) to the caller.

MFAudioInput_Destroy(MFAudioInput* audioInput);

This call destroys an MFAudioInput object. Be very careful here: if this is called on an MFAudioInput object that is associate with a running MFMediaID object, your program can crash.

MFAudioInput_ListenToSamples(MFAudioInput audioInput, const void* inputSamples, unsigned long inNumSamples, float inSampleRate, int inNumChannels, MFSampleFormat inSampleFormat);

Sends sample data to the specified MFAudioInput object. See `MFListenToSamples` for a description of the call parameters.

MFMediaID_SetAudioInput(MFMediaID mediaID, MFAudioInput audioInput);

This call tells the specified MFMediaID object to operate on the audio stream in the specified audioInput object. Again, one MFAudioInput object can be associated with many MFMediaID

objects. All MFMediaID objects associated with an MFAudioInput object should be destroyed before that MFAudioInput object is destroyed.

Response XML

Every ID request will return an ID response in XML format. When a match has been identified, the appropriate XML elements will be populated with information about the match. The amount of metadata supplied will depend upon your service contract with Audible Magic.

The caller should use a proper XML / XSLT library to parse the ID response, as we will add fields to the XML to support new capabilities without notice. For example, libxml2 (http://xmlsoft.org/) and libxslt (http://xmlsoft.org/XSLT/) are both powerful open-source libraries released under the MIT license.

There are some error situations (network problems, server over capacity) where the server will not return properly formed XML. In general, in those instances, the sender should retry the ID after a short time.

When processing the Response XML, you should pay particular attention to the elements within the <Details> section.

As described in the API section above, `MFMediaIDResponse_GetIDStatus` should always be used to retrieve the response status. The returned information is usually enough for applications, but a more detailed status field can also be found inside the XML under the IdStatus key here:

`AMIdServerResponse/Details/IdResponseInfo/IdResponse/IdStatus`

An example XSLT to retrieve this data:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output omit-xml-declaration="yes"/>
       <xsl:template match="/">
              <xsl:value-of
select="AMIdServerResponse/Details/IdResponseInfo/IdResponse/IdStatus"/>
       </xsl:template>
</xsl:stylesheet>
```

The following table lists some values of the IdStatus key. The most frequent will be 2006 (identification was made successfully) and 2005 (search was completed successfully, but no match was found). For a complete list, see the error codes whose names begin with MF_ID_RESPONSE_STATUS in the “Error Codes” section of this document or in the file mfErrors_oem.h (which is included in the “include” directory of the release package).

Status	Name	Description
2004	ERROR	Generic server error.
2005	SONG_NOT_FOUND	The submitted fingerprint did not match any content in the database. (This applies to all kinds of content, not just “songs.”)
2006	MATCH	The submitted fingerprint matched content in the database.
2010	SERVER_BUSY	ID server is busy.
2012	OVER_CONCURRENCY_LIMIT	Your application has exceeded the limit for concurrency specified in your contract with AM. The application should stop sending transactions for some extended period of time.
2013	REJECTED	The server rejected the request.

2014	OVER_PER_DAY_LIMIT	Your application has exceeded the per-day limit for requests specified in your contract with AM.

The overall structure of the ID response XML is shown here for a response to a request sent to a remote ID server. For a response resulting from a lookup in a local database, see "Local Lookup or Submitted Content XML," below.

Remote Response XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AMIdServerResponse>
  <Status>
  </Status>
  <IdServerInfo>
  </IdServerInfo>
  <TxnGUID>7a568986-d123-4302-9b28-bd2411c4d6a5</TxnGUID>
  <Details>
    <IdResponseInfo>
      <IdResponse>
        <IdStatus>2006</IdStatus>
        <IdRequestGUID>10500393-1423-4404-92e2-b75eae43e9b5</IdRequestGUID>
        <IdDetails>
          <AMItemID>10362845_NKK</AMItemID>
          <SigOffset>8</SigOffset>
          <MatchOffset>8</MatchOffset>
          <MatchDuration>24</MatchDuration>
        </IdDetails>
      </IdResponse>
    </IdResponseInfo>
    <IdResponseInfo>
      <IdResponse>
        <IdStatus>2006</IdStatus>
        <IdRequestGUID>10500393-1423-4404-92e2-b75eae43e9b5</IdRequestGUID>
        <IdDetails>
          <AMItemID>10365692_JBS</AMItemID>
          <SigOffset>25</SigOffset>
          <MatchOffset>69</MatchOffset>
```

```
          <MatchDuration>36</MatchDuration>
        </IdDetails>
      </IdResponse>
    </IdResponseInfo>
  </Details>
</AMIdServerResponse>
```

Copyright Compliance Applications XML

For compliance applications, the Action element at the same level as IdStatus is important:

`AMIdServerResponse/Details/IdResponseInfo/IdResponse/Action`

returns the business usage rule associated with this match specific. The possible values returned are:

- Allow
  - The submitter of this reference fingerprint says you are allowed to use this content
- Block
  - The submitter of this reference fingerprint says you are not allowed to use this content
- Share
  - The submitted of this reference fingerprint says you are allowed to use this content under your own special arrangements with the content owner.

