mutagenwrapper Documentation

Release 0.0.5

Choongmin Lee

January 29, 2015

Contents

mutagenwrapper is a wrapper for mutagen that normalizes tags between various audio file formats. The normalization is done as in foobar2000, which supports seamless tag management of various audio file formats.

Note that this module is still in its early development stage. Many features are not supported like changing an album art. See Mappings for more details. There may be undocumented limitations too.

You must backup your files if you are going to edit tags and make changes with this module. There is no way to recover files without a backup if something goes wrong.

mutagenwrapper can read tags with multiple values (e.g. multiple artists), but due to the incompatibility of mutagen and foobar2000, the way it saves tags makes them unreadable by foobar2000 in some cases, especially for MP4 files. Hidden tags, if exist, may be lost when you make another change and save tags in foobar2000.

CHAPTER 1

Examples

You can access tags via human-readable names. For the mappings between these names and the actual, internal tag names, see Mappings:

```
>>> from mutagenwrapper import read_tags
>>> mp3 = read_tags('test.mp3')
>>> mp3['artist']
[u'Holst, Gustav (1874-1934)']
>>> mp3['album']
[u'The Planets']
>>> mp4 = read_tags('test.m4a')
>>> mp4['artist']
[u'Holst, Gustav (1874-1934)']
>>> mp4['album']
[u'The Planets']
```

You can also get album arts in binary:

```
>>> mp3['pictures']
['\xff\xd8\xff\xe0\x00\x10JFIF\x00\x01\x01\x01\x01,\x01,...']
```

Some tags return a single value, not a list. You can use `find()` to always get a single value even for those tags that returns a list. It will throw a `KeyError` if there is more than one value. `find()` returns *None* for non-existing names, but you can set a default value:

```
>>> mp4['compilation']
False
>>> mp4.find('compilation')
False
>>> mp4.find('artist')
u'Holst, Gustav (1874-1934)'
>>> mp4.find('date')
>>> mp4.find('date', '1916')
'1916'
```

Whenever you find the wrapper is not enough, you can access the internal mutagen object:

```
>>> mp3.raw_tags['TALB']
TALB(encoding=3, text=[u'The Planets'])
>>> mp4.raw_tags['\xa9alb']
[u'The Planets']
```

Mappings

Tag names in FLAC files are not mapped and the same name is used as the internal name, except for `pictures`, for which there is no corresponding Vorbis comment (they are stored elsewhere, not as a comment). A blank cell means there is no mapping and values may be hidden by the wrapper (but they are still there and doesn't get lost when saving) or custom tags can be used.

Custom tags (starting with `TXXX:` for ID3 and `----:com.apple.iTunes:` for MP4) have their name as the internal name without prefix and lowercased, e.g. both `TXXX:CUSTOM` and `----:com.apple.iTunes:CUSTOM` are mapped to `custom`.

In the current version, `pictures`, `tracknumber`, `tracktotal`, `discnumber`, and `disctotal` are read-only. You can read and write other tags.

Name	ID3v2	iTunes MP4
album	TALB	©alb
albumartist	TPE2	aART
albumartistsortorder	TSO2	soaa
albumsortorder	TSOA	soal
artist	TPE1	©ART
artistsortorder	TSOP	soar
comment		©cmt
composer	TCOM	©wrt
composersortorder	TSOC	soco
conductor	TPE3	
copyright	TCOP	cprt
date	TDRC	©day
discnumber	TPOS	disk
disctotal	TPOS	disk
encodedby	TENC	©too
genre	TCON	©gen
grouping	TIT1	©grp
lyrics	TEXT	©lyr
pictures	APIC:	covr
title	TIT2	©nam
titlesortorder	TSOT	sonm
tracknumber	TRCK	trkn
tracktotal	TRCK	trkn

CHAPTER 3

API reference

mutagenwrapper.**read_tags**(*name*, *raw=False*, *format=None*)

Returns tags in the audio file as a `TagsWrapper`, a dictionary-like object with mapped tag keys and values. If *raw* is *True*, returns a raw mutagen object with non-mapped keys and values.

Currently it supports FLAC, MP3, iTunes MP4 (.m4a) files. File format is determined by extension, but you can specify it by *format* (possible values: 'flac', 'mp3', 'm4a').

class mutagenwrapper.**TagsWrapper**(*filename*)

A dictionary-like object that wraps the raw mutagen tags. You can access tags via mapped tag keys, like 'artist' or 'album'.

Note that the implementation is rather inefficient. It iterates the internal keys and convert them everytime `iterkeys()` is called. Since many methods rely on that method, it could be a problem especially when there are many inclusion tests using `__contains__()` (e.g. `'artist' in mywrapper`)

filename

Path of the file

find(*key*, *default=None*)

Returns a single value associated with a given key. Raises `KeyError` if the key is not found or there is no or more than one value.

pprint(*raw=False*)

Print formatted representation of tags. if *raw* is *True*, use keys and values from the underlying mutagen object.

reload()

Reload tags from the file.

save()

Save changes to the file.

CHAPTER 4

Indices and tables

- *genindex*
- *modindex*
- *search*

Python Module Index

m

Index