eyeD3 Documentation

Release 0.9

Travis Shirk

Jan 30, 2020

CONTENTS

1 Status **1**

2 About **3**

3 Features **5**

4 Get Started **7**

4.1 Installation 7
4.2 Documentation 8
4.3 ChangeLog 92
4.4 References 92
4.5 Indices and tables 93

Python Module Index **95**

Index **97**

CHAPTER

ONE

STATUS

CHAPTER

TWO

ABOUT

eyeD3 is a Python tool for working with audio files, specifically MP3 files containing ID3 metadata (i.e. song info).

It provides a command-line tool (`eyeD3`) and a Python library (`import eyed3`) that can be used to write your own applications or plugins that are callable from the command-line tool.

For example, to set some song information in an mp3 file called `song.mp3`:

```
$ eyeD3 -a Integrity -A "Humanity Is The Devil" -t "Hollow" -n 2 song.mp3
```

With this command we've set the artist (`-a/--artist`), album (`-A/--album`), title (`-t/--title`), and track number (`-n/--track-num`) properties in the ID3 tag of the file. This is the standard interface that eyeD3 has always had in the past, therefore it is also the default plugin when no other is specified.

The results of this command can be seen by running the `eyeD3` with no options.

```
$ eyeD3 song.mp3
song.mp3      [ 3.06 MB ]
--------------------------------------------------------------------------------
ID3 v2.4:
title: Hollow
artist: Integrity
album: Humanity Is The Devil
album artist: None
track: 2
--------------------------------------------------------------------------------
```

The same can be accomplished using Python.

```
import eyed3

audiofile = eyed3.load("song.mp3")
audiofile.tag.artist = u"Integrity"
audiofile.tag.album = u"Humanity Is The Devil"
audiofile.tag.album_artist = u"Integrity"
audiofile.tag.title = u"Hollow"
audiofile.tag.track_num = 2

audiofile.tag.save()
```

eyeD3 is written and maintained by Travis Shirk and is licensed under version 3 of the GPL.

CHAPTER

THREE

FEATURES

- Python package for writing application and/or plugins.
- Command-line tool driver script that supports plugins. viewer/editor interface.
- Easy editing/viewing of audio metadata from the command-line, using the ‘classic’ plugin.
- Support for ID3 versions 1.x, 2.2 (read-only), 2.3, and 2.4.
- Support for the MP3 audio format exposing details such as play time, bit rate, sampling frequency, etc.
- Abstract design allowing future support for different audio formats and metadata containers.

CHAPTER

FOUR

GET STARTED

Python >= 3.6 is required.

For installation instructions or more complete documentation see http://eyeD3.nicfit.net/

Please post feedback and/or defects on the issue tracker, or mailing list.

4.1 Installation

Stable releases of eyeD3 are best installed via `pip` or `easy_install`; or you may download TGZ or ZIP source archives from a couple of official locations. Detailed instructions and links may be found on the *Installation* page.

Otherwise, if you want to live on the edge, you can pull down the source code from the Git repository at GitHub. The *Installation* page has details for how to access the source code.

4.1.1 Installation

Easy Installation

Install using 'pip'

pip is a tool for installing Python packages from Python Package Index and is a replacement for *easy_install*. It will install the package using the first 'python' in your path so it is especially useful when used along with virtualenv, otherwise root access may be required.

```
$ pip install eyeD3
# Optional: To install the ultra powerful Display plugin (-P display)
$ pip install eyeD3[display-plugin]
```

Note that on Windows, you also need to install the libmagic binaries.

```
$ pip install python-magic-bin
```

Dependencies

eyeD3 0.9 has been tested with Python 2.7, >=3.3 (see the 0.7.x series for Python 2.6 support).

The primary interface for building and installing is Setuptools. For example, `python setup.py install`.

Development Dependencies

If you are interested in doing development work on eyeD3 (or even just running the test suite), you may also need to install some additional packages:

$ pip install -r requirements/test.txt $ pip install -r requirements/dev.txt

Download Source Archive

Source packages are available from the release archive in tar.gz and zip formats. After un-archiving the distribution file you can install in the common manner:

```
$ tar xzf eyeD3-X.Y.Z.tar.gz
$ cd eyeD3-X.Y.Z
# This may require root access
$ python setup.py install
```

Or you can run from the archive directory directly:

```
$ tar xzf eyeD3-X.Y.Z.tar.gz
$ cd eyeD3-X.Y.Z
$ python setup.py build
$ export PYTHONPATH=`pwd`/build/lib
$ export PATH=${PATH}:`pwd`/bin
```

Checking Out the Source Code

```
$ git clone https://github.com/nicfit/eyeD3.git
```

Note: When submitting patches please base them on the 'master' branch.

4.2 Documentation

4.2.1 'eyeD3' Command Line Tool

The `eyeD3` command line interface is based on plugins. The main driver knows how to traverse file systems and load audio files for hand-off to the plugin to do something interesting. With no plugin selected a simplified usage is:

```
$ eyeD3 --help
usage: eyeD3 [-h] [--version] [--exclude PATTERN]
             [--plugins] [--plugin NAME]
             [PATH [PATH ...]]
```

(continues on next page)

(continued from previous page)

```
positional arguments:
  PATH                  Files or directory paths

optional arguments:
  -h, --help            show this help message and exit
  --version             Display version information and exit
  --exclude PATTERN     A regular expression for path exclusion. May be
                        specified multiple times.
  --plugins             List all available plugins
  --plugin NAME         Specify which plugin to use.
```

The `PATH` argument(s) along with optional usage of `--exclude` are used to tell `eyeD3` what files or directories to process. Directories are searched recursively and every file encountered is passed to the plugin until no more files are found.

To list the available plugins use the `--plugins` option and to select a plugin pass its name using `--plugin=<name>`.

If no `--plugin=` option is provided the *default* plugin is selected. Currently this is set to be the command line tag viewer/editor that has been the primary interface in all versions of eyeD3 prior to 0.7.x.

Plugins

classic - Tag Viewer/Editor

Classic eyeD3 interface for viewing and editing tags.

Names

classic

Description

All PATH arguments are parsed and displayed. Directory paths are searched recursively. Any editing options (–artist, –title) are applied to each file read.

All date options (-Y, –release-year excepted) follow ISO 8601 format. This is `yyyy-mm-ddThh:mm:ss`. The year is required, and each component thereafter is optional. For example, 2012-03 is valid, 2012–12 is not.

Options

```
  -a STRING, --artist STRING
                        Set the artist name.
  -A STRING, --album STRING
                        Set the album name.
  -b STRING, --album-artist STRING
                        Set the album artist name. 'Various Artists', for
                        example. Another example is collaborations when the
                        track artist might be 'Eminem featuring Proof' the
                        album artist would be 'Eminem'.
  -t STRING, --title STRING
```

(continues on next page)

(continued from previous page)

```
                        Set the track title.
  -n NUM, --track NUM   Set the track number. Use 0 to clear.
  -N NUM, --track-total NUM
                        Set total number of tracks. Use 0 to clear.
  --track-offset N      Increment/decrement the track number by [-]N. This
                        option is applied after --track=N is set.
  -d NUM, --disc-num NUM
                        Set the disc number. Use 0 to clear.
  -D NUM, --disc-total NUM
                        Set total number of discs in set. Use 0 to clear.
  -G GENRE, --genre GENRE
                        Set the genre. If the argument is a standard ID3 genre
                        name or number both will be set. Otherwise, any string
                        can be used. Run 'eyeD3 --plugin=genres' for a list of
                        standard ID3 genre names/ids.
  --non-std-genres      Disables certain ID3 genre standards, such as the
                        mapping of numeric value to genre names.
  -Y YEAR, --release-year YEAR
                        Set the year the track was released. Use the date
                        options for more precise values or dates other than
                        release.
  -c STRING, --comment STRING
                        Set a comment. In ID3 tags this is the comment with an
                        empty description. See --add-comment to add multiple
                        comment frames.
  --rename PATTERN      Rename file (the extension is not affected) based on
                        data in the tag using substitution variables: $album,
                        $album_artist, $artist, $best_date,
                        $best_date:prefer_recording,
                        $best_date:prefer_recording:year,
                        $best_date:prefer_release,
                        $best_date:prefer_release:year, $best_date:year,
                        $disc:num, $disc:total, $file, $file:ext,
                        $original_release_date, $original_release_date:year,
                        $recording_date, $recording_date:year, $release_date,
                        $release_date:year, $title, $track:num, $track:total

ID3 options:
  -1, --v1              Only read and write ID3 v1.x tags. By default, v1.x
                        tags are only read or written if there is not a v2 tag
                        in the file.
  -2, --v2              Only read/write ID3 v2.x tags. This is the default
                        unless the file only contains a v1 tag.
  --to-v1.1             Convert the file's tag to ID3 v1.1 (Or 1.0 if there is
                        no track number)
  --to-v2.3             Convert the file's tag to ID3 v2.3
  --to-v2.4             Convert the file's tag to ID3 v2.4
  --release-date DATE   Set the date the track/album was released
  --orig-release-date DATE
                        Set the original date the track/album was released
  --recording-date DATE
                        Set the date the track/album was recorded
  --encoding-date DATE  Set the date the file was encoded
  --tagging-date DATE   Set the date the file was tagged
  --publisher STRING    Set the publisher/label name
  --play-count <+>N     Set the number of times played counter. If the
                        argument value begins with '+' the tag's play count is
```

(continues on next page)

(continued from previous page)

```
                      incremented by N, otherwise the value is set to
                      exactly N.
  --bpm N             Set the beats per minute value.
  --unique-file-id OWNER_ID:ID
                      Add a unique file ID frame. If the ID arg is empty the
                      frame is removed. An OWNER_ID is required. The ID may
                      be no more than 64 bytes.
  --add-comment COMMENT[:DESCRIPTION[:LANG]]
                      Add or replace a comment. There may be more than one
                      comment in a tag, as long as the DESCRIPTION and LANG
                      values are unique. The default DESCRIPTION is '' and
                      the default language code is 'eng'.
  --remove-comment DESCRIPTION[:LANG]
                      Remove comment matching DESCRIPTION and LANG. The
                      default language code is 'eng'.
  --remove-all-comments
                      Remove all comments from the tag.
  --add-lyrics LYRICS_FILE[:DESCRIPTION[:LANG]]
                      Add or replace a lyrics. There may be more than one
                      set of lyrics in a tag, as long as the DESCRIPTION and
                      LANG values are unique. The default DESCRIPTION is ''
                      and the default language code is 'eng'.
  --remove-lyrics DESCRIPTION[:LANG]
                      Remove lyrics matching DESCRIPTION and LANG. The
                      default language code is 'eng'.
  --remove-all-lyrics   Remove all lyrics from the tag.
  --text-frame FID:TEXT
                      Set the value of a text frame. To remove the frame,
                      specify an empty value. For example, --text-
                      frame='TDRC:'
  --user-text-frame DESC:TEXT
                      Set the value of a user text frame (i.e., TXXX). To
                      remove the frame, specify an empty value. e.g.,
                      --user-text-frame='SomeDesc:'
  --url-frame FID:URL   Set the value of a URL frame. To remove the frame,
                      specify an empty value. e.g., --url-frame='WCOM:'
  --user-url-frame DESCRIPTION:URL
                      Set the value of a user URL frame (i.e., WXXX). To
                      remove the frame, specify an empty value. e.g.,
                      --user-url-frame='SomeDesc:'
  --add-image IMG_PATH:TYPE[:DESCRIPTION]
                      Add or replace an image. There may be more than one
                      image in a tag, as long as the DESCRIPTION values are
                      unique. The default DESCRIPTION is ''. If PATH begins
                      with 'http[s]://' then it is interpreted as a URL
                      instead of a file containing image data. The TYPE must
                      be one of the following: OTHER, ICON, OTHER_ICON,
                      FRONT_COVER, BACK_COVER, LEAFLET, MEDIA, LEAD_ARTIST,
                      ARTIST, CONDUCTOR, BAND, COMPOSER, LYRICIST,
                      RECORDING_LOCATION, DURING_RECORDING,
                      DURING_PERFORMANCE, VIDEO, BRIGHT_COLORED_FISH,
                      ILLUSTRATION, BAND_LOGO, PUBLISHER_LOGO.
  --remove-image DESCRIPTION
                      Remove image matching DESCRIPTION.
  --remove-all-images   Remove all images from the tag
  --write-images DIR    Causes all attached images (APIC frames) to be written
                      to the specified directory.
```

(continues on next page)

(continued from previous page)

```
  --add-object OBJ_PATH:MIME-TYPE[:DESCRIPTION[:FILENAME]]
                        Add or replace an object. There may be more than one
                        object in a tag, as long as the DESCRIPTION values are
                        unique. The default DESCRIPTION is ''.
  --remove-object DESCRIPTION
                        Remove object matching DESCRIPTION.
  --write-objects DIR   Causes all attached objects (GEOB frames) to be
                        written to the specified directory.
  --remove-all-objects  Remove all objects from the tag
  --add-popularity EMAIL:RATING[:PLAY_COUNT]
                        Adds a pupularity metric. There may be multiples
                        popularity values, but each must have a unique email
                        address component. The rating is a number between 0
                        (worst) and 255 (best). The play count is optional,
                        and defaults to 0, since there is already a dedicated
                        play count frame.
  --remove-popularity EMAIL
                        Removes the popularity frame with the specified email
                        key.
  --remove-v1           Remove ID3 v1.x tag.
  --remove-v2           Remove ID3 v2.x tag.
  --remove-all          Remove ID3 v1.x and v2.x tags.
  --remove-frame FID    Remove all frames with the given ID. This option may
                        be specified multiple times.
  --max-padding NUM_BYTES
                        Shrink file if tag padding (unused space) exceeds the
                        given number of bytes. (Useful e.g. after removal of
                        large cover art.) Default is 64 KiB, file will be
                        rewritten with default padding (1 KiB) or max padding,
                        whichever is smaller.
  --no-max-padding      Disable --max-padding altogether.
  --encoding latin1|utf8|utf16|utf16-be
                        Set the encoding that is used for all text frames.
                        This option is only applied if the tag is updated as
                        the result of an edit option (e.g. --artist, --title,
                        etc.) or --force-update is specified.

Misc options:
  --force-update        Rewrite the tag despite there being no edit options.
  -v, --verbose         Show all available tag data
  --preserve-file-times
                        When writing, do not update file modification times.
```

Examples

eyeD3 can do more than edit exiting tags, it can also create new tags from nothing. For these examples we'll make a dummy file to work with.

```
$ rm -f example.id3
$ touch example.id3
$ ls -o example.id3

-rw-r--r-- 1 travis 0 Feb 26 17:14 example.id3
```

Now let's set some common attributes like artist and title.

