

Making Atomic Warfare Fun Again



CFX_AudioInfoMX

v2.3 October 29th 2003

OPEN SOURCE. FREeware

(includes full **Java** source code)

Java CFX for **Macromedia** ColdFusionMX and up
by Lewis A. Sellers (aka Min)

<http://www.intrafoundation.com>

products@intrafoundation.com

Yes, I'm probably for hire. Need something? I may or may not have the time, but you can ask.

READ THE VERSION HISTORY

[documentation](#) | [version](#) | [file](#) | [folder](#) | [filter](#) | [bad](#) | [extended folder](#) | [uploadtest](#) | [unknownfile](#) -- [stresstest](#) | [concurrencytest](#)

F O R E W O R D

This Java language ColdFusion extension tag (that is, a CFX) for **Macromedia** ColdFusionMX, **CFX_AudioInfoMX**, will scan either a single file or an entire folder, looking for audio files of mp3, wav, au and midi format. From these files it will gather up all manner of information such as song title, author, compression type, bitrate, song length, etc. Handles ID3v1, ID3v1.1 and ID3v2 4.0. Works with all Java versions of ColdFusionMX.

The source code to this program is included as open source. It has been tested under Windows only at the moment. If you have compiled and used it on other platforms please feel free to drop a note.

COPYRIGHT & USAGE RIGHTS

This program and all documentation is open source free software; You can redistribute and/or modify it as you like so long as you provide documentation as to it's original source and author or this archive entire.

There is no warranty as to it's operation. It is your responsibility to test thoroughly in development environments before moving to production machines. Further, you should have the full and complete source code which is bundled with this compiled software, with which to make any corrections or enhancements yourself. Please do as others have and send in your useful changes so they may be added to the common code base.

SECURITY ON SHARED SYSTEMS

Is this tag safe to use on a shared system, that is for web hosting or a shared co-located machine? Most Likely. It only reads a file or the files in a folder and returns specific information on known file formats.

If you have concerns about it's security – well, you do have the complete source code. If you're paranoid, read through it and recompile a version yourself.

SOURCE CODE

The complete source code is included. It was compiled under Windows 2000 SP4 using Java SDK 1.4 and Netbeans 3.5.1. The version of ColdFusion it was tested against was Redsky ColdFusionMX 6.1.

INSTALLATION

1. Copy CFX_AudioInfoMX.jar to your ColdFusion classes folder. This is usually C:\CFusionMX\cfx\java\ if you're using ColdFusionMX 6 under Windows.
2. Go into the ColdFusion administrator, Extensions, CFX Tags. Register the Java CFX as:

Tag Name: CFX_AudioInfoMX

Class Name: CFX_AudioInfoMX

3. If you haven't already, go into the ColdFusion administrator, "Java and JVM" section and add to the Class Path the following:
c:\CFusionMX\cfx\java,C:\CFusionMX\lib\cfx.jar,
(assuming this is where your version of ColdFusion keeps these files by default.)
4. Reboot. (Or restart ColdFusion server through the services.)

USING THE TAG

```
<CFX_AudioInfoMX FOLDER="c:\mymusic">
```

```
<CFX_AudioInfoMX FILE="c:\mymusic\mycatssusyandsissyoying.mp3">
```

The tag has only two input arguments: FILE or FOLDER. If you wish to scan a folder for audio files you would use the FOLDER parameter. If you wish to query a specific file for information you would use the FILE parameter.

Either way, the information will be returned in a SQL Query named "AudioInfo". The query may either 0, 1 or many rows depending on the argument you use and how many audio files exist.

The returned fields are as follows:

- File
- Error
- Type
- Encoding
- Extra
- Frequency
- BitRate
- Channels

- Tracks
- Seconds
- FileSize
- Copyright
- Original
- Title
- Artist
- Album
- Year
- Genre
- Lyrics
- URL
- Country
- Language
- Comments
- Source

Query Field Elements: Detailed

File

The full file path, including folders and filename.

Error

Errors, if any, relating to the file.

A detailed error message. If it's blank then no error occurred and the other fields can probably be trusted. Use a simple string length function, ie `LEN(Error)`, in your AudioInfoMX loop to determine if you can trust the data or not.

As of 2.1, formal error codes you can use `FIND`, etc against have been added. The following error codes are now added to the end of each "sentence" of text describing the error. The text may change from version to version or between image formats, but the following hard error codes will not.

For example, a simple `Find(' [CORRUPT] ', #Error#)` will allow you to tell if the file was corrupt and display your own custom error message or take your own actions more readily.

The codes are defined thusly:

[GRAMMAR]

You used the wrong parameters when calling the tag, or misspelled them.

[UNSUPPORTEDFORMAT]

The file you're trying to get information on is not currently one of those that the tag understands how to get information on. Trying to get info on an AVI would be an example of an unsupported image format.

[UNSUPPORTEDSUBTYPE]

This means that yes, we do know the image format (for example TIFF), but this particular file is using some rare or new internal format that the tag doesn't know how to completely handle.

[CORRUPT]

The file structure doesn't seem to conform to anything we know about how it should be laid out. This could mean the file is corrupt. It could mean it's not actually the

format you think it is (ie, it's a JPG but it's been saved with the BMP extension.) It could also mean it's a radically unsupported image type which has the tag saying "I have no idea how to decode this information".

[ENDOFFILE]

Premature end of file. You'll get this when the tag tries to read a block of information (such as an image header) and unexpectedly the file ends before it should. When you receive this error you almost always will get a [CORRUPT] error immediately following it for the obvious reason that the image can not be completely and correctly decoded.

[FILEDOESNOTEXIST]

Either the file does not exist or the folder as specified does not.