Any ID response XML may have more than one `IdResponse` section in it, each section corresponding to a match to a reference fingerprint. You should be aware that your transaction may result in matches to more that one reference, each reference having its own Action value. And note that these Action values may be in conflict. Audible Magic does not take responsibility for the veracity of the Action; Audible Magic simply passes on to you the Action values that have been associated with the reference fingerprints by the reference submitter for your particular application.

Following is a sample response XML returned from an identification request made against our song reference database. Note that the IdStatus is 2006, showing a successful ID, that the Action is Block. Some other fields of interest are the title:

`AMIdServerResponse/Details/IdResponseInfo/IdResponse/IdDetails/Title`

and the artist:

`AMIdServerResponse/Details/IdResponseInfo/IdResponse/IdDetails/Artist`

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AMIdServerResponse>
  <Status>
    <Number>2000</Number>
    <Description>OKAY</Description>
    <Source/>
    <ConcurrencyInfo>Standard</ConcurrencyInfo>
  </Status>
  <IdServerInfo>
    <IdServerGMT>2011/02/14 16:43:42</IdServerGMT>
    <IdServerDate>2011/02/14 08:43:42</IdServerDate>
    <IdServer>QATEST5</IdServer>
    <AppName>AMIDSvrLookup</AppName>
    <AppVersion>AM=4.1.43/MF=</AppVersion>
```



```
  <LookupContentType>CustomerResponse</LookupContentType>
</IdServerInfo>
<TxnType>IdRequest</TxnType>
<TxnGUID>7a568986-d123-4302-9b28-bd2411c4d6a5</TxnGUID>
<Details>
  <IdResponseInfo>
    <IdResponse>
      <IdResponseDate>2011/02/14 08:43:42</IdResponseDate>
      <IdResponseGMT>2011/02/14 16:43:42</IdResponseGMT>
      <IdStatus>2006</IdStatus>
      <IdRequestGUID>10500393-1423-4404-92e2-b75eae43e9b5</IdRequestGUID>
      <MediaGUID>10500393-1423-4404-92e2-b75eae43e9b5</MediaGUID>
      <IdServer>QATEST5</IdServer>
      <Action>Block</Action>
      <MatchType>Content</MatchType>
      <ReportingAction/>
      <IdDetails>
        <Type>Music</Type>
        <AMItemID>10362845_NKK</AMItemID>
        <OwnerItemID>USRE10700357</OwnerItemID>
        <AssetId>123</AssetId>
        <SigOffset>8</SigOffset>
        <MatchOffset>8</MatchOffset>
        <MatchDuration>24</MatchDuration>
        <Title>Always On My Mind</Title>
        <Description/>
        <LengthSeconds>270</LengthSeconds>
        <RunningTime>04:30</RunningTime>
        <ReleaseDate>01-May-2007</ReleaseDate>
        <Performer>Tom Jones</Performer>
        <Genre/>
        <Advisory/>
        <AdditionalInfo/>
        <Music>
          <Artist>Tom Jones</Artist>
          <AlbumCount>1</AlbumCount>
          <AlbumTitle>Call Me Irresponsible</AlbumTitle>
          <AlbumUPC>0093624999881</AlbumUPC>
          <AlbumDate>01-May-2007</AlbumDate>
          <Label>143/Reprise</Label>
          <Disc>1</Disc>
          <Track>11</Track>
          <Isrc>USRE10700357</Isrc>
          <CDLength/>
          <MediaType/>
          <ComposerWork/>
          <TrackComposer/>
          <TrackComposerWork/>
          <TrackConductor/>
          <TrackOrchestra/>
          <TrackPerformer/>
        </Music>
      </IdDetails>
      <SigOffset>8</SigOffset>
      <LookupScore/>
      <SongStartOffset/>
      <SongLength>270</SongLength>
      <SigGUID>805DAA6F-8A98-45D5-A392-32C94D001E93</SigGUID>
```