Most options have a shorter name that can be used to save typing. Let's add the album name (`-A`), the genre (`-G`), and the year (`-Y`) the record was released.

Notice how the genre displayed as "Hardcore (id 129)" in the above tag listing. This happens because the genre is a recognized value as defined by the ID3 v1 standard. eyeD3 used to be very strict about genres, but no longer. You can store any value you'd like. For a list of recognized genres and their respective IDs see the genres plugin.

By default writes ID3 v2.4 tags. This is the latest standard and supports UTF-8 which is a very nice thing. Some players are not caught up with the latest standards (iTunes, pfft) so it may be necessary to convert amongst the various versions. In some cases this can be a lossy operation if a certain data field is not supported, but eyeD3 does its best to convert when the data whenever possible.

```
# Convert the current v2.4 frame to v2.3
$ eyeD3 --to-v2.3 example.id3 -Q

/home/travis/devel/eyeD3/git/example.id3                          [ 0.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v2.4: 0 frames
Writing ID3 version v2.3
--------------------------------------------------------------------------------

# Convert back
$ eyeD3 --to-v2.4 example.id3 -Q

/home/travis/devel/eyeD3/git/example.id3                        [ 266.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v2.3: 0 frames
Writing ID3 version v2.4
--------------------------------------------------------------------------------

# Convert to v1, this will lose all the more advanced data members ID3 v2 offers
$ eyeD3 --to-v1.1 example.id3 -Q

/home/travis/devel/eyeD3/git/example.id3                        [ 266.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v2.4: 0 frames
Writing ID3 version v1.1
--------------------------------------------------------------------------------
```

The last conversion above converted to v1.1, or so the output says. The final listing shows that the tag is version 2.4. This is because tags can contain both versions at once and eyeD3 will always show/load v2 tags first. To select the version 1 tag use the `-1` option. Also note how the the non-standard genre was lost by the conversion, thankfully it is still in the v2 tag.

```
$ eyeD3 -1 example.id3

/home/travis/devel/eyeD3/git/example.id3                        [ 394.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v1.0:
title:
artist:
album:
album artist: None
track:                genre: Other (id 12)
--------------------------------------------------------------------------------
```

The `-1` and `-2` options also determine which tag will be edited, or even which tag will be converted when one of the conversion options is passed.

```
# Set an artist value in the ID3 v1 tag
$ eyeD3 -1 example.id3 -a id3v1

/home/travis/devel/eyeD3/git/example.id3                    [ 394.00 Bytes ]
--------------------------------------------------------------------------------
Setting artist: id3v1
ID3 v1.0:
title:
artist: id3v1
album:
album artist: None
track:                genre: Other (id 12)
Writing ID3 version v1.0
--------------------------------------------------------------------------------

# The file now has a v1 and v2 tag, change the v2 artist
$ eyeD3 -2 example.id3 -a id3v2

/home/travis/devel/eyeD3/git/example.id3                    [ 394.00 Bytes ]
--------------------------------------------------------------------------------
Setting artist: id3v2
ID3 v2.4:
title:
artist: id3v2
album:
album artist: None
track:
Writing ID3 version v2.4
--------------------------------------------------------------------------------

# Take all the values from v2.4 tag (the default) and set them in the v1 tag.
$ eyeD3 -2 --to-v1.1 example.id3

/home/travis/devel/eyeD3/git/example.id3                    [ 394.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v2.4:
title:
artist: id3v2
album:
album artist: None
track:
Writing ID3 version v1.1
--------------------------------------------------------------------------------

# Take all the values from v1 tag and convert to ID3 v2.3
$ eyeD3 -1 --to-v2.3 example.id3

/home/travis/devel/eyeD3/git/example.id3                    [ 394.00 Bytes ]
--------------------------------------------------------------------------------
ID3 v1.0:
title:
artist: id3v2
album:
album artist: None
track:                genre: Other (id 12)
Writing ID3 version v2.3
--------------------------------------------------------------------------------
```

At this point the tag is all messed up with by these experiments, you can always remove the tags to start again.

```
$ eyeD3 --remove-all example.id3

/home/travis/devel/eyeD3/git/example.id3                    [ 394.00 Bytes ]
--------------------------------------------------------------------------------
Removing ID3 v1.x and/or v2.x tag: SUCCESS
No ID3 v1.x/v2.x tag found!
```

Complex Options

Some of the command line options contain multiple pieces of information in a single value. Take for example the `--add-image` option:

```
--add-image IMG_PATH:TYPE[:DESCRIPTION]
```

This option has 3 pieced of information where one (DESCRIPTION) is optional (denoted by the square brackets). Each invidual value is seprated by a ':' like so:

```
$ eyeD3 --add-image cover.png:FRONT_COVER
```

This will load the image data from `cover.png` and store it in the tag with the type value for FRONT_COVER images. The list of valid image types are listed in the `--help` usage information which also states that the IMG_PATH value may be a URL so that the image data does not have to be stored in the the tag itself. Let's try that now.

```
$ eyeD3 --add-image http://example.com/cover.jpg:FRONT_COVER
eyeD3: error: argument --add-image: invalid ImageArg value: 'http://example.com/cover.
↪jpg:FRONT_COVER'
```

The problem is the `':'` character in the the URL, it confuses the format description of the option value. To solve this escape all delimeter characters in option values with `'\\'` (for linux and macOS), single `'\'` for Windows).

Linux/MacOS:

```
$ eyeD3 --add-image http\\://example.com/cover.jpg:FRONT_COVER example.id3

/home/travis/devel/eyeD3/git/example.id3                      [ 0.00 Bytes ]
--------------------------------------------------------------------------------
Adding image http://example.com/cover.jpg
ID3 v2.4:
title:
artist:
album:
album artist: None
track:
FRONT_COVER Image: [Type: -->] [URL: b'http://example.com/cover.jpg']
Description:

Writing ID3 version v2.4
--------------------------------------------------------------------------------
```

Windows:

```
$ eyeD3 --add-image http\://example.com/cover.jpg:FRONT_COVER example.id3

C:\Users\user\Downloads\example.id3  [ 0.00 Bytes ]
```

(continues on next page)

(continued from previous page)

```
--------------------------------------------------------------------------------
Adding image http://example.com/cover.jpg
ID3 v2.4:
title:
artist:
album:
album artist: None
track:
FRONT_COVER Image: [Type: -->] [URL: b'http://example.com/cover.jpg']
Description:

Writing ID3 version v2.4
--------------------------------------------------------------------------------
```

display - Display tag information by pattern

Prints specific tag information which are specified by a pattern.

Names

display

Description

Displays tag information for each file. With a pattern the concrete output can be specified.

The pattern EBNF:

```
pattern    :=  { <text> | tag | function }*
tag        :=  '%' <name> { ',' parameter }* '%'
function   :=  '$' <name> '(' [ parameter { ',' parameter }* ]  ')'
parameter  :=  [ <name> '=' ] [ pattern ]
<text>     :=  string with escaped special characters
<name>     :=  string without special characters
```

Tags are surrounded by two '%'. There are also functions that starts with a '$'. Both tag and function could be parametrized.

Options

```
--pattern-help        Detailed pattern help
-p STRING, --pattern STRING
                      Pattern string
-f FILE, --pattern-file FILE
                      Pattern file
--no-newline          Print no newline after each output
```

Pattern elements

ID3 Tags:

```
a, artist                Artist
A, album                 Album
b, album-artist          Album artist
t, title                 Title
n, track                 Track number
N, track-total           Total track number
d, disc, disc-num        Disc number
D, disc-total            Total disc number
G, genre                 Genre
genre-id                 Genre ID
Y, year                  Release year
c, comment               First comment that matches description and language.
                         Parameters:
                            description (optional)
                            language (optional)
comments                 All comments that are matching description and language (with
                         output placeholders #d as description, #l as language & #t as␣
↪text).
                         Parameters:
                            description (optional)
                            language (optional)
                            output (optional, default='Comment: [Description: #d]␣
↪[Lang: #l]: #t')
                            separation (optional, default='\n')
lyrics                   All lyrics that are matching description and language (with␣
↪output
                         placeholders #d as description, #l as language & #t as text).
                         Parameters:
                            description (optional)
                            language (optional)
                            output (optional, default='Lyrics: [Description: #d]␣
↪[Lang: #l]: #t')
                            separation (optional, default='\n')
release-date             Relase date
original-release-date    Original Relase date
recording-date           Recording date
encoding-date            Encoding date
tagging-date             Tagging date
play-count               Play count
popm, popularities       Popularities (with output placeholders #e as email, #r as␣
↪rating &
                         #c as count)
                         Parameters:
                            output (optional, default='Popularity: [email: #e]␣
↪[rating: #r] [play count: #c]')
                            separation (optional, default='\n')
bpm                      BPM
publisher                Publisher
ufids, unique-file-ids   Unique File IDs (with output placeholders #o as owner & #i as␣
↪unique id)
                         Parameters:
                            output (optional, default='Unique File ID: [#o] : #i')
                            separation (optional, default='\n')
```

(continues on next page)

(continued from previous page)

```
txxx, texts             User text frames (with output placeholders #d as description &
                        #t as text)
                        Parameters:
                           output (optional, default='UserTextFrame: [Description:
↪#d] #t')
                           separation (optional, default='\n')
user-urls               User URL frames (with output placeholders #i as frame id, #d␣
↪as
                        description & #u as url)
                        Parameters:
                           output (optional, default='#i [Description: #d]: #u')
                           separation (optional, default='\n')
artist-url              Artist URL
audio-source-url        Audio source URL
audio-file-url          Audio file URL
internet-radio-url      Internet radio URL
commercial-url          Comercial URL
payment-url             Payment URL
publisher-url           Publisher URL
copyright-url           Copyright URL
images, apic            Attached pictures (APIC)
                        (with output placeholders #t as image type, #m as mime type,
↪#s as size in bytes & #d as description)
                        Parameters:
                           output (optional, default='#t Image: [Type: #m] [Size: #b␣
↪bytes] #d')
                           separation (optional, default='\n')
image-urls              Attached pictures URLs
                        (with output placeholders #t as image type, #m as mime type,
↪#u as URL & #d as description)
                        Parameters:
                           output (optional, default='#t Image: [Type: #m] [URL: #u]
↪#d')
                           separation (optional, default='\n')
objects, gobj           Objects (GOBJ)
                        (with output placeholders #s as size, #m as mime type, #d as␣
↪description and #f as file name)
                        Parameters:
                           output (optional, default='GEOB: [Size: #s bytes] [Type:
↪#t] Description: #d | Filename: #f')
                           separation (optional, default='\n')
privates, priv          Privates (with output placeholders #c as content, #b as␣
↪number of bytes & #o as owner)
                        Parameters:
                           output (optional, default='PRIV-Content: #b bytes | Owner:
↪#o')
                           separation (optional, default='\n')
music-cd-id, mcdi       Music CD Identification
terms-of-use            Terms of use
```

Functions:

```
format              Formats text bold and colored (grey, red, green, yellow, blue,␣
↪magenta,
                    cyan or white)
                    Parameters:
                       text
```

(continues on next page)

(continued from previous page)

```
                        bold (optional)
                        color (optional)
num, number-format   Appends leading zeros
                     Parameters:
                        number
                        digits
filename, fn         File name
                     Parameter:
                        basename (optional)
filesize             Size of file
tag-version          Tag version
length               Length of aufio file
mpeg-version         MPEG version (with output placeholders #v as version & #l as␣
↪layer)
                     Parameter:
                        output (optional, default='MPEG#v\, Layer #l')
bit-rate             Bit rate of aufio file
sample-freq          Sample frequence of aufio file in Hz
audio-mode           Mode of aufio file: mono/stereo
not-empty            If condition is not empty (with output placeholder #t as text)
                     Parameters:
                        text
                        output (optional, default='#t')
                        empty (optional)
repeat               Repeats text
                     Parameters:
                        text
                        count
```

Special characters:

```
escape seq.    character
\\             \
\%             %
\$             $
\,             ,
\(             (
\)             )
\=             =
\n             New line
\t             Tab
```

Example

Asuming an audio file with artist 'Madonna', title 'Frozen' and album 'Ray of Light'

```
%artist% - %album% - %title%
%a% - %A% - %t%
```

Both patterns produce the following output: Madonna - Ray of Light - Frozen

```
$format(title:,bold=y) %title%\n
```

This pattern produces th output: **title:** Frozen

fixup - Music directory fixer

Performs various checks and fixes to directories of audio files.