[SECURITY]

The file probably does exist, but file permission settings are not allowing use to read any of it's data.

Type

What is the format of audio file? MP3, WAV, AU, MIDI, etc.

Encoding

A long bit of text describing what algorithms were used to compress and encode the audio of the file.

Frequency

The sampling frequency in hz.

BitRate

This is typically a number such as 96 or 128. It CAN however be either the text "variable" or "forbidden". "variable" is encountered where a variable bitrate is being used. "forbidden" should never be encountered unless the file is corrupt.

Channels

The number of audio channels being output. Almost always either 1 or 2. 1 is mono, 2 is stereo, etc.

Tracks

A track is essentially a sequential piece of music from a single instrument or groups of instruments.

Seconds

Length of song or performance.

FileSize

Size of file in bytes.

Copyright

0=No, 1=Yes (boolean).

Original

0=No, 1=Yes (boolean).

Title

Title of the song or performance piece, if applicable.

Artist

Name of the artist, if applicable.

Album

Name of the source album/cd, if applicable.

Year

The year the music was recorded, if applicable.

Genre

	The family of music (if any) the audio file belongs to. Uses MPEG3 definitions and WinAMP extended definitions. If genre is an unknown extension returns "[Unknown-(n)]" where n is unknown the genre number. "[Undefined]" is returned for genre number 255.
Lyrics	Rarely are text lyrics encountered in an audio file except in midi and id3v2 mp3 formats.
URL	Usually the homepage URL of the group that made the song.
Country	Country of origin.
Language	Primary spoken language of the audio file.
Comments	Collected comments, if any, embedded in the file.
Source	Source of the audio. Where it physically came from.
InfoType	Type of Info being used. This is essentially used to report what ID3 version is being found for MP3. Ie, ID3v1, ID3v1.1, ID3v2.x, etc.
Extra	Extra misc info.

Returned Variables: Detailed

The following variables are always returned.

AudiInfoDescription	Text description of the tag and copyright notices.
AudiInfoVersion	The numbered version of the tag.
AudiInfoFile	Echo of FILE parameter (only with FILE).
AudiInfoFolder	Echo of FOLDER parameter (only with FOLDER).
AudiInfoFilter	Echo of FILTER parameter (only with FILTER).
AudiInfoError	A detailed error message. If blank no error occurred.

Version History

- **2.3**, October 29th 2004.
Cleaned up documentation, examples and code. Verified will work with Apache 2.
- **2.2**, November 18th 2003.
Fixed track bug in id3v2. Misc cleanup in code/docs. Added lyrics support.

Todo: Add support for RIFF MP3, WMA (ASF), OGG, MP4, M4A, M4P, AAC, RA and AIFF. Still much work could be done to add full support for all formats.

- **2.1c**, November 13th 2003.

Fixed a divide-by-zero bug that could occur with the ID3 test mp3's which would cause looping to a folder to abort prematurely.

Also note that yes, BitRateMin and BitRateMax do exist as returned fields but you shouldn't use them right now. Why? Cause to actually find out the min/max you have to loop through `_all_` the mp3 frame structures, and that is very slow. At some point in time I'll see about making it as a selectable OPTION that can be toggled on though.

Changed Copyright and Original to isCopyright and isOriginal respectively. Added Tracks, Composer, Picture and Copyright fields.

- **2.1**, November 12th 2003.

Cleaned up code a bit and added better exception handling.

- **2.0**, November 8th 2003.

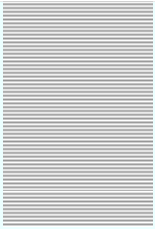
Yes, this is a brand new Java version of the old C++ CFX tag. I'd been job-hunting for some steady, long-term work for a change and thought I should get a few of those pesky certificates, ie the SCJP and SCJD. In that spirit converted a few older C++ apps over to Java to brush up on the language before taking the certification exams. Thus is born the Java version of the ancient CFX_AudioInfo.

All future update to this tag will happen in the Java version. Mainly because when making this tag tonight I noticed several pesky bugs in the old C++ version I don't feel like going back and fixing now.

I'll get around to properly documenting and cleaning up the source around v2.1 after it's been tested in the wild a while.

Note: Looking for a *Programmer Analyst III* in the **Knoxville/Oak Ridge, TN** area? Email me (at webmaster@intrafoundation.com). Long-term, stable and interesting work is preferred, but as always I'm up for short-term contract jobs as well.

- **1.0 July 20th 2003.** Fixed to work under Java CFMX. Also repaired a few problems I noticed including an incorrect null-pointer with the "file" parameter.
- **0.4a Dec 18th 2001.** Slight repackaging to make work with CF5 during a server move. Recomplied. Didn't feel like dealing with the hassle of redoing the version graphics, so... Pretend I did. This tag is about to become obsolete soon anyway. Good tag. Sit now. Roll over.
- **0.4 May 13 2000, Midnight.** mp3 header decoding was extremely screwed up in the last version. Fixed. Sorry about that. Added mp3 ID3 v1.1 support. Added support for the ridiculously complex ID3v2 format. Added misc other refinements and fields. There are a few other additions that can be made to help populate the fields, but it is close to a finished state. Send feedback.
(0.4 brought to you by the fucking uncensored version of Korn: Issues (esp. track 6), not the "edited" version I accidentally bought first. Damn all censors to the 7st level of hell.)
- **0.3alpha May 12 2000.**

- 
- **0.2alpha** *May 11 2000*. Actually worked on the tag for the first time in months. First release. The tag is "alpha" quality, thus may or may not work. No one else but me has even seen the tag yet, let alone tested it.
 - **0.1alpha** *April 26 2000*.
 - **0.0alpha** *March 2 2000*. This tag is a refit of CFX_ImageInfo 1.4.