```
        <Song>Always On My Mind</Song>
        <AMItemID>10362845_NKK</AMItemID>
        <OwnerItemID>USRE10700357</OwnerItemID>
        <ContentType>Song</ContentType>
        <Artist>Tom Jones</Artist>
        <AlbumCount>1</AlbumCount>
        <AlbumTitle>Call Me Irresponsible</AlbumTitle>
        <AlbumUPC>0093624999881</AlbumUPC>
        <AlbumDate>01-May-2007</AlbumDate>
        <CopyrightDate>01-May-2007</CopyrightDate>
        <Genre/>
        <Label>143/Reprise</Label>
        <NbrDiscs>0</NbrDiscs>
        <NbrTracks>13</NbrTracks>
        <IdRetryCount>0</IdRetryCount>
        <MiscInfo/>
        <Disc>1</Disc>
        <Track>11</Track>
        <Isrc>USRE10700357</Isrc>
        <CDLength/>
        <MediaType/>
        <ComposerWork/>
        <TrackComposer/>
        <TrackComposerWork/>
        <TrackConductor/>
        <TrackOrchestra/>
        <TrackPerformer/>
        <CoverArtURL/>
        <PLine/>
        <RecordingOwner/>
        <Advisory/>
        <PurchaseDetails>
          <VendorName>MusicNet</VendorName>
          <VendorItemId/>
        </PurchaseDetails>
      </IdResponse>
    </IdResponseInfo>
    <AMKey>PgcUz27syaY2osjOSbctHzDZ4mENwLYG9SwLKKzZsYlv9IXlGCy8Pn/7WnB6hii-
JANoWO6Jd1xKK0hU9HHA2I5NqK/kn7a22Etac5KqAN8/aaBrxty536snE5X1BGITmjvJNpdj+6AAH
IrvRhSg9DI3baTV0eEU6XjP5utW9w4c=</AMKey>
  </Details>
  <UserInfo>
    <FullName>khoa</FullName>
    <Email>zzz@xxx.yyy</Email>
    <PostalCode>95032</PostalCode>
    <UserGUID>0b1a8198-ffa6-4bdd-a93e-1cc3751b84f4</UserGUID>
    <Language>English (United States)</Language>
    <GMTOffset>-480</GMTOffset>
  </UserInfo>
</AMIdServerResponse>
```

Notes:

The maximum length of each XML value in the response is 255, except for the AMItemID value, which is limited to 50 characters.

Following is a description of some more of the fields in the XML.

<AMIdServerResponse>
<Status>Status of the entire transaction
<Number>2000 – Transaction completed successfully</Number>
<Description>
OKAY – Response for Number 2000
Other response numbers will contain a description of the error here.
</Description>
<Source>If not Number 2000, then the source of the error will be here. </Source>
</Status>
<IdServerInfo>
<IdServerGMT>GMT time the ID was made</IdServerGMT>
<IdServerDate>Local time the ID was made</IdServerDate>
<IdServer>Name of the server that handled the transaction</IdServer>
</IdServerInfo>
<TxnType>IdRequest – This is the only type at this time</TxnType>
<Details>
<IdResponseInfo>
<IdResponse>
<IdResponseDate>Local time this response was generated</IdResponseDate>
<IdStatus>
2005 – ID was successful, but negative: no match was found
2006 – ID was successful, and a match (positive ID) was found
2009 – no media signature in the ID request
2010 – the ID server is currently busy
</IdStatus>
<Action>
If this tag has a value of "Block" it means the recommended action is for the application to prevent access to the file, generally because the copyright owner has not permitted free access. If the tag is missing but IdStatus is 2006, the recommended action is still "Block." If the tag is present and has any other value (such as "Allow"), it means don't block the file.
</Action>
<IdRequestGUID>
A unique GUID is made for each ID request and is returned here.
</IdRequestGUID>
<SigOffset>The number of seconds into the reference that the signature was found.</SigOffset>
<SongStartOffset>The offset to the start of the media, **relative to the SigOffset.**</SongStartOffset>
<SigGUID>A unique identifier used in debugging.</SigGUID>
<Song>Song Title</Song>
<AMItemID>Globally unique identifier assigned to this track by Audible Magic Corporation.</AMItemID>
<Artist>Song Artist</Artist>
<AlbumCount>Not used at this time</AlbumCount>
<AlbumTitle>Album Title</AlbumTitle>
<AlbumUPC>
Album UPC code, if available. Note that UPC codes are not guaranteed unique. Some albums do not have UPC codes. In these cases Audible Magic Corporation has assigned a code to the album. Audible Magic Corporation assigned album codes always have at least one non-numeric character in them.


```
                        </AlbumUPC>
                        <AlbumDate>Date the album was released, often reported as the copyright
                        date.</AlbumDate>
                        <IdRetryCount>Unused at this time.</IdRetryCount>
                        <MiscInfo>
                              May be unused. Any information in here may be logged for debugging
                              purposes.
                        </MiscInfo>
                  </IdResponse>
            </IdResponseInfo>
</Details>
<UserInfo>
            <FullName></FullName>
            <Email>xxxyyyzzz@audiblemagic.com  Where xxxyyyzzz is the strCustomerID as
            passed to one of the MFMediaID_Create... calls.</Email>
            <PostalCode>95032</PostalCode>
            <UserGUID>Returns what had been passed to one of the MFMediaID_Create... callsas the value
            of strUserGUID.</UserGUID>
            <Language>English</Language>
            <GMTOffset>-420</GMTOffset>
</UserInfo>
</AMIdServerResponse>
```

Local Lookup or Submitted Content XML

This is a sample of the XML that is returned when you do a local lookup with your own reference database or when you do remote identifications using a submitted content database. The response is essentially the same as the Copyright Compliance response, but metadata is not available and there is no Action.

SigOffset is seconds into the reference signature and MatchOffset is the corresponding match time based on the local device clock.

For synchronization applications please see the section in this document titled “Application Synchronization to Media.”