Names

fixup

Description

Operates on directories at a time, fixing each as a unit (album, compilation, live set, etc.). All of these should have common dates, for example but other characteristics may vary. The `--type` should be used whenever possible, `lp` is the default.

The following test and fixes always apply:

1. Every file will be given an ID3 tag if one is missing.
2. Set ID3 v2.4.
3. Set a consistent album name for all files in the directory.
4. Set a consistent artist name for all files, unless the type is `various` in which case the artist may vary (but must exist).
5. Ensure each file has a title.
6. Ensure each file has a track # and track total.
7. Ensure all files have a release and original release date, unless the type is `live` in which case the recording date is set.
8. All ID3 frames of the following types are removed: USER, PRIV
9. All ID3 files have TLEN (track length in ms) set (or updated).
10. The album/dir type is set in the tag. Types of `lp` and `various` do not have this field set since the latter is the default and the former can be determined during sync. In ID3 terms the value is in TXXX (description: `eyeD3#album_type`).
11. Files are renamed as follows: - Type `various`: ${track:num} - ${artist} - ${title} - Type `single`: ${artist} - ${title} - All other types: ${artist} - ${track:num} - ${title} - A rename template can be supplied in –file-rename-pattern
12. Directories are renamed as follows: - Type `live`: ${best_date:prefer_recording} - ${album} - All other types: ${best_date:prefer_release} - ${album} - A rename template can be supplied in –dir-rename-pattern

Album types:

- `lp`: A traditinal “album” of songs from a single artist. No extra info is written to the tag since this is the default.
- `ep`: A short collection of songs from a single artist. The string ‘ep’ is written to the tag’s `eyeD3#album_type` field.
- `various`: A collection of songs from different artists. The string ‘various’ is written to the tag’s `eyeD3#album_type` field.
- `live`: A collection of live recordings from a single artist. The string ‘live’ is written to the tag’s `eyeD3#album_type` field.

- `compilation`: A collection of songs from various recordings by a single artist. The string 'compilation' is written to the tag's `eyeD3#album_type` field. Compilation dates, unlike other types, may differ.
- `demo`: A demo recording by a single artist. The string 'demo' is written to the tag's `eyeD3#album_type` field.
- `single`: A track that should no be associated with an album (even if it has album metadata). The string 'single' is written to the tag's `eyeD3#album_type` field.

Options

```
--file-rename-pattern 12. Directories are renamed as follows: - Type
``live``: ${best_date:prefer_recording} - ${album} - All other types:
${best_date:prefer_release} - ${album} - A rename template can be supplied
in --dir-rename-pattern Album types: - ``lp``: A traditinal "album" of
songs from a single artist. No extra info is written to the tag since this
is the default. - ``ep``: A short collection of songs from a single
artist. The string 'ep' is written to the tag's ``eyeD3#album_type``
field. - ``various``: A collection of songs from different artists. The
string 'various' is written to the tag's ``eyeD3#album_type`` field. -
``live``: A collection of live recordings from a single artist. The string
'live' is written to the tag's ``eyeD3#album_type`` field. -
``compilation``: A collection of songs from various recordings by a single
artist. The string 'compilation' is written to the tag's
``eyeD3#album_type`` field. Compilation dates, unlike other types, may
differ. - ``demo``: A demo recording by a single artist. The string 'demo'
is written to the tag's ``eyeD3#album_type`` field. - ``single``: A track
that should no be associated with an album (even if it has album
metadata). The string 'single' is written to the tag's
``eyeD3#album_type`` field.

-t {lp,ep,compilation,live,various,demo,single}, --type {lp,ep,compilation,live,
↪various,demo,single}
                      How to treat each directory. The default is 'lp',
                      although you may be prompted for an alternate choice
                      if the files look like another type.
--fix-case            Fix casing on each string field by capitalizing each
                      word.
-n, --dry-run         Only print the operations that would take place, but
                      do not execute them.
--no-prompt           Exit if prompted.
--dotted-dates        Separate date with '.' instead of '-' when naming
                      directories.
--file-rename-pattern FILE_RENAME_PATTERN
                      Rename file (the extension is not affected) based on
                      data in the tag using substitution variables: $album,
                      $album_artist, $artist, $best_date,
                      $best_date:prefer_recording,
                      $best_date:prefer_recording:year,
                      $best_date:prefer_release,
                      $best_date:prefer_release:year, $best_date:year,
                      $disc:num, $disc:total, $file, $file:ext,
                      $original_release_date, $original_release_date:year,
                      $recording_date, $recording_date:year, $release_date,
                      $release_date:year, $title, $track:num, $track:total
--dir-rename-pattern DIR_RENAME_PATTERN
                      Rename directory based on data in the tag using
```

(continues on next page)

(continued from previous page)

```
                        substitution variables: $album, $album_artist,
                        $artist, $best_date, $best_date:prefer_recording,
                        $best_date:prefer_recording:year,
                        $best_date:prefer_release,
                        $best_date:prefer_release:year, $best_date:year,
                        $disc:num, $disc:total, $file, $file:ext,
                        $original_release_date, $original_release_date:year,
                        $recording_date, $recording_date:year, $release_date,
                        $release_date:year, $title, $track:num, $track:total
```

itunes-podcast - Convert files so iTunes recognizes them as podcasts

Adds (or removes) the tags necessary for Apple iTunes to identify the file as a podcast.

Names

itunes-podcast

Description

Options

```
--add       Add the podcast frames.
--remove    Remove the podcast frames.
```

Example

```
$ eyeD3 -P itunes-podcast example.id3

/home/travis/devel/eyeD3/git/example.id3
      iTunes podcast? :-(

$ eyeD3 -P itunes-podcast example.id3 --add

/home/travis/devel/eyeD3/git/example.id3
      iTunes podcast? :-(
      Adding...
      iTunes podcast? :-)

$ eyeD3 -P itunes-podcast example.id3 --remove

/home/travis/devel/eyeD3/git/example.id3
      iTunes podcast? :-)
      Removing...
      iTunes podcast? :-(
```

genres - ID3 Genre List

Display the full list of standard ID3 genres.

Names

genres

Description

ID3 v1 defined a list of genres and mapped them to to numeric values so they can be stored as a single byte. It is *recommended* that these genres are used although most newer software (including eyeD3) does not care.

Options

```
-1, --single-column  List on genre per line.
```

Example

```
$ eyeD3 --plugin=genres

  0: Blues                            96: Big Band
  1: Classic Rock                     97: Chorus
  2: Country                          98: Easy Listening
  3: Dance                            99: Acoustic
  4: Disco                           100: Humour
  5: Funk                            101: Speech
  6: Grunge                          102: Chanson
  7: Hip-Hop                         103: Opera
  8: Jazz                            104: Chamber Music
  9: Metal                           105: Sonata
 10: New Age                         106: Symphony
 11: Oldies                          107: Booty Bass
 12: Other                           108: Primus
 13: Pop                             109: Porn Groove
 14: R&B                             110: Satire
 15: Rap                             111: Slow Jam
 16: Reggae                          112: Club
 17: Rock                            113: Tango
 18: Techno                          114: Samba
 19: Industrial                      115: Folklore
 20: Alternative                     116: Ballad
 21: Ska                             117: Power Ballad
 22: Death Metal                     118: Rhythmic Soul
 23: Pranks                          119: Freestyle
 24: Soundtrack                      120: Duet
 25: Euro-Techno                     121: Punk Rock
 26: Ambient                         122: Drum Solo
 27: Trip-Hop                        123: A Cappella
 28: Vocal                           124: Euro-House
```

(continues on next page)

(continued from previous page)

```
 29: Jazz+Funk                    125: Dance Hall
 30: Fusion                       126: Goa
 31: Trance                       127: Drum & Bass
 32: Classical                    128: Club-House
 33: Instrumental                 129: Hardcore
 34: Acid                         130: Terror
 35: House                        131: Indie
 36: Game                         132: BritPop
 37: Sound Clip                   133: Negerpunk
 38: Gospel                       134: Polsk Punk
 39: Noise                        135: Beat
 40: AlternRock                   136: Christian Gangsta Rap
 41: Bass                         137: Heavy Metal
 42: Soul                         138: Black Metal
 43: Punk                         139: Crossover
 44: Space                        140: Contemporary Christian
 45: Meditative                   141: Christian Rock
 46: Instrumental Pop             142: Merengue
 47: Instrumental Rock            143: Salsa
 48: Ethnic                       144: Thrash Metal
 49: Gothic                       145: Anime
 50: Darkwave                     146: JPop
 51: Techno-Industrial            147: Synthpop
 52: Electronic                   148: Abstract
 53: Pop-Folk                     149: Art Rock
 54: Eurodance                    150: Baroque
 55: Dream                        151: Bhangra
 56: Southern Rock                152: Big Beat
 57: Comedy                       153: Breakbeat
 58: Cult                         154: Chillout
 59: Gangsta Rap                  155: Downtempo
 60: Top 40                       156: Dub
 61: Christian Rap                157: EBM
 62: Pop / Funk                   158: Eclectic
 63: Jungle                       159: Electro
 64: Native American              160: Electroclash
 65: Cabaret                      161: Emo
 66: New Wave                     162: Experimental
 67: Psychedelic                  163: Garage
 68: Rave                         164: Global
 69: Showtunes                    165: IDM
 70: Trailer                      166: Illbient
 71: Lo-Fi                        167: Industro-Goth
 72: Tribal                       168: Jam Band
 73: Acid Punk                    169: Krautrock
 74: Acid Jazz                    170: Leftfield
 75: Polka                        171: Lounge
 76: Retro                        172: Math Rock
 77: Musical                      173: New Romantic
 78: Rock & Roll                  174: Nu-Breakz
 79: Hard Rock                    175: Post-Punk
 80: Folk                         176: Post-Rock
 81: Folk-Rock                    177: Psytrance
 82: National Folk                178: Shoegaze
 83: Swing                        179: Space Rock
 84: Fast Fusion                  180: Trop Rock
 85: Bebob                        181: World Music
```

(continues on next page)

(continued from previous page)

```
 86: Latin                          182: Neoclassical
 87: Revival                        183: Audiobook
 88: Celtic                         184: Audio Theatre
 89: Bluegrass                      185: Neue Deutsche Welle
 90: Avantgarde                     186: Podcast
 91: Gothic Rock                    187: Indie Rock
 92: Progressive Rock               188: G-Funk
 93: Psychedelic Rock               189: Dubstep
 94: Symphonic Rock                 190: Garage Rock
 95: Slow Rock                      191: Psybient
```

lameinfo (xing) - Lame (Xing) Header Information

Outputs lame header (if one exists) for file.

Names

lameinfo (aliases: xing)

Description

The 'lame' (or xing) header provides extra information about the mp3 that is useful to players and encoders but not officially part of the mp3 specification. Variable bit rate mp3s, for example, use this header.

For more details see here

Options

```
No extra options supported
```

Example

```
$ eyeD3 -P lameinfo src/test/data/notag-vbr.mp3


notag-vbr.mp3 [ 5.98 MB ]
--------------------------------------------------------------------------------
Encoder Version     : LAME3.91
LAME Tag Revision   : 0
VBR Method          : Variable Bitrate method2 (mtrh)
Lowpass Filter      : 19500
Encoding Flags      : --nspsytune
ATH Type            : 3
Bitrate (Minimum)   : 0
Encoder Delay       : 576 samples
Encoder Padding     : 1848 samples
Noise Shaping       : 1
Stereo Mode         : Joint
```

(continues on next page)

(continued from previous page)

```
Unwise Settings      : False
Sample Frequency     : 44.1 kHz
MP3 Gain             : 0 (+0.0 dB)
Preset               : Unknown
Surround Info        : None
Music Length         : 5.98 MB
Music CRC-16         : 675C
LAME Tag CRC-16      : 5B62
```

nfo - (I)NFO File Generator

Create NFO files for each directory scanned.

Names

nfo

Description

Each directory scanned is treated as an album and a NFO file is written to standard out.

NFO files are often found in music archives.

Options

```
No extra options supported
```

Example

```
$ eyeD3 -P nfo ~/music/Nine\ Inch\ Nails/1992\ -\ Broken/
```

```
Artist   : Nine Inch Nails
Album    : Broken
Released : 1992
Genre    : Noise

Source   :
Encoder  : LAME3.95
Codec    : mp3
Bitrate  : ~167 K/s @ 44100 Hz, Joint stereo
Tag      : ID3 v2.3

Ripped By:

Track Listing
-------------
 1. Pinion                 (01:02)
 2. Wish                   (03:46)
```

(continues on next page)

(continued from previous page)

```
 3. Last                    (04:44)
 4. Help Me I am in Hell    (01:56)
 5. Happiness in Slavery    (05:21)
 6. Gave Up                 (04:08)
 7. Physical (You're So)    (05:29)
 8. Suck                    (05:07)

Total play time : 31:33
Total size      : 37.74 MB

================================================================================
.NFO file created with eyeD3 0.7.0 on Tue Oct 23 23:44:27 2012
For more information about eyeD3 go to http://eyeD3.nicfit.net/
================================================================================
```

pymod - Use simple python modules as eyeD3 plugins

Imports a Python module file and calls its functions for the the various plugin events.

Names

pymod

Description

If no module if provided (see -m/–module) a file named eyeD3mod.py in the current working directory is imported. If any of the following methods exist they still be invoked:

def audioFile(audio_file): ‘’‘Invoked for every audio file that is encountered. The `audio_file` is of type `eyed3.core.AudioFile`; currently this is the concrete type `eyed3.mp3.Mp3AudioFile`.’‘ pass

def audioDir(d, audio_files, images): ‘’‘This function is invoked for any directory (`d`) that contains audio (`audio_files`) or image (`images`) media.’‘ pass

def done(): ‘’‘This method is invoke before successful exit.’‘ pass

Options

```
-m MODULE, --module MODULE
                      The Python module module to invoke. The default is
                      ./eyeD3mod.py
```

Example

TODO

stats - Music Collection Statistics

Computes statistics for all audio files scanned.

Names

stats

Description

Options

```
--verbose   Show details for each file with rule violations.
```

xep-118 - Jabber (XMPP) Tune Format

Outputs all tags in XEP-118 XML format. (see: http://xmpp.org/extensions/xep-0118.html)

Names

xep-118

Description

Options

```
No extra options supported
```

Configuration Files

Command line options can be read from a configuration file using the `-C/--config` option. It expects a path to an Ini file contain sections with option values. A sample config file, for example:

```
# eyeD3 config file.
# default: ~/.eyeD3/config.ini
# overridde using -c/--config
[default]

# Default plugin to use.
plugin =
```

(continues on next page)

(continued from previous page)

```
# General options to always use. These can be plugin specific but SHOULD NOT be.
# Any -C/--config and -P/--plugin options are ignored.
options =
#options = --pdb

# Extra directories to load plugins. Separated by ':'
plugin_path = ~/.eyeD3


# vim: set filetype=dosini:
```

If the file `${HOME}/.eyeD3/config.ini` exists it is loaded each time eyeD3 is run and the values take effect. This can be disabled with `--no-config`.

Custom Plugins

Plugins are any class found in the plugin search path (see 'plugin_path' in *Configuration Files*) that inherits from *eyed3.plugins.Plugin*. The interface is simple, the basic attributes of the plugin (name, description, etc.) are set using menber variables and for each file `eyeD3` traverses (using the given path(s) and optional `--exclude` options) the method `handleFile` will be called. The return value of this call is ignored, but if you wish to halt processing of files a `StopIteration` exception can be raised. Here is where the plugin should does whatever interesting it things it would like to do with the files it is passed. When all input files are processed the method `handleDone` is called and the program exits. Below is an 'echo' plugin that prints each filename/path and the file's mime-type.

```
import eyed3
from eyed3.plugins import Plugin
from eyed3.utils import guessMimetype


class EchoPlugin(eyed3.plugins.Plugin):
    NAMES = ["echo"]
    SUMMARY = u"Displays each filename and mime-type passed to the plugin"

    def handleFile(self, f):
        print("%s\t[ %s ]" % (f, guessMimetype(f)))
```

Many plugins might prefer to deal with only file types `eyeD3` natively supports, namely mp3 audio files. To automatically load *eyed3.core.AudioFile* objects using *eyed3.core.load()* inherit from the *eyed3.plugins.LoaderPlugin* class. In this model the member `self.audio_file` is initialized to the parsed mp3/id3 objects. If the file is not a supported audio file type the value is set to `None`.

In the next example the `LoaderPlugin` is used to set the `audio_file` member variable which contains the info and tag objects.

```
from eyed3.plugins import LoaderPlugin


class Echo2Plugin(LoaderPlugin):
    SUMMARY = u"Displays details about audio files"
    NAMES = ["echo2"]

    def handleFile(self, f):
        super(Echo2Plugin, self).handleFile(f)
```

(continues on next page)

(continued from previous page)

```
        if not self.audio_file:
            print("%s: Unsupported type" % f)
        else:
            print("Audio info: %s Metadata tag: %s " %
                  ("yes" if self.audio_file.info else "no",
                   "yes" if self.audio_file.tag else "no"))
```

See also:

Configuration Files, *eyed3.plugins.Plugin*, *eyed3.plugins.classic.ClassicPlugin*, *eyed3.mp3.Mp3AudioInfo*, *eyed3.id3.tag.Tag*

4.2.2 Compliance

ID3

Unsupported Features

- ID3 frame encryption
- Writing of sync-safe data (i.e. unsynchronized) because it is 2012. Reading of unsyncronized tags (v2.3) and frames (v2.4) **is** supported.

Dates

One of the major differences between 2.3 and 2.4 is dates.

ID3 v2.3 Date Frames

- TDAT date (recording date of form DDMM, always 4 bytes)
- TYER year (recording year of form YYYY, always 4 bytes)
- TIME time (recording time of form HHMM, always 4 bytes)
- TORY orig release year
- TRDA recording date (more freeform replacement for TDAT, TYER, TIME. e.g., “4th-7th June, 12th June” in combination with TYER)
- TDLY playlist delay (also defined in ID3 v2.4)

ID3 v2.4 Date Frames

All v2.4 dates follow ISO 8601 formats.

- TDEN encoding datetime
- TDOR orig release date
- TDRC recording date
- TDRL release date
- TDTG tagging time
- TDLY playlist delay (also defined in ID3 v2.3)

From the ID3 specs:

```
yyyy-MM-ddTHH:mm:ss (year, "-", month, "-", day, "T", hour (out of
24), ":", minutes, ":", seconds), but the precision may be reduced by
removing as many time indicators as wanted. Hence valid timestamps
are yyyy, yyyy-MM, yyyy-MM-dd, yyyy-MM-ddTHH, yyyy-MM-ddTHH:mm
and yyyy-MM-ddTHH:mm:ss. All time stamps are UTC. For
durations, use the slash character as described in 8601, and for
multiple non- contiguous dates, use multiple strings, if allowed
by the frame definition.
```

The ISO 8601 'W' delimiter for numeric weeks is NOT supported.

Times that contain a 'Z' at the end to signal the time is UTC is supported.

Common Date Frame Extensions

MusicBrainz uses *XDOR* in v2.3 tags as the **full** original release date, whereas *TORY* (v2.3) only represents the release year. Version 2.4 does not use/need this extension since *TDOR* is available.

v2.4 <-> 2.3 mappings

When converting to/from v2.3 and v2.4 it is neceswsary to convert date frames. The following is the mappings eyeD3 uses when converting.

Version 2.3 –> version 2.4

- TYER, TDAT, TIME –> TDRC
- TORY –> TDOR
- TRDA –> none
- XDOR –> TDOR

If both *TORY* and *XDOR* exist, XDOR is preferred.

Version 2.4 –> version 2.3

- TDRC –> TYER, TDAT, TIME
- TDOR –> TORY
- TDRL –> TORY
- TDEN –> none

- TDTG –> none

Non Standard Frame Support

NCON

A MusicMatch extension of unknown binary format. Frames of this type are parsed as raw `Frame` objects, therefore the data is not parsed. The frames are preserved and can be deleted and written (as is).

TCMP

An iTunes extension to signify that a track is part of a compilation. This frame is handled by `TextFrame` and the data is either a '1' if part of a compilation or '0' (or empty) if not.

XSOA, XSOP, XSOT

These are alternative sort-order strings for album, performer, and title, respectively. They are often added to ID3v2.3 tags while v2.4 does not require them since TSOA, TSOP, and TSOT are native frames.

These frames are preserved but are not written when using v2.3. If the tag is converted to v2.4 then the corresponding native frame is used.

XDOR

A MusicBrainz extension for the **full** original release date, since TORY only contains the year of original release. In ID3 v2.4 this frame became TDOR.

PCST, WFED, TKWD, TDES, TGID

Apple extensions for podcasts.

4.2.3 eyeD3

eyed3 package

Subpackages

eyed3.id3 package

Submodules

eyed3.id3.apple module

Here lies Apple frames, all of which are non-standard. All of these would have been standard user text frames by anyone not being a bastard, on purpose.

class eyed3.id3.apple.**PCST**(*id=b'PCST'*)
Bases: *eyed3.id3.frames.Frame*

Indicates a podcast. The 4 bytes of data is undefined, and is typically all 0.

render()

class eyed3.id3.apple.**TKWD**(*id=b'TKWD'*)
Bases: *eyed3.id3.frames.TextFrame*

Podcast keywords.

class eyed3.id3.apple.**TDES**(*id=b'TDES'*)
Bases: *eyed3.id3.frames.TextFrame*

Podcast description. One encoding byte followed by text per encoding.

class eyed3.id3.apple.**TGID**(*id=b'TGID'*)
Bases: *eyed3.id3.frames.TextFrame*

Podcast URL of the audio file. This should be a W frame!

class eyed3.id3.apple.**WFED**(*id=b'WFED'*, *url=''*)
Bases: *eyed3.id3.frames.TextFrame*

Another podcast URL, the feed URL it is said.

eyed3.id3.frames module

exception eyed3.id3.frames.**FrameException**(**args*)
Bases: *eyed3.Error*

class eyed3.id3.frames.**Frame**(***kwargs*)
Bases: object

property header

parse(***kwargs*)

render()

static decompress(*data*)

static compress(*data*)

static decrypt(*data*)

static encrypt(*data*)

property text_delim

property encoding

class eyed3.id3.frames.**TextFrame**(***kwargs*)
Bases: *eyed3.id3.frames.Frame*

Text frames. Data string format: encoding (one byte) + text

property text

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**UserTextFrame**(***kwargs*)
Bases: *eyed3.id3.frames.TextFrame*

property description

parse(*data*, *frame_header*)
Data string format: encoding (one byte) + description + b"" + text

render()

class eyed3.id3.frames.**DateFrame**(*id*, *date=''*)
Bases: *eyed3.id3.frames.TextFrame*

parse(*data*, *frame_header*)

property date

class eyed3.id3.frames.**UrlFrame**(*id*, *url=''*)
Bases: *eyed3.id3.frames.Frame*

property url

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**UserUrlFrame**(***kwargs*)
Bases: *eyed3.id3.frames.UrlFrame*

Data string format: encoding (one byte) + description + b"" + url (iso-8859-1)

property description

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**ImageFrame**(***kwargs*)
Bases: *eyed3.id3.frames.Frame*

OTHER = 0

ICON = 1

OTHER_ICON = 2

FRONT_COVER = 3

BACK_COVER = 4

LEAFLET = 5

MEDIA = 6

LEAD_ARTIST = 7

ARTIST = 8

CONDUCTOR = 9

BAND = 10

COMPOSER = 11

LYRICIST = 12

RECORDING_LOCATION = 13

DURING_RECORDING = 14

DURING_PERFORMANCE = 15

VIDEO = 16

BRIGHT_COLORED_FISH = 17

ILLUSTRATION = 18

BAND_LOGO = 19

PUBLISHER_LOGO = 20

MIN_TYPE = 0

MAX_TYPE = 20

URL_MIME_TYPE = b'-->'

URL_MIME_TYPE_STR = '-->'

URL_MIME_TYPE_VALUES = (b'-->', '-->')

property description

property mime_type

property picture_type

parse(*data*, *frame_header*)

render()

static picTypeToString(*t*)

static stringToPicType(*s*)

makeFileName(*name=None*)

class eyed3.id3.frames.**ObjectFrame**(***kwargs*)

Bases: *eyed3.id3.frames.Frame*

property description

property mime_type

property filename

parse(*data*, *frame_header*)

Parse the frame from `data` bytes using details from `frame_header`.