```
<?xml version='1.0' encoding='UTF-8'?>
<AMIdServerResponse>
  <Status>
  </Status>
  <Details>
    <IdResponseInfo>
      <IdResponse>
        <IdStatus>2006</IdStatus>
        <IdDetails>
          <AMItemID>theFileNameUsedWithAMDBBuilder.mpg</AMItemID>
          <OwnerItemID />
          <AssetId />
          <SigOffset>521.6263342</SigOffset>
          <MatchOffset>1346967061.508806</MatchOffset>
          <MatchDuration>5.9000000953674316</MatchDuration>
          <ReferenceDuration>137.5</ReferenceDuration>
        </IdDetails>
      </IdResponse>
    </IdResponseInfo>
  </Details>
</AMIdServerResponse>
```

For a local lookup, the response XML will be similar to the above, but will also contain a `LocalLookupInfo` field inside `AMIdServerResponse`, such as:

```
  <LocalLookupInfo>
    <DatabaseFilename>C:\myProject\myDBs\myMediaFiles.amdb</DatabaseFilename>
```

```
</LocalLookupInfo>
```

LiveTV Identification XML

A LiveTV identification adds a TvId section to the ID response XML that the application can use to map to an EPG data provider. It is also possible for Audible Magic to do the EPG mapping and provide you the metadata. If you plan to use your own EPG provider, please request the LiveTV Application Note from Audible Magic Support.

```
<?xml version='1.0' encoding='UTF-8'?>
<AMIdServerResponse>
  <Status>
  </Status>
  <TxnType>TvIdRequest</TxnType>
  <TxnGUID>9d037fe4-c460-4e1e-9817-2d95b0455220</TxnGUID>
  <Details>
    <IdResponseInfo>
      <IdResponse>
        <IdDetails>
          <SigOffset>1346433324.6263342</SigOffset>
          <MatchOffset>1346967061.508806</MatchOffset>
          <MatchDuration>5.9000000953674316</MatchDuration>
          <ReferenceDuration>120.0</ReferenceDuration>
          <TvId>
            <IdType>Program</IdType>
            <Channel>278</Channel>
            <MatchTime>2012-08-31T17:15:25Z</MatchTime>
            <Locale>en-US</Locale>
            <PostalCode>95113</PostalCode>
            <CountryCode>US</CountryCode>
            <ServiceProvider>DirecTV</ServiceProvider>
            <City>San Jose</City>
            <ProgramId>EP012838310004</ProgramId>
            <Title>Man, Woman, Wild</Title>
            <EpisodeTitle>Tasmania</EpisodeTitle>
            <Description>Mykel and Ruth battle hypothermia, dehydration and
       hunger on the coastline of Tasmania.</Description>
            <ShowDuration>3600</ShowDuration>
            <ShowOffset>925</ShowOffset>
          </TvId>
```