Data string format: <Header for 'General encapsulated object', ID: "GEOB"> Text encoding $xx MIME type <text string> $00 Filename <text string according to encoding> $00 (00) Content description <text string according to encoding> $00 (00) Encapsulated object <binary data>

render()

class eyed3.id3.frames.**PrivateFrame**(*id=b'PRIV'*, *owner_id=b''*, *owner_data=b''*)

Bases: *eyed3.id3.frames.Frame*

PRIV

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**MusicCDIdFrame**(*id=b'MCDI'*, *toc=b''*)

Bases: *eyed3.id3.frames.Frame*

property toc

parse(*data*, *frame_header*)

class eyed3.id3.frames.**PlayCountFrame**(*id=b'PCNT'*, *count=0*)
Bases: *eyed3.id3.frames.Frame*

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**PopularityFrame**(*id=b'POPM'*, *email=b''*, *rating=0*, *count=0*)
Bases: *eyed3.id3.frames.Frame*

Frame type for 'POPM' frames; popularity. Frame format: <Header for 'Popularimeter', ID: "POPM"> Email to user <text string> $00 Rating $xx Counter $xx xx xx xx (xx ...)

property rating

property email

property count

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**UniqueFileIDFrame**(*id=b'UFID'*, *owner_id=b''*, *uniq_id=b''*)
Bases: *eyed3.id3.frames.Frame*

property owner_id

property uniq_id

parse(*data*, *frame_header*)
Data format Owner identifier <text string> $00 Identifier up to 64 bytes binary data>

render()

class eyed3.id3.frames.**LanguageCodeMixin**
Bases: object

property lang

class eyed3.id3.frames.**DescriptionLangTextFrame**(***kwargs*)
Bases: *eyed3.id3.frames.Frame*, *eyed3.id3.frames.LanguageCodeMixin*

property description

property text

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**CommentFrame**(*id=b'COMM'*, *description=''*, *lang=b'eng'*, *text=''*)
Bases: *eyed3.id3.frames.DescriptionLangTextFrame*

class eyed3.id3.frames.**LyricsFrame**(*id=b'USLT'*, *description=''*, *lang=b'eng'*, *text=''*)
Bases: *eyed3.id3.frames.DescriptionLangTextFrame*

class eyed3.id3.frames.**TermsOfUseFrame**(***kwargs*)
Bases: *eyed3.id3.frames.Frame*, *eyed3.id3.frames.LanguageCodeMixin*

property text

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**TocFrame**(***kwargs*)
Bases: *eyed3.id3.frames.Frame*

Table of content frame. There may be more than one, but only one may have the top-level flag set.

Data format: Element ID: <string>TOC flags: %000000ab Entry count: %xx Child elem IDs: <string>(... num entry count) Description: TIT2 frame (optional)

TOP_LEVEL_FLAG_BIT = 6

ORDERED_FLAG_BIT = 7

parse(*data*, *frame_header*)

render()

class eyed3.id3.frames.**StartEndTuple**(*start*, *end*)
Bases: tuple

A 2-tuple, with names 'start' and 'end'.

property end
Alias for field number 1

property start
Alias for field number 0

class eyed3.id3.frames.**ChapterFrame**(*id=b'CHAP'*, *element_id=None*, *times=None*, *offsets=None*, *sub_frames=None*)
Bases: *eyed3.id3.frames.Frame*

Frame type for chapter/section of the audio file. <ID3v2.3 or ID3v2.4 frame header, ID: "CHAP"> (10 bytes) Element ID <text string> $00 Start time $xx xx xx xx End time $xx xx xx xx Start offset $xx xx xx xx End offset $xx xx xx xx <Optional embedded sub-frames>

NO_OFFSET = 4294967295
No offset value, aka '0xff0xff0xff0xff'

parse(*data*, *frame_header*)

render()

property title

property subtitle

property user_url

class eyed3.id3.frames.**FrameSet**
Bases: dict

parse(*f*, *tag_header*, *extended_header*)
Read frames starting from the current read position of the file object. Returns the amount of padding which occurs after the tag, but before the audio content. A return valule of 0 does not mean error.

getAllFrames()
Return all the frames in the set as a list. The list is sorted in an arbitrary but consistent order.

setTextFrame(***kwargs*)

eyed3.id3.frames.**deunsyncData**(*data*)

eyed3.id3.frames.**createFrame**(*tag_header*, *frame_header*, *data*)

eyed3.id3.frames.**decodeUnicode**(*bites*, *encoding*)

eyed3.id3.frames.**splitUnicode**(*data*, *encoding*)

eyed3.id3.frames.**id3EncodingToString**(*encoding*)

eyed3.id3.frames.**stringToEncoding**(*s*)

eyed3.id3.frames.**map2_2FrameId**(*orig_id*)

eyed3.id3.headers module

class eyed3.id3.headers.**TagHeader**(*version=(2, 4, 0)*)

Bases: object

SIZE = 10

clear()

property version

property major_version

property minor_version

property rev_version

parse(*f*)

Parse an ID3 v2 header starting at the current position of *f*.

If a header is parsed *True* is returned, otherwise *False*. If a header is found but malformed an *eyed3.id3.tag.TagException* is thrown.

render(*tag_len=None*)

class eyed3.id3.headers.**ExtendedTagHeader**

Bases: object

RESTRICT_TAG_SZ_LARGE = 0

RESTRICT_TAG_SZ_MED = 1

RESTRICT_TAG_SZ_SMALL = 2

RESTRICT_TAG_SZ_TINY = 3

RESTRICT_TEXT_ENC_NONE = 0

RESTRICT_TEXT_ENC_UTF8 = 1

RESTRICT_TEXT_LEN_NONE = 0

RESTRICT_TEXT_LEN_1024 = 1

RESTRICT_TEXT_LEN_128 = 2

RESTRICT_TEXT_LEN_30 = 3

RESTRICT_IMG_ENC_NONE = 0

RESTRICT_IMG_ENC_PNG_JPG = 1

RESTRICT_IMG_SZ_NONE = 0

RESTRICT_IMG_SZ_256 = 1

RESTRICT_IMG_SZ_64 = 2

RESTRICT_IMG_SZ_64_EXACT = 3

property update_bit

property crc_bit

property crc

property restrictions_bit

property tag_size_restriction

property tag_size_restriction_description

property text_enc_restriction

property text_enc_restriction_description

property text_length_restriction

property text_length_restriction_description

property image_enc_restriction

property image_enc_restriction_description

property image_size_restriction

property image_size_restriction_description

render(*version*, *frame_data*, *padding=0*)

parse(*fp*, *version*)

Parse an ID3 v2 extended header starting at the current position of `fp` and per the format defined by `version`. This method should only be called when the presence of an extended header is known since it moves the file position. If a header is found but malformed an `eyed3.id3.tag.TagException` is thrown. The return value is `None`.

class eyed3.id3.headers.**FrameHeader**(***kwargs*)

Bases: `object`

A header for each and every ID3 frame in a tag.

TAG_ALTER = None

FILE_ALTER = None

READ_ONLY = None

COMPRESSED = None

ENCRYPTED = None

GROUPED = None

UNSYNC = None

DATA_LEN = None

copyFlags(*rhs*)

property major_version

property minor_version

property version

property tag_alter

property file_alter

property read_only

property compressed

property encrypted

property grouped

property unsync

property data_length_indicator

render(*data_size*)

static parse(*f*, *version*)

eyed3.id3.tag module

exception eyed3.id3.tag.**TagException**(**args*)
Bases: *eyed3.Error*

class eyed3.id3.tag.**Tag**(***kwargs*)
Bases: *eyed3.core.Tag*

clear()
Reset all tag data.

parse(*fileobj*, *version=(3*, *None*, *None)*)

property version

isV1()
Test ID3 major version for v1.x

isV2()
Test ID3 major version for v2.x

setTextFrame(***kwargs*)

getTextFrame(*fid*)

property composer

property comments

property bpm

property play_count

property publisher

property cd_id

property images

property encoding_date

property best_release_date
This method tries its best to return a date of some sort, amongst alll the possible date frames. The order of preference for a release date is 1) date of original release 2) date of this versions release 3) the recording date. Or None is returned.

getBestDate(*prefer_recording_date=False*)
This method returns a date of some sort, amongst all the possible date frames. The order of preference is:

1) date of original release
2) date of this versions release
3) the recording date.

Unless `prefer_recording_date` is `True` in which case the order is 3, 1, 2.

`None` will be returned if no dates are available.

property release_date
The date the audio was released. This is NOT the original date the work was released, instead it is more like the pressing or version of the release. Original release date is usually what is intended but many programs use this frame and/or don't distinguish between the two.

property original_release_date
The date the work was originally released.

property recording_date
The date of the recording. Many applications use this for release date regardless of the fact that this value is rarely known, and release dates are more correct.

property tagging_date

property lyrics

property disc_num

property objects

property privates

property popularities

property genre

property non_std_genre
partial(func, *args, **keywords) - new function with partial application of the given arguments and keywords.

property user_text_frames

property commercial_url

property copyright_url

property audio_file_url

property audio_source_url

property artist_url

property internet_radio_url

property payment_url

property publisher_url

property user_url_frames

property unique_file_ids

property terms_of_use

save(*filename=None*, *version=None*, *encoding=None*, *backup=False*, *preserve_file_time=False*, *max_padding=None*)
Save the tag. If `filename` is not give the value from the `file_info` member is used, or a `TagException` is raised. The `version` argument can be used to select an ID3 version other than the version read. `Select text encoding with ``encoding` or use the existing (or default) encoding. If `backup` is True the orignal file is preserved; likewise if `preserve_file_time` is True the file´s modification/access times are not updated.

static remove(*filename*, *version=(3*, *None*, *None)*, *preserve_file_time=False*)

property chapters

property table_of_contents

property album_type

property artist_origin
Returns None or a *ArtistOrigin* dataclass: (city, state, country) Any may be `None`.

frameiter(*fids=None*)
A iterator for tag frames. If `fids` is passed it must be a list of frame IDs to filter and return.

property original_artist

class eyed3.id3.tag.**FileInfo**(*file_name*, *tagsz=0*, *tpadd=0*)
Bases: `object`

This class is for storing information about a parsed file. It contains info such as the filename, original tag size, and amount of padding; all of which can make rewriting faster.

initStatTimes()

touch(*times*)
times is a 2-tuple of (atime, mtime).

class eyed3.id3.tag.**AccessorBase**(*fid*, *fs*, *match_func=None*)
Bases: `object`

get(**args*, ***kwargs*)

remove(**args*, ***kwargs*)
Returns the removed item or `None` if not found.

class eyed3.id3.tag.**DltAccessor**(*FrameClass*, *fid*, *fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(***kwargs*)

get(***kwargs*)

class eyed3.id3.tag.**CommentsAccessor**(*fs*)
Bases: *eyed3.id3.tag.DltAccessor*

class eyed3.id3.tag.**LyricsAccessor**(*fs*)
Bases: *eyed3.id3.tag.DltAccessor*

class eyed3.id3.tag.**ImagesAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(***kwargs*)

get(***kwargs*)

class eyed3.id3.tag.**ObjectsAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(***kwargs*)

get(***kwargs*)

class eyed3.id3.tag.**PrivatesAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(*data*, *owner_id*)

remove(*owner_id*)
Returns the removed item or None if not found.

get(*owner_id*)

class eyed3.id3.tag.**UserTextsAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(***kwargs*)

get(***kwargs*)

class eyed3.id3.tag.**UniqueFileIdAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(*data*, *owner_id*)

remove(*owner_id*)
Returns the removed item or None if not found.

get(*owner_id*)

class eyed3.id3.tag.**UserUrlsAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(***kwargs*)

get(***kwargs*)

class eyed3.id3.tag.**PopularitiesAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(*email*, *rating*, *play_count*)

remove(*email*)
Returns the removed item or None if not found.

get(*email*)

class eyed3.id3.tag.**ChaptersAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(*element_id*, *times*, *offsets=(None*, *None)*, *sub_frames=None*)

remove(*element_id*)
Returns the removed item or None if not found.

get(*element_id*)

class eyed3.id3.tag.**TocAccessor**(*fs*)
Bases: *eyed3.id3.tag.AccessorBase*

set(***kwargs*)

remove(*element_id*)
Returns the removed item or None if not found.

get(*element_id*)

class eyed3.id3.tag.**TagTemplate**(*pattern*, *path_friendly='-'*, *dotted_dates=False*)
Bases: string.Template

pattern = re.compile('\n \\$(?:\n (?P<escaped>\\$) | # Escape sequence of two delimite

idpattern = '[_a-z][_a-z0-9:]*'

substitute(*tag*, *zeropad=True*)

safe_substitute(*tag*, *zeropad=True*)

Module contents

eyed3.id3.**isValidVersion**(*v*, *fully_qualified=False*)
Check the tuple v against the list of valid ID3 version constants. If fully_qualified is True it is enforced that there are 3 components to the version in v. Returns True when valid and False otherwise.

eyed3.id3.**normalizeVersion**(*v*)
If version tuple v is of the non-specific type (v1 or v2, any, etc.) a fully qualified version is returned.

eyed3.id3.**versionToString**(*v*)
Conversion version tuple v to a string description.

exception eyed3.id3.**GenreException**(**args*)
Bases: *eyed3.Error*

Excpetion type for exceptions related to genres.

class eyed3.id3.**Genre**(***kwargs*)
Bases: object

A genre in terms of a name and and id. Only when name is a "standard" genre (as defined by ID3 v1) will id be a value other than None.

property id
The Genre's id property. When setting the value is strictly enforced and if the value is not a valid genre code a ValueError is raised. Otherwise the id is set **and** the name property is updated to the code's string name.

property name
The Genre's name property. When setting the value the name is looked up in the standard genre map and if found the id ppropery is set to the numeric valud **and** the name is normalized to the sting found in the map. Non standard genres are set (with a warning log) and the id is set to None. It is valid to set the value to None.

static parse(**args*, ***kwargs*)

class eyed3.id3.**GenreMap**(**args*)
Bases: dict

Classic genres defined around ID3 v1 but suitable anywhere. This class is used primarily as a way to map numeric genre values to a string name. Genre strings on the other hand are not required to exist in this list.

The optional *args are passed directly to the dict constructor.

GENRE_MIN = 0

GENRE_MAX = 191

ID3_GENRE_MIN = 0

ID3_GENRE_MAX = 79

WINAMP_GENRE_MIN = 80

WINAMP_GENRE_MAX = 191

class eyed3.id3.**TagFile**(*path*, *version=(3*, *None*, *None)*)
Bases: *eyed3.core.AudioFile*

A shim class for dealing with files that contain only ID3 data, no audio.

Construct with a path and invoke _read. All other members are set to None.

initTag(*version=(2*, *4*, *0)*)
Add a id3.Tag to the file (removing any existing tag if one exists).

eyed3.mp3 package

Submodules

eyed3.mp3.headers module

eyed3.mp3.headers.**isValidHeader**(*header*)
Determine if header (an integer, 4 bytes compared) is a valid mp3 frame header.

eyed3.mp3.headers.**findHeader**(*fp*, *start_pos=0*)
Locate the first mp3 header in file stream fp starting a offset start_pos (defaults to 0). Returned is a 3-tuple containing the offset where the header was found, the header as an integer, and the header as 4 bytes. If no header is found header_int will equal 0.

eyed3.mp3.headers.**timePerFrame**(*mp3_header*, *vbr*)
Computes the number of seconds per mp3 frame. It can be used to compute overall playtime and bitrate. The mp3 layer and sample rate from mp3_header are used to compute the number of seconds (fractional float point value) per mp3 frame. Be sure to set vbr True when dealing with VBR, otherwise playtimes may be incorrect.

eyed3.mp3.headers.**compute_time_per_frame**(*mp3_header*)
Deprecated since version 0.9a2: This will be removed in 1.0. Use timePerFrame instead

class eyed3.mp3.headers.**Mp3Header**(*header_data=None*)
Bases: object

Header container for MP3 frames.

decode(*header*)

class eyed3.mp3.headers.**VbriHeader**
Bases: object

decode(*frame*)

class eyed3.mp3.headers.**XingHeader**
Bases: object

Header class for the Xing header extensions.

decode(*frame*)

class eyed3.mp3.headers.**LameHeader**(*frame*)
Bases: dict

Mp3 Info tag (AKA LAME Tag)

Lame (and some other encoders) write a tag containing various bits of info about the options used at encode time. If available, the following are parsed and stored in the LameHeader dict:

encoder_version: short encoder version [str] tag_revision: revision number of the tag [int] vbr_method: VBR method used for encoding [str] lowpass_filter: lowpass filter frequency in Hz [int] replaygain: if available, radio and audiofile gain (see below) [dict] encoding_flags: encoding flags used [list] nogap: location of gaps when –nogap was used [list] ath_type: ATH type [int] bitrate: bitrate and type (Constant, Target, Minimum) [tuple] encoder_delay: samples added at the start of the mp3 [int] encoder_padding: samples added at the end of the mp3 [int] noise_shaping: noise shaping method [int] stereo_mode: stereo mode used [str] unwise_settings: whether unwise settings were used [boolean] sample_freq: source sample frequency [str] mp3_gain: mp3 gain adjustment (rarely used) [float] preset: preset used [str] surround_info: surround information [str] music_length: length in bytes of original mp3 [int] music_crc: CRC-16 of the mp3 music data [int] infotag_crc: CRC-16 of the info tag [int]

Prior to ~3.90, Lame simply stored the encoder version in the first frame. If the infotag_crc is invalid, then we try to read this version string. A simple way to tell if the LAME Tag is complete is to check for the infotag_crc key.

Replay Gain data is only available since Lame version 3.94b. If set, the replaygain dict has the following structure:

code peak_amplitude: peak signal amplitude [float] radio:

name: name of the gain adjustment [str] adjustment: gain adjustment [float] originator: originator of the gain adjustment [str]

audiofile: [same as radio]

endcode

Note that as of 3.95.1, Lame uses 89dB as a reference level instead of the 83dB that is specified in the Replay Gain spec. This is not automatically compensated for. You can do something like this if you want:

code import eyeD3 af = eyeD3.mp3.Mp3AudioFile('/path/to/some.mp3') lamever = af.lameTag['encoder_version'] name, ver = lamever[:4], lamever[4:] gain = af.lameTag['replaygain']['radio']['adjustment'] if name == 'LAME' and eyeD3.mp3.lamevercmp(ver, '3.95') > 0:

gain -= 6

endcode

Radio and Audiofile Replay Gain are often referrered to as Track and Album gain, respectively. See http://replaygain.hydrogenaudio.org/ for futher details on Replay Gain.

See http://gabriel.mp3-tech.org/mp3infotag.html for the gory details of the LAME Tag.

Read the LAME info tag. frame should be the first frame of an mp3.

```
ENCODER_FLAGS = {'NOGAP_NEXT': 4, 'NOGAP_PREV': 8, 'NSPSYTUNE': 1, 'NSSAFEJOINT': 2}

PRESETS = {0:  'Unknown', 410:  'V9', 420:  'V8', 430:  'V7', 440:  'V6', 450:  'V5',

REPLAYGAIN_NAME = {0:  'Not set', 1:  'Radio', 2:  'Audiofile'}

REPLAYGAIN_ORIGINATOR = {0:  'Not set', 1:  'Set by artist', 2:  'Set by user', 3:  'S

SAMPLE_FREQUENCIES = {0:  '<= 32 kHz', 1:  '44.1 kHz', 2:  '48 kHz', 3:  '> 48 kHz'}

STEREO_MODES = {0:  'Mono', 1:  'Stereo', 2:  'Dual', 3:  'Joint', 4:  'Force', 5:  'A

SURROUND_INFO = {0:  'None', 1:  'DPL encoding', 2:  'DPL2 encoding', 3:  'Ambisonic e

VBR_METHODS = {0:  'Unknown', 1:  'Constant Bitrate', 2:  'Average Bitrate', 3:  'Vari
```

decode(*frame*)
Decode the LAME info tag.

eyed3.mp3.headers.**lamevercmp**(*x*, *y*)
Compare LAME version strings.

alpha and beta versions are considered older. Versions with sub minor parts or end with 'r' are considered newer.

Parameters

- **x** – The first version to compare.
- **y** – The second version to compare.

Returns Return negative if x<y, zero if x==y, positive if x>y.

Module contents

exception eyed3.mp3.**Mp3Exception**(**args*)
Bases: *eyed3.Error*

Used to signal mp3-related errors.

class eyed3.mp3.**Mp3AudioInfo**(*file_obj*, *start_offset*, *tag*)
Bases: *eyed3.core.AudioInfo*

property bit_rate_str

class eyed3.mp3.**Mp3AudioFile**(*path*, *version=(3*, *None*, *None)*)
Bases: *eyed3.core.AudioFile*

Audio file container for mp3 files.

Construct with a path and invoke `_read`. All other members are set to None.

initTag(*version=(2*, *4*, *0)*)
Add a id3.Tag to the file (removing any existing tag if one exists).

property tag
Returns a concrete implemenation of *eyed3.core.Tag*

eyed3.plugins package

Submodules

eyed3.plugins.art module

class eyed3.plugins.art.**ArtFile**(*file_path*)
Bases: `object`

property image_data

property mime_type

class eyed3.plugins.art.**ArtPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

SUMMARY = 'Art for albums, artists, etc.'

DESCRIPTION = ''

NAMES = ['art']

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

handleDirectory(*d*, *_*)
Override to make use of `self._file_cache`. By default the list is cleared, subclasses should consider doing the same otherwise every AudioFile will be cached.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.plugins.art.**pilImage**(*source*)

eyed3.plugins.art.**pilImageDetails**(*img*)

eyed3.plugins.art.**md5Data**(*data*)

eyed3.plugins.art.**md5File**(*file_name*)
Compute md5 hash for contents of `file_name`.

eyed3.plugins.classic module

class eyed3.plugins.classic.**ClassicPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

SUMMARY = 'Classic eyeD3 interface for viewing and editing tags.'

DESCRIPTION = '\nAll PATH arguments are parsed and displayed. Directory paths are sear

NAMES = ['classic']

handleFile(*f*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

printHeader(*file_path*)

printAudioInfo(*info*)

printTag(*tag*)

handleRemoves(*tag*)

handlePadding(*tag*)

handleEdits(*tag*)

eyed3.plugins.display module

class eyed3.plugins.display.**Pattern**(*text=None*, *sub_patterns=None*)
Bases: object

output_for(*audio_file*)

property sub_patterns

static sub_pattern_classes(*base_class*)

static pattern_class_parameters(*pattern_class*)

class eyed3.plugins.display.**TextPattern**(*text*)
Bases: *eyed3.plugins.display.Pattern*

SPECIAL_CHARACTERS = ['\\', '%', '$', ',', '(', ')', '=', 'n', 't']

SPECIAL_CHARACTERS_DESCRIPTIONS = ['\\', '%', '$', ',', '(', ')', '=', 'New line', 'Ta

output_for(*audio_file*)

class eyed3.plugins.display.**ComplexPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.Pattern*

TYPE = 'unknown'

NAMES = []

DESCRIPTION = ''

PARAMETERS = []

class ExpectedParameter(*name*, ***kwargs*)
Bases: object

class Parameter(*value*, *provided*)
Bases: object

output_for(*audio_file*)

property parameters

property name

class eyed3.plugins.display.**PlaceholderUsagePattern**
Bases: object

class eyed3.plugins.display.**TagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.ComplexPattern*

TYPE = 'tag'

class eyed3.plugins.display.**ArtistTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['a', 'artist']

DESCRIPTION = 'Artist'

class eyed3.plugins.display.**AlbumTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['A', 'album']

DESCRIPTION = 'Album'

class eyed3.plugins.display.**AlbumArtistTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['b', 'album-artist']

DESCRIPTION = 'Album artist'

class eyed3.plugins.display.**ComposerTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['C', 'composer']

DESCRIPTION = 'Composer'

class eyed3.plugins.display.**TitleTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['t', 'title']

DESCRIPTION = 'Title'

class eyed3.plugins.display.**TrackTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['n', 'track']

DESCRIPTION = 'Track number'

class eyed3.plugins.display.**TrackTotalTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['N', 'track-total']

DESCRIPTION = 'Total track number'

class eyed3.plugins.display.**DiscTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['d', 'disc', 'disc-num']

DESCRIPTION = 'Disc number'

class eyed3.plugins.display.**DiscTotalTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['D', 'disc-total']

DESCRIPTION = 'Total disc number'

class eyed3.plugins.display.**GenreTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['G', 'genre']

DESCRIPTION = 'Genre'

class eyed3.plugins.display.**GenreIdTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['genre-id']

DESCRIPTION = 'Genre ID'

class eyed3.plugins.display.**YearTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['Y', 'year']

DESCRIPTION = 'Release year'

class eyed3.plugins.display.**DescriptableTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

PARAMETERS = [description(None), language(None)]

class eyed3.plugins.display.**CommentTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.DescriptableTagPattern*

NAMES = ['c', 'comment']

PARAMETERS = [description(None), language(None)]

DESCRIPTION = 'First comment that matches description and language.'

class eyed3.plugins.display.**AllCommentsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.DescriptableTagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['comments']

PARAMETERS = [description(None), language(None), output(Comment: [Description: #d] [

DESCRIPTION = 'All comments that are matching description and language (with output pla

class eyed3.plugins.display.**AbstractDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

class eyed3.plugins.display.**ReleaseDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.AbstractDateTagPattern*

NAMES = ['release-date']

DESCRIPTION = 'Relase date'

class eyed3.plugins.display.**OriginalReleaseDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.AbstractDateTagPattern*

NAMES = ['original-release-date']

DESCRIPTION = 'Original Relase date'

class eyed3.plugins.display.**RecordingDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.AbstractDateTagPattern*

NAMES = ['recording-date']

DESCRIPTION = 'Recording date'

class eyed3.plugins.display.**EncodingDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.AbstractDateTagPattern*

NAMES = ['encoding-date']

DESCRIPTION = 'Encoding date'

class eyed3.plugins.display.**TaggingDateTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.AbstractDateTagPattern*

NAMES = ['tagging-date']