```
        </IdDetails>
      </IdResponse>
    </IdResponseInfo>
  </Details>
</AMIdServerResponse>
```

Contacting Customer Support

It is our goal to provide Audible Magic API users with helpful and timely customer support. We are available to answer any questions you may have and to provide technical assistance should you encounter any problems with the Audible Magic API.

Audible Magic Developer Portal

If you need help with the OEM API, the first place to look is the Audible Magic Developer Portal. This site will provide you with updates to our API, documentation, and FAQs. This site can be accessed at: developers.audiblemagic.com.

Level 1: General questions and non-urgent support

These are general questions and program issues that do not need to be resolved immediately.

Please submit requests for Level 1 support in one of two ways:

1. Via our web form at http://www.audiblemagic.com/support/contact.asp
2. Via email at: support@audiblemagic.com

Please be sure to include as much information as possible so that we can provide timely answers to your questions. If relevant, provide the version number of the AM SDK, the platform on which the problem occurs, whether you are using the C or Java API, and the function producing the error. For Level One support questions, we will:

- Handle requests during regular business hours. Monday-Friday 9:00AM– 5:00PM Pacific Time
- Confirm your support request via email
- Respond within 8 business hours.

Level 2: Urgent support request

For questions and support issues that need immediate attention, we are available by phone at +1(408)399-6405, extension 504. After dialing the main number and getting the the voice-mail greeting, enter 504 to connect to the support department.

When we receive a request for urgent support, we will make every effort to respond to you as quickly as possible, within 1 business hour.

During the trial test license period, we do not offer 24-hour support. We do offer 7x24 support for our production customers. If, during your trial period, your business requires round-the-clock support, please contact us for a quote.

Legal Notices

General

As used below, “the Audible Magic Library” or “the library” refers to Audible Magic’s OEM SDK, and more specifically to the library having "mfcbr" in its name: mfcbr_oem.dll on Windows, libmfcbroem.so or libmfcbroem.a on Linux, libmfcbroem.dylib or libmfcbroem.a. on Mac OS X, libmfcbroem_ios.a on iOS, libmfcbr.so on Android.

The Audible Magic Library incorporates (or dynamically links against) some third-party software whose license agreements may require or encourage Audible Magic to include the following notices in this documentation. Audible Magic does not offer legal advice, but we believe some of these licenses may require or encourage users of our SDK to do the same.

As a general statement, Audible Magic strives to comply with all licensing requirements of every library we use. Our desktop / server SDKs may include and dynamically link to libraries covered by the GNU Lesser General Public License (LGPL) v2.1; we do not statically link to these libraries. Our desktop / server SDKs may contain an FFmpeg executable file. If so, this is distributed under the GNU General Public License (GPL) and our library may make use of it only in a way that maintains the proprietary nature of our code and our customers' code so that no release of source code is required. Our mobile SDKs do not include GPL or LGPL code.

- MPEG Layer-3 audio decoding technology licensed from Fraunhofer IIS and Thomson.

Customers of the Audible Magic Library are responsible for verifying that they have licenses to decode any audio or video formats that they will be fingerprinting through use of this library. The library permits decoding of many formats, including the following, which to the best of our knowledge (this does not constitute legal advice) do not require the customer to obtain separate licenses: MPEG audio, PCM audio (e.g., in WAV files), and (on Windows) Windows Media. Other codecs may require licenses. However, many customers may already have licenses for various codecs, whether the licenses are obtained separately or bundled with software such as an operating system. If you have questions or are having difficulty with a particular format, please contact Audible Magic.

An Inclusive List of Third-Party Licenses

The section below lists all licenses for any third-party software used in any Audible Magic SDK. Not all these licenses will apply to every SDK. Each SDK includes a text file whose name begins "3rdPartyLicenses". You should review this file in the SDK(s) you use to make sure your application complies with all the license requirements.

Section Name: Audible Magic Patents

The Audible Magic SDK software may be covered by one or more of the following patents.

US Patents

5,918,223; 6,834,308; 6,968,337; 7,500,007 B2; 7,529,659; 7,562,012; 7,877,438; 7,917,645; 8,006,314; 8,082,150; 8,086,445; 8,112,818; 8,130,746; 8,199,651; 8,332,326; 8,645,279; 8,972,481

European Patents

EP1354276 B1; EP1485815 B1; EP1771791 A2; EP1449103 B1; GB2464049 B

Other patents pending.

CopySense is a registered trademark of Audible Magic Corporation.

. . . . . .

Section Name: media decoder patents

MPEG Layer-3 audio decoding technology licensed from Fraunhofer IIS and Thomson.

Customers of the Audible Magic SDK are responsible for verifying that they have licenses to decode any audio or video formats that they will be fingerprinting through use of this library. The library permits decoding of many formats, including the following, which to the best of our knowledge (this does not constitute legal advice) do not require the customer to obtain separate licenses: MPEG audio, PCM audio (e.g., in WAV files), and (on Windows) Windows Media. Other codecs may require licenses. However, many customers may already have licenses for various codecs, whether the licenses are obtained separately or bundled with software such as an operating system. If you have questions or are having difficulty with a particular format, please contact Audible Magic.