DESCRIPTION = 'Tagging date'

class eyed3.plugins.display.**PlayCountTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['play-count']

DESCRIPTION = 'Play count'

class eyed3.plugins.display.**PopularitiesTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['popm', 'popularities']

PARAMETERS = [output(Popularity: [email: #e] [rating: #r] [play count: #c]), separa

DESCRIPTION = 'Popularities (with output placeholders #e as email, #r as rating & #c a

class eyed3.plugins.display.**BPMTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['bpm']

DESCRIPTION = 'BPM'

class eyed3.plugins.display.**PublisherTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['publisher']

DESCRIPTION = 'Publisher'

class eyed3.plugins.display.**UniqueFileIDTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['ufids', 'unique-file-ids']

PARAMETERS = [output(Unique File ID: [#o] : #i), separation(\n)]

DESCRIPTION = 'Unique File IDs (with output placeholders #o as owner & #i as unique id

class eyed3.plugins.display.**LyricsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.DescriptableTagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['lyrics']

PARAMETERS = [description(None), language(None), output(Lyrics: [Description: #d] [L

DESCRIPTION = 'All lyrics that are matching description and language (with output plac

class eyed3.plugins.display.**TextsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['txxx', 'texts']

PARAMETERS = [output(UserTextFrame: [Description: #d] #t), separation(\n)]

DESCRIPTION = 'User text frames (with output placeholders #d as description & #t as te

class eyed3.plugins.display.**ArtistURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['artist-url']

DESCRIPTION = 'Artist URL'

class eyed3.plugins.display.**AudioSourceURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['audio-source-url']

DESCRIPTION = 'Audio source URL'

class eyed3.plugins.display.**AudioFileURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['audio-file-url']

DESCRIPTION = 'Audio file URL'

class eyed3.plugins.display.**InternetRadioURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['internet-radio-url']

DESCRIPTION = 'Internet radio URL'

class eyed3.plugins.display.**CommercialURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['commercial-url']

DESCRIPTION = 'Comercial URL'

class eyed3.plugins.display.**PaymentURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['payment-url']

DESCRIPTION = 'Payment URL'

class eyed3.plugins.display.**PublisherURLTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['publisher-url']

DESCRIPTION = 'Publisher URL'

class eyed3.plugins.display.**CopyrightTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['copyright-url']

DESCRIPTION = 'Copyright URL'

class eyed3.plugins.display.**UserURLsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['user-urls']

PARAMETERS = [output(#i [Description: #d]: #u), separation(\n)]

DESCRIPTION = 'User URL frames (with output placeholders #i as frame id, #d as descrip

class eyed3.plugins.display.**ImagesTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['images', 'apic']

PARAMETERS = [output(#t Image: [Type: #m] [Size: #s bytes] #d), separation(\n)]

DESCRIPTION = 'Attached pictures (APIC)(with output placeholders #t as image type, #m

class eyed3.plugins.display.**ImageURLsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['image-urls']

PARAMETERS = [output(#t Image: [Type: #m] [URL: #u] #d), separation(\n)]

DESCRIPTION = 'Attached pictures URLs(with output placeholders #t as image type, #m as

class eyed3.plugins.display.**ObjectsTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['objects', 'gobj']

PARAMETERS = [output(GEOB: [Size: #s bytes] [Type: #t] Description: #d | Filename:

DESCRIPTION = 'Objects (GOBJ)(with output placeholders #s as size, #m as mime type, #d

class eyed3.plugins.display.**PrivatesTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['privates', 'priv']

PARAMETERS = [output(PRIV-Content: #b bytes | Owner: #o), separation(\n)]

DESCRIPTION = 'Privates (APIC) (with output placeholders #c as content, #b as number o

class eyed3.plugins.display.**MusicCDIdTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['music-cd-id', 'mcdi']

DESCRIPTION = 'Music CD Identification'

class eyed3.plugins.display.**TermsOfUseTagPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.TagPattern*

NAMES = ['terms-of-use']

DESCRIPTION = 'Terms of use'

class eyed3.plugins.display.**FunctionPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.ComplexPattern*

TYPE = 'function'

class eyed3.plugins.display.**FunctionFormatPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['format']

PARAMETERS = [text, bold(None), color(None)]

DESCRIPTION = 'Formats text bold and colored (grey, red, green, yellow, blue, magenta,

class eyed3.plugins.display.**FunctionNumberPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['num', 'number-format']

PARAMETERS = [number, digits]

DESCRIPTION = 'Appends leading zeros'

class eyed3.plugins.display.**FunctionFilenamePattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['filename', 'fn']

PARAMETERS = [basename(None)]

DESCRIPTION = 'File name'

class eyed3.plugins.display.**FunctionFilesizePattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['filesize']

DESCRIPTION = 'Size of file'

class eyed3.plugins.display.**FunctionTagVersionPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['tag-version']

DESCRIPTION = 'Tag version'

class eyed3.plugins.display.**FunctionLengthPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['length']

DESCRIPTION = 'Length of aufio file'

class eyed3.plugins.display.**FunctionMPEGVersionPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['mpeg-version']

PARAMETERS = [output(MPEG#v\, Layer #l)]

DESCRIPTION = 'MPEG version (with output placeholders #v as version & #l as layer)'

class eyed3.plugins.display.**FunctionBitRatePattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['bit-rate']

DESCRIPTION = 'Bit rate of aufio file'

class eyed3.plugins.display.**FunctionSampleFrequencePattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['sample-freq']

DESCRIPTION = 'Sample frequence of aufio file in Hz'

class eyed3.plugins.display.**FunctionAudioModePattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['audio-mode']

DESCRIPTION = 'Mode of aufio file: mono/stereo'

class eyed3.plugins.display.**FunctionNotEmptyPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*, *eyed3.plugins.display.PlaceholderUsagePattern*

NAMES = ['not-empty']

PARAMETERS = [text, output(#t), empty(None)]

DESCRIPTION = 'If condition is not empty (with output placeholder #t as text)'

class eyed3.plugins.display.**FunctionRepeatPattern**(*name*, *parameters*)
Bases: *eyed3.plugins.display.FunctionPattern*

NAMES = ['repeat']

PARAMETERS = [text, count]

DESCRIPTION = 'Repeats text'

class eyed3.plugins.display.**DisplayPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['display']

SUMMARY = 'Tag Display'

DESCRIPTION = '\nPrints specific tag information.\n'

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

handleFile(*f*, **args*, ***kwargs*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

exception eyed3.plugins.display.**DisplayException**(*message*)
Bases: `Exception`

property message

exception eyed3.plugins.display.**PatternCompileException**(*message*)
Bases: `Exception`

property message

eyed3.plugins.fixup module

eyed3.plugins.fixup.**dirDate**(*d*)

class eyed3.plugins.fixup.**FixupPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['fixup']

SUMMARY = 'Performs various checks and fixes to directories of audio files.'

DESCRIPTION = '\nOperates on directories at a time, fixing each as a unit (album,\ncom

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

handleFile(*f*, **args*, ***kwargs*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDirectory(*directory*, *_*)
Override to make use of `self._file_cache`. By default the list is cleared, subclasses should consider doing the same otherwise every AudioFile will be cached.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.plugins.genres module

class eyed3.plugins.genres.**GenreListPlugin**(*arg_parser*)
Bases: *eyed3.plugins.Plugin*

SUMMARY = 'Display the full list of standard ID3 genres.'

DESCRIPTION = 'ID3 v1 defined a list of genres and mapped them to to numeric values so

NAMES = ['genres']

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

eyed3.plugins.itunes module

class eyed3.plugins.itunes.**Podcast**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['itunes-podcast']

SUMMARY = 'Adds (or removes) the tags necessary for Apple iTunes to identify the file

handleFile(*f*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

eyed3.plugins.lameinfo module

class eyed3.plugins.lameinfo.**LameInfoPlugin**(*arg_parser*, *cache_files=False*, *track_images=False*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['lameinfo', 'xing']

SUMMARY = 'Outputs lame header (if one exists) for file.'

DESCRIPTION = "The 'lame' (or xing) header provides extra information about the mp3 tha

printHeader(*file_path*)

handleFile(*f*, *_, **__)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

eyed3.plugins.nfo module

class eyed3.plugins.nfo.**NfoPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['nfo']

SUMMARY = 'Create NFO files for each directory scanned.'

DESCRIPTION = 'Each directory scanned is treated as an album and a `NFO <http://en.wik

handleFile(*f*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.plugins.pymod module

class eyed3.plugins.pymod.**PyModulePlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

SUMMARY = 'Imports a Python module file and calls its functions for the the various pl

DESCRIPTION = "\nIf no module if provided (see -m/--module) a file named eyeD3mod.py i

NAMES = ['pymod']

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

handleFile(*f*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDirectory(*d*, *_*)
Override to make use of `self._file_cache`. By default the list is cleared, subclasses should consider doing the same otherwise every AudioFile will be cached.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.plugins.stats module

class eyed3.plugins.stats.**Rule**
Bases: object

test(*path*, *audio_file*)

class eyed3.plugins.stats.**Id3TagRules**
Bases: *eyed3.plugins.stats.Rule*

test(*path*, *audio_file*)

class eyed3.plugins.stats.**BitrateRule**
Bases: *eyed3.plugins.stats.Rule*

BITRATE_DEDUCTIONS = [(128, -20), (192, -10)]

test(*path*, *audio_file*)

class eyed3.plugins.stats.**FileRule**
Bases: *eyed3.plugins.stats.Rule*

test(*path*, *audio_file*)

class eyed3.plugins.stats.**ArtworkRule**
Bases: *eyed3.plugins.stats.Rule*

test(*path*, *audio_file*)

class eyed3.plugins.stats.**Id3FrameRules**
Bases: *eyed3.plugins.stats.Rule*

test(*path*, *audio_file*)

class eyed3.plugins.stats.**Stat**(**args*, ***kwargs*)
Bases: collections.Counter

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

TOTAL = 'total'

compute(*file*, *audio_file*)

report()

percent(*key*)

class eyed3.plugins.stats.**AudioStat**(**args*, ***kwargs*)
Bases: *eyed3.plugins.stats.Stat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

compute(*audio_file*)

class eyed3.plugins.stats.**FileCounterStat**
Bases: *eyed3.plugins.stats.Stat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

SUPPORTED_AUDIO = 'audio'

UNSUPPORTED_AUDIO = 'audio (unsupported)'

HIDDEN_FILES = 'hidden'

OTHER_FILES = 'other'

class eyed3.plugins.stats.**MimeTypeStat**(**args*, ***kwargs*)
Bases: *eyed3.plugins.stats.Stat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**Id3VersionCounter**
Bases: *eyed3.plugins.stats.AudioStat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**Id3FrameCounter**(**args*, ***kwargs*)
Bases: *eyed3.plugins.stats.AudioStat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**BitrateCounter**
Bases: *eyed3.plugins.stats.AudioStat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**RuleViolationStat**(**args*, ***kwargs*)
Bases: *eyed3.plugins.stats.Stat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**Id3ImageTypeCounter**
Bases: *eyed3.plugins.stats.AudioStat*

Create a new, empty Counter object. And if given, count elements from an input iterable. Or, initialize the count from another mapping of elements to their counts.

```
>>> c = Counter()                           # a new, empty counter
>>> c = Counter('gallahad')                 # a new counter from an iterable
>>> c = Counter({'a': 4, 'b': 2})           # a new counter from a mapping
>>> c = Counter(a=4, b=2)                   # a new counter from keyword args
```

class eyed3.plugins.stats.**StatisticsPlugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['stats']

SUMMARY = 'Computes statistics for all audio files scanned.'

handleFile(*path*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.plugins.xep_118 module

class eyed3.plugins.xep_118.**Xep118Plugin**(*arg_parser*)
Bases: *eyed3.plugins.LoaderPlugin*

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

NAMES = ['xep-118']

SUMMARY = 'Outputs all tags in XEP-118 XML format. (see: http://xmpp.org/extensions/x

handleFile(*f*, **args*, ***kwargs*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

getXML(*audio_file*)

Module contents

`eyed3.plugins.`**load**(*name=None*, *reload=False*, *paths=None*)
Returns the eyed3.plugins.Plugin *class* identified by `name`. If `name` is `None` then the full list of plugins is returned. Once a plugin is loaded its class object is cached, and future calls to this function will returned the cached version. Use `reload=True` to refresh the cache.

class `eyed3.plugins.`**Plugin**(*arg_parser*)
Bases: *eyed3.utils.FileHandler*

Base class for all eyeD3 plugins

SUMMARY = 'eyeD3 plugin'

DESCRIPTION = ''

NAMES = []

start(*args*, *config*)
Called after command line parsing but before any paths are processed. The `self.args` argument (the parsed command line) and `self.config` (the user config, if any) is set here.

handleFile(*f*)
Called for each file walked. The file `f` is the full path and the return value is ignored. If the walk should abort the method should raise a `StopIteration` exception.

handleDone()
Called after all file/directory processing; before program exit. The return value is passed to sys.exit (None results in 0).

class `eyed3.plugins.`**LoaderPlugin**(*arg_parser*, *cache_files=False*, *track_images=False*)
Bases: *eyed3.plugins.Plugin*

A base class that provides auto loading of audio files

Constructor. If `cache_files` is True (off by default) then each AudioFile is appended to `_file_cache` during `handleFile` and the list is cleared by `handleDirectory`.

handleFile(*f*, **args*, ***kwargs*)
Loads `f` and sets `self.audio_file` to an instance of *eyed3.core.AudioFile* or `None` if an error occurred or the file is not a recognized type.