. . . . . .

Section Name: boost

http://www.boost.org/LICENSE_1_0.txt

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and

all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

. . . . . .

Section Name: log4j

Apache license v2.0, listed later in this file.

. . . . . .

Section Name: log4cplus

Copyright (C) 1999-2009 Contributors to log4cplus project.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

. . . . . .

Section Name: utf8

// Copyright 2006 Nemanja Trifunovic

/*

Permission is hereby granted, free of charge, to any person or organization
obtaining a copy of the software and accompanying documentation covered by
this license (the "Software") to use, reproduce, display, distribute,
execute, and transmit the Software, and to prepare derivative works of the
Software, and to permit third-parties to whom the Software is furnished to
do so, all subject to the following:

The copyright notices in the Software and this entire statement, including
the above license grant, this restriction and the following disclaimer,
must be included in all copies of the Software, in whole or in part, and
all derivative works of the Software, unless such copies or derivative
works are solely in the form of machine-executable object code generated by
a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT
SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE
FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.

. . . . . .

Section Name: gecko

*** Version: MPL 1.1/GPL 2.0/LGPL 2.1**
*****
*** The contents of this file are subject to the Mozilla Public License Version**
*** 1.1 (the "License"); you may not use this file except in compliance with**
*** the License. You may obtain a copy of the License at**
*** http://www.mozilla.org/MPL/**
*****
*** Software distributed under the License is distributed on an "AS IS" basis,**
*** WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License**
*** for the specific language governing rights and limitations under the**
*** License.**
*****
*** The Original Code is mozilla.org code.**

*****

*** The Initial Developer of the Original Code is**

*** Netscape Communications Corporation.**

*** Portions created by the Initial Developer are Copyright (C) 1998**

*** the Initial Developer. All Rights Reserved.**

*****

*** Contributor(s):**

*****

*** Alternatively, the contents of this file may be used under the terms of**

*** either the GNU General Public License Version 2 or later (the "GPL"), or**

*** the GNU Lesser General Public License Version 2.1 or later (the "LGPL"),**

*** in which case the provisions of the GPL or the LGPL are applicable instead**

*** of those above. If you wish to allow use of your version of this file only**

*** under the terms of either the GPL or the LGPL, and not to allow others to**

*** use your version of this file under the terms of the MPL, indicate your**

*** decision by deleting the provisions above and replace them with the notice**

*** and other provisions required by the GPL or the LGPL. If you do not delete**

*** the provisions above, a recipient may use your version of this file under**

*** the terms of any one of the MPL, the GPL or the LGPL.**

*****

. . . . . .

Section Name: libiconv

http://www.gnu.org/software/libiconv/

LGPL v2.1, listed later in this file.

. . . . . .

Section Name: libcharset

http://www.gnu.org/software/libiconv/

LGPL v2.1, listed later in this file.

. . . . . .

Section Name: id3lib

http://id3lib.sourceforge.net

LGPL v2.1, listed later in this file.

. . . . . .

Section Name: unrar

******** ***** ****** UnRAR - free utility for RAR archives**

****   ** **   ** **   ** ~~~~~~**

********   ******* ******   License for use and distribution of**

****   ** **   ** **   **   ~~~~~~**

****   ** **   ** **   **   FREE portable version**

~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The source code of UnRAR utility is freeware. This means:

1. All copyrights to RAR and the utility UnRAR are exclusively owned by the author - Alexander Roshal.

2. The UnRAR sources may be used in any software to handle RAR archives without limitations free of charge, but cannot be used to re-create the RAR compression algorithm, which is proprietary. Distribution of modified UnRAR sources in separate form or as a part of other software is permitted, provided that it is clearly stated in the documentation and source comments that the code may not be used to develop a RAR (WinRAR) compatible archiver.

3. The UnRAR utility may be freely distributed. It is allowed to distribute UnRAR inside of other software packages.

4. THE RAR ARCHIVER AND THE UnRAR UTILITY ARE DISTRIBUTED "AS IS". NO WARRANTY OF ANY KIND IS EXPRESSED OR IMPLIED. YOU USE AT YOUR OWN RISK. THE AUTHOR WILL NOT BE LIABLE FOR DATA LOSS, DAMAGES, LOSS OF PROFITS OR ANY OTHER KIND OF LOSS WHILE USING OR MISUSING THIS SOFTWARE.

5. Installing and using the UnRAR utility signifies acceptance of these terms and conditions of the license.

6. If you don't agree with terms of the license you must remove UnRAR files from your storage devices and cease to use the utility.

Thank you for your interest in RAR and UnRAR.

Alexander L. Roshal

. . . . .

Section Name: libunzip from Info-ZIP

Copyright (c) 1990-2004 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Christian Spieler, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. **Redistributions of source code must retain the above copyright notice, definition, disclaimer, and this list of conditions.**

2. **Redistributions in binary form (compiled executables) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.**

3. **Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, and dynamic, shared, or static library versions--must be plainly marked as such and must not be misrepresented as being the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but not limited to, labeling of the altered versions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or of the Info-ZIP URL(s).**

4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

. . . . . .

Section Name: portaudio

```
/*
 * PortAudio Portable Real-Time Audio Library
 * PortAudio API Header File
 * Latest version available at: http://www.portaudio.com
 *
 * Copyright (c) 1999-2006 Ross Bencina and Phil Burk
 *
 * Permission is hereby granted, free of charge, to any person obtaining
 * a copy of this software and associated documentation files
 * (the "Software"), to deal in the Software without restriction,
 * including without limitation the rights to use, copy, modify, merge,
 * publish, distribute, sublicense, and/or sell copies of the Software,
 * and to permit persons to whom the Software is furnished to do so,
 * subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be
 * included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
 * EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
 * MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
 * IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR
 * ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
 * WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

/*
 * The text above constitutes the entire PortAudio license; however,
 * the PortAudio community also makes the following non-binding requests:
 *
 * Any person wishing to distribute modifications to the Software is
 * requested to send the modifications to the original developer so that
 * they can be incorporated into the canonical version. It is also
 * requested that these non-binding requests be included along with the
```

*** license above.**

***/**

. . . . . .

Section Name: expat XML parser

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd
and Clark Cooper
Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

. . . . . .

Section Name: Libcurl
http://curl.haxx.se/docs/copyright.html
COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2012, Daniel Stenberg, <daniel@haxx.se>.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

. . . . . .

Section Name: libeay32

http://www.openssl.org/source/license.html

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License

. . . . . .

```
/* ====================================================================
 * Copyright (c) 1998-2011 The OpenSSL Project.  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 *    software must display the following acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
```

*****

*** 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to**

*** endorse or promote products derived from this software without**

*** prior written permission. For written permission, please contact**

*** openssl-core@openssl.org.**

*****

*** 5. Products derived from this software may not be called "OpenSSL"**

*** nor may "OpenSSL" appear in their names without prior written**

*** permission of the OpenSSL Project.**

*****

*** 6. Redistributions of any form whatsoever must retain the following**

*** acknowledgment:**

*** "This product includes software developed by the OpenSSL Project**

*** for use in the OpenSSL Toolkit (http://www.openssl.org/)"**

*****

*** THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY**

*** EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE**

*** IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR**

*** PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR**

*** ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,**

*** SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT**

*** NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;**

*** LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)**

*** HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,**

*** STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)**

*** ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED**

*** OF THE POSSIBILITY OF SUCH DAMAGE.**

*** ==**

*****

*** This product includes cryptographic software written by Eric Young**

*** (eay@cryptsoft.com). This product includes software written by Tim**

*** Hudson (tjh@cryptsoft.com).**

*****

***/**

Original SSLeay License

. . . . . .

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

*** All rights reserved.**

*****

*** This package is an SSL implementation written**

*** by Eric Young (eay@cryptsoft.com).**

*** The implementation was written so as to conform with Netscapes SSL.**

```
 *
 * The licence and distribution terms for any publically available version or
 * derivative of this code cannot be changed.  i.e. this code cannot simply be
 * copied and put under another distribution licence
 * [including the GNU Public Licence.]
 */
. . . . . .
```

Section Name: libssleay32

http://www.openssl.org/source/license.html

http://www.openssl.org/source/license.html

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of
the OpenSSL License and the original SSLeay license apply to the toolkit.
See below for the actual license texts. Actually both licenses are BSD-style
Open Source licenses. In case of any license issues related to OpenSSL
please contact openssl-core@openssl.org.

OpenSSL License

. . . . . .

```
/* ====================================================================
 * Copyright (c) 1998-2011 The OpenSSL Project.  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 *    software must display the following acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
 *
 * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
 *    endorse or promote products derived from this software without
 *    prior written permission. For written permission, please contact
```

*** openssl-core@openssl.org.**

*****

*** 5. Products derived from this software may not be called "OpenSSL"**

*** nor may "OpenSSL" appear in their names without prior written**

*** permission of the OpenSSL Project.**

*****

*** 6. Redistributions of any form whatsoever must retain the following**

*** acknowledgment:**

*** "This product includes software developed by the OpenSSL Project**

*** for use in the OpenSSL Toolkit (http://www.openssl.org/)"**

*****

*** THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY**

*** EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE**

*** IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR**

*** PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR**

*** ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,**

*** SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT**

*** NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;**

*** LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)**

*** HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,**

*** STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)**

*** ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED**

*** OF THE POSSIBILITY OF SUCH DAMAGE.**

*** ==**

*****

*** This product includes cryptographic software written by Eric Young**

*** (eay@cryptsoft.com). This product includes software written by Tim**

*** Hudson (tjh@cryptsoft.com).**

*****

***/**

Original SSLeay License

. . . . . .

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

*** All rights reserved.**

*****

*** This package is an SSL implementation written**

*** by Eric Young (eay@cryptsoft.com).**

*** The implementation was written so as to conform with Netscapes SSL.**

*****

*** This library is free for commercial and non-commercial use as long as**

*** the following conditions are aheared to. The following conditions**

*** apply to all code found in this distribution, be it the RC4, RSA,**

*** [including the GNU Public Licence.]**

*/

. . . . . .

Section Name: Libst

/* libst.h - include file for portable sound tools library

******

**** Copyright (C) 1989 by Jef Poskanzer.**

******

**** Permission to use, copy, modify, and distribute this software and its**

**** documentation for any purpose and without fee is hereby granted, provided**

**** that the above copyright notice appear in all copies and that both that**

**** copyright notice and this permission notice appear in supporting**

**** documentation. This software is provided "as is" without express or**

**** implied warranty.**

*/

. . . . . .

Section Name: MD5 Implementation

http://userpages.umbc.edu/~mabzug1/cs/md5/md5.html

Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

. . . . . .

Section Name: LibSSL

http://www.openssl.org/

The OpenSSL toolkit is licensed under an Apache-style license, which basically means that you are free to get and use it for commercial and non-commercial purposes subject to some simple license conditions.

. . . . . .

Section Name: Libcrypto

http://www.openssl.org/

We compile with no-idea no-mdc2 no-rc5

The OpenSSL toolkit is licensed under an Apache-style license, which basically means that you are free to get and use it for commercial and non-commercial purposes subject to some simple license conditions.

. . . . . .

Section Name: libgcrypt

From the source code

 # Copyright (c) 2010, Pierre-Olivier Latour

 # All rights reserved.

#

 # Redistribution and use in source and binary forms, with or without

 # modification, are permitted provided that the following conditions are met:

 # * Redistributions of source code must retain the above copyright

 # notice, this list of conditions and the following disclaimer.

 # * Redistributions in binary form must reproduce the above copyright

 # notice, this list of conditions and the following disclaimer in the

 # documentation and/or other materials provided with the distribution.

 # * The name of Pierre-Olivier Latour may not be used to endorse or

 # promote products derived from this software without specific prior

 # written permission.

. . . . . .

Section Name: libgpg-error

http://lists.gnupg.org/pipermail/gnupg-devel/2007-December/024093.html

LGPL v2.1, listed later in this document.

. . . . . .

Section Name: Libssh2

http://www.libssh2.org/license.html

/* Copyright (c) 2004-2007 Sara Golemon <sarag@libssh2.org>

*** Copyright (c) 2005,2006 Mikhail Gusarov <dottedmag@dottedmag.net>**

*** Copyright (c) 2006-2007 The Written Word, Inc.**

*** Copyright (c) 2007 Eli Fant <elifantu@mail.ru>**

*** Copyright (c) 2009 Daniel Stenberg**

*** Copyright (C) 2008, 2009 Simon Josefsson**

*** All rights reserved.**

*****

*** Redistribution and use in source and binary forms,**

*** with or without modification, are permitted provided**

*** that the following conditions are met:**

*****

*** Redistributions of source code must retain the above**

*** copyright notice, this list of conditions and the**

*** following disclaimer.**

*****

*** Redistributions in binary form must reproduce the above**

*** copyright notice, this list of conditions and the following**

*** disclaimer in the documentation and/or other materials**

*** provided with the distribution.**

*****

*** Neither the name of the copyright holder nor the names**

*** of any other contributors may be used to endorse or**

*** promote products derived from this software without**

*** specific prior written permission.**

*****

*** THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND**

*** CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,**

*** INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES**

*** OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE**

*** ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR**

*** CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,**

*** SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,**

*** BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR**

*** SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS**

*** INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,**

*** WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING**

*** NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE**

*** USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY**

*** OF SUCH DAMAGE.**

***/**

. . . . . .

Section Name: Libz

http://www.zlib.net/zlib_license.html

/* zlib.h ** interface of the 'zlib' general purpose compression library

version 1.2.7, May 2nd, 2012

Copyright (C) 1995-2012 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied

warranty. In no event will the authors be held liable for any damages

arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
claim that you wrote the original software. If you use this software
in a product, an acknowledgment in the product documentation would be
appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be
misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly Mark Adler
jloup@gzip.org madler@alumni.caltech.edu

*/

. . . . . .

Section Name: direntWin32
Copyright Kevlin Henney, 1997, 2003. All rights reserved.
Permission to use, copy, modify, and distribute this software and its documentation for any purpose is hereby granted without fee,
provided that this copyright and permissions notice appear in all copies and derivatives.
This software is supplied "as is" without express or implied warranty.
But that said, if there are any problems please get in touch.

. . . . . .

Section Name: FFMPEG
http://ffmpeg.org/legal.html
Our distribution may contain an FFmpeg executable file. If so, this is distributed
under the GPL and our library may make use of it only in a way that maintains the
proprietary nature of our code and our customer's code.
As of MFCBR version 26 (2015), our library does not link any FFmpeg libraries.

Section Name: libroxml
http://www.libroxml.net
version 2.2.3

Since 2013-07-15 an exception to LGPL v2.1 for static linking was added. This should makes it easier for people wanting to integrate the library into iOS / android packages. Read the License.txt file for more informations.

https://code.google.com/p/libroxml/wiki/RoxmlLicense

As a special exception, the copyright holders of this library give you permission to statically link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. It is also appreciated if you mention in the README or CREDITS the use of this library. An independent module is a module which is not derived from or based on this library. If you modify this library, you may extend this exception to your version of the library, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

LGPL v2.1, listed later in this document.

. . . . . .

Section Name: pthreadGC2

http://sourceware.org/pthreads-win32/copying.html

This file is covered under the following Copyright:

Copyright (C) 2001,2006 Ross P. Johnson
All rights reserved.

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Pthreads-win32 is covered by the GNU Lesser General Public License

----

Pthreads-win32 is open software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public License
as published by the Free Software Foundation version 2.1 of the
License.

Pthreads-win32 is several binary link libraries, several modules,
associated interface definition files and scripts used to control
its compilation and installation.

Pthreads-win32 is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU Lesser General Public License for more details.

A copy of the GNU Lesser General Public License is distributed with
pthreads-win32 under the filename:

COPYING.LIB

You should have received a copy of the version 2.1 GNU Lesser General
Public License with pthreads-win32; if not, write to:

Free Software Foundation, Inc.
59 Temple Place
Suite 330
Boston, MA 02111-1307
USA

The contact addresses for pthreads-win32 is as follows:

Web: http://sources.redhat.com/pthreads-win32
Email: Ross Johnson
Please use: Firstname.Lastname@homemail.com.au

Pthreads-win32 copyrights and exception files

With the exception of the files listed below, Pthreads-win32
is covered under the following GNU Lesser General Public License
Copyrights:

Pthreads-win32 - POSIX Threads Library for Win32
Copyright(C) 1998 John E. Bossom
Copyright(C) 1999,2006 Pthreads-win32 contributors

The current list of contributors is contained
in the file CONTRIBUTORS included with the source
code distribution. The current list of CONTRIBUTORS
can also be seen at the following WWW location:
http://sources.redhat.com/pthreads-win32/contributors.html

Contact Email: Ross Johnson
Please use: Firstname.Lastname@homemail.com.au

These files are not covered under one of the Copyrights listed above:

COPYING
COPYING.LIB
tests/rwlock7.c

This file, COPYING, is distributed under the Copyright found at the
top of this file. It is important to note that you may distribute
verbatim copies of this file but you may not modify this file.

The file COPYING.LIB, which contains a copy of the version 2.1 GNU Lesser General Public License, is itself copyrighted by the Free Software Foundation, Inc. Please note that the Free Software Foundation, Inc. does NOT have a copyright over Pthreads-win32, only the COPYING.LIB that is supplied with pthreads-win32.

The file tests/rwlock7.c is derived from code written by Dave Butenhof for his book 'Programming With POSIX(R) Threads'. The original code was obtained by free download from his website http://home.earthlink.net/~anneart/family/Threads/source.html and did not contain a copyright or author notice. It is assumed to be freely distributable.

In all cases one may use and distribute these exception files freely. And because one may freely distribute the LGPL covered files, the entire pthreads-win32 source may be freely used and distributed.

. . . . . .

LGPL v2.1

http://www.gnu.org/licenses/lgpl-2.1.html

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages - typically libraries - of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or

can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will

operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

. . . . . .

Apache License v2.0

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,

and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent

to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier

identification within third-party archives.

Copyright 1999-2005 The Apache Software Foundation

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

. . . . . .