The `*args` and `**kwargs` are passed to *eyed3.core.load()*.

handleDirectory(*d*, *_*)
Override to make use of `self._file_cache`. By default the list is cleared, subclasses should consider doing the same otherwise every AudioFile will be cached.

handleDone()
If no audio files were loaded this simply prints 'Nothing to do'.

eyed3.utils package

Submodules

eyed3.utils.art module

eyed3.utils.art.**FRONT_COVER = 'FRONT_COVER'**
Album front cover.

eyed3.utils.art.**BACK_COVER = 'BACK_COVER'**
Album back cover.

eyed3.utils.art.**MISC_COVER = 'MISC_COVER'**
Other part of the album cover; liner notes, gate-fold, etc.

eyed3.utils.art.**LOGO = 'LOGO'**
Artist/band logo.

eyed3.utils.art.**ARTIST = 'ARTIST'**
Artist/band images.

eyed3.utils.art.**LIVE = 'LIVE'**
Artist/band images.

eyed3.utils.art.**FILENAMES = {'ARTIST': ['artist*'], 'BACK_COVER': ['cover-back', 'back', 'c**
A mapping of art types to lists of filename patterns (excluding file extension): type -> [file_pattern, ..].

eyed3.utils.art.**TO_ID3_ART_TYPES = {'ARTIST': [7, 8, 10], 'BACK_COVER': [4], 'FRONT_COVER'**
A mapping of art types to ID3 APIC (image) types: type -> [apic_type, ..]

eyed3.utils.art.**FROM_ID3_ART_TYPES = {0: 'FRONT_COVER', 1: 'FRONT_COVER', 3: 'FRONT_COVE**
A mapping of ID3 art types to eyeD3 art types; the opposite of TO_ID3_ART_TYPES.

eyed3.utils.art.**matchArtFile**(*filename*)
Compares `filename` (case insensitive) with lists of common art file names and returns the type of art that was matched, or None if no types were matched.

eyed3.utils.art.**getArtFromTag**(*tag*, *type_=None*)
Returns a list of eyed3.id3.frames.ImageFrame objects matching `type_`, all if `type_` is None, or empty if tag does not contain art.

eyed3.utils.binfuncs module

eyed3.utils.binfuncs.**bytes2bin**(*bites*, *sz=8*)
Accepts a string of `bytes` (chars) and returns an array of bits representing the bytes in big endian byte order. An optional max `sz` for each byte (default 8 bits/byte) which can be used to mask out higher bits.

eyed3.utils.binfuncs.**bin2bytes**(*x*)
Convert an array of bits (MSB first) into a string of characters.

eyed3.utils.binfuncs.**bin2dec**(*x*)
Convert `x`, an array of "bits" (MSB first), to it's decimal value.

eyed3.utils.binfuncs.**bytes2dec**(*bytes*, *sz=8*)

eyed3.utils.binfuncs.**dec2bin**(*n*, *p=1*)
Convert a decimal value `n` to an array of bits (MSB first). Optionally, pad the overall size to `p` bits.

eyed3.utils.binfuncs.**dec2bytes**(*n*, *p=1*)

eyed3.utils.binfuncs.**bin2synchsafe**(*x*)
Convert x, a list of bits (MSB first), to a synch safe list of bits. (section 6.2 of the ID3 2.4 spec).

eyed3.utils.console module

class eyed3.utils.console.**AnsiCodes**(*codes*)
Bases: object

classmethod init(*allow_colors*)

class eyed3.utils.console.**AnsiFore**
Bases: object

GREY = 30

RED = 31

GREEN = 32

YELLOW = 33

BLUE = 34

MAGENTA = 35

CYAN = 36

WHITE = 37

RESET = 39

class eyed3.utils.console.**AnsiBack**
Bases: object

GREY = 40

RED = 41

GREEN = 42

YELLOW = 43

BLUE = 44

MAGENTA = 45

CYAN = 46

WHITE = 47

RESET = 49

class eyed3.utils.console.**AnsiStyle**
Bases: object

RESET_ALL = 0

BRIGHT = 1

RESET_BRIGHT = 22

DIM = 2

RESET_DIM = 22

ITALICS = 3

RESET_ITALICS = 23

UNDERLINE = 4

RESET_UNDERLINE = 24

BLINK_SLOW = 5

RESET_BLINK_SLOW = 25

BLINK_FAST = 6

RESET_BLINK_FAST = 26

INVERSE = 7

RESET_INVERSE = 27

STRIKE_THRU = 9

RESET_STRIKE_THRU = 29

eyed3.utils.console.**ERROR_COLOR**()

eyed3.utils.console.**WARNING_COLOR**()

eyed3.utils.console.**HEADER_COLOR**()

class eyed3.utils.console.**Spinner**(*msg*, *file=None*, *step=1*, *chars=None*, *use_unicode=True*, *print_done=True*)

Bases: object

A class to display a spinner in the terminal.

It is designed to be used with the *with* statement:

```
with Spinner("Reticulating splines", "green") as s:
    for item in enumerate(items):
        s.next()
```

class eyed3.utils.console.**ProgressBar**(*total_or_items*, *file=None*)

Bases: object

A class to display a progress bar in the terminal.

It is designed to be used either with the *with* statement:

```
with ProgressBar(len(items)) as bar:
    for item in enumerate(items):
        bar.update()
```

or as a generator:

```
for item in ProgressBar(items):
    item.process()
```

total_or_items [int or sequence] If an int, the number of increments in the process being tracked. If a sequence, the items to iterate over.

file [writable file-like object, optional] The file to write the progress bar to. Defaults to *sys.stdout*. If *file* is not a tty (as determined by calling its *isatty* member, if any), the scrollbar will be completely silent.

next()

update(*value=None*)
Update the progress bar to the given value (out of the total given to the constructor).

classmethod map(*function*, *items*, *multiprocess=False*, *file=None*)
Does a *map* operation while displaying a progress bar with percentage complete.

```
def work(i):
    print(i)

ProgressBar.map(work, range(50))
```

Parameters:

function [function] Function to call for each step

items [sequence] Sequence where each element is a tuple of arguments to pass to *function*.

multiprocess [bool, optional] If *True*, use the *multiprocessing* module to distribute each task to a different processor core.

file [writeable file-like object, optional] The file to write the progress bar to. Defaults to *sys.stdout*. If *file* is not a tty (as determined by calling its *isatty* member, if any), the scrollbar will be completely silent.

eyed3.utils.console.**printMsg**(*s*)

eyed3.utils.console.**printError**(*s*)

eyed3.utils.console.**printWarning**(*s*)

eyed3.utils.console.**printHeader**(*s*)

eyed3.utils.console.**boldText**(*s*, *c=None*)

eyed3.utils.console.**formatText**(*s*, *b=False*, *c=None*)

eyed3.utils.console.**cformat**(*msg*, *fg*, *bg=None*, *styles=None*)
Format `msg` with foreground and optional background. Optional `styles` lists will also be applied. The formatted string is returned.

eyed3.utils.console.**getTtySize**(*fd=<_io.TextIOWrapper name='<stdout>' mode='w' encoding='UTF-8'>*, *check_tty=True*)

eyed3.utils.console.**cprint**(*msg*, *fg*, *bg=None*, *styles=None*, *file=<_io.TextIOWrapper name='<stdout>' mode='w' encoding='UTF-8'>*)
Calls `cformat` and prints the result to output stream `file`.

eyed3.utils.log module

class eyed3.utils.log.**Logger**(*name*)
Bases: `logging.Logger`

Base class for all loggers

Initialize the logger with a name and an optional level.

verbose(*msg*, **args*, ***kwargs*)
Log msg at 'verbose' level, debug < verbose < info

eyed3.utils.log.**getLogger**(*name*)

eyed3.utils.log.**initLogging**()
initialize the default logger with console output

eyed3.utils.prompt module

eyed3.utils.prompt.**DISABLE_PROMPT = None**
Whenever a prompt occurs and this value is not None it can be exit to call sys.exit (see EXIT_STATUS) or raise to throw a RuntimeError, which can be caught if desired.

exception eyed3.utils.prompt.**PromptExit**
Bases: RuntimeError

Raised when DISABLE_PROMPT is 'raise' and prompt is called.

eyed3.utils.prompt.**parseIntList**(*resp*)

eyed3.utils.prompt.**prompt**(*msg*, *default=None*, *required=True*, *type_=<class 'str'>*, *validate=None*, *choices=None*)
Prompt user for imput, the prequest is in msg. If default is not None it will be displayed as the default and returned if not input is entered. The value None is only returned if required is False. The response is passed to type_ for conversion (default is unicode) before being returned. An optional list of valid responses can be provided in choices.

Module contents

eyed3.utils.**os_walk**(*top='.'*, *topdown=True*, *onerror=None*, ***, *follow_symlinks=True*, *dir_fd=None*)
Directory tree generator.

This behaves exactly like walk(), except that it yields a 4-tuple

dirpath, dirnames, filenames, dirfd

dirpath, *dirnames* and *filenames* are identical to walk() output, and *dirfd* is a file descriptor referring to the directory *dirpath*.

The advantage of fwalk() over walk() is that it's safe against symlink races (when follow_symlinks is False).

If dir_fd is not None, it should be a file descriptor open to a directory, and top should be relative; top will then be relative to that directory. (dir_fd is always supported for fwalk.)

Caution: Since fwalk() yields file descriptors, those are only valid until the next iteration step, so you should dup() them if you want to keep them for a longer period.

Example:

import os for root, dirs, files, rootfd in os.fwalk('python/Lib/email'):

print(root, "consumes", end="") print(sum([os.stat(name, dir_fd=rootfd).st_size for name in files]),

end="")

print("bytes in", len(files), "non-directory files") if 'CVS' in dirs:

dirs.remove('CVS') # don't visit CVS directories

eyed3.utils.**os_walk_unpack**(*w*)

eyed3.utils.**guessMimetype**(*filename*, *with_encoding=False*)
Deprecated since version 0.9a2: This will be removed in 1.0. Use eyed3.mimetype.guessMimetype() instead.

eyed3.utils.**walk**(*handler*, *path*, *excludes=None*, *fs_encoding='utf-8'*, *recursive=False*)
A wrapper around os.walk which handles exclusion patterns and multiple path types (str, pathlib.Path, bytes).

class eyed3.utils.**FileHandler**
Bases: object

A handler interface for *eyed3.utils.walk()* callbacks.

handleFile(*f*)
Called for each file walked. The file `f` is the full path and the return value is ignored. If the walk should abort the method should raise a `StopIteration` exception.

handleDirectory(*d*, *files*)
Called for each directory `d` **after** `handleFile` has been called for each file in `files`. `StopIteration` may be raised to halt iteration.

handleDone()
Called when there are no more files to handle.

eyed3.utils.**requireUnicode**(**args*)
Function decorator to enforce str/unicode argument types. `None` is a valid argument value, in all cases, regardless of not being unicode. `*args` Positional arguments may be numeric argument index values (requireUnicode(1, 3) - requires argument 1 and 3 are unicode) or keyword argument names (requireUnicode("title")) or a combination thereof.

eyed3.utils.**requireBytes**(**args*)
Function decorator to enforce byte string argument types.

eyed3.utils.**formatTime**(*seconds*, *total=None*, *short=False*)
Format `seconds` (number of seconds) as a string representation. When `short` is False (the default) the format is:

HH:MM:SS.

Otherwise, the format is exacly 6 characters long and of the form:

1w 3d 2d 4h 1h 5m 1m 4s 15s

If `total` is not None it will also be formatted and appended to the result seperated by ' / '.

eyed3.utils.**formatSize**(*size*, *short=False*)
Format `size` (nuber of bytes) into string format doing KB, MB, or GB conversion where necessary.

When `short` is False (the default) the format is smallest unit of bytes and largest gigabytes; '234 GB'. The short version is 2-4 characters long and of the form

256b 64k 1.1G

eyed3.utils.**formatTimeDelta**(*td*)
Format a timedelta object `td` into a string.

eyed3.utils.**chunkCopy**(*src_fp*, *dest_fp*, *chunk_sz=524288*)
Copy `src_fp` to `dest_fp` in `chunk_sz` byte increments.

class eyed3.utils.**ArgumentParser**(**args*, ***kwargs*)
Bases: `argparse.ArgumentParser`

Subclass of argparse.ArgumentParser that adds version and log level options.

class eyed3.utils.**LoggingAction**(**args*, ***kwargs*)
Bases: `argparse._AppendAction`

eyed3.utils.**datePicker**(*thing*, *prefer_recording_date=False*)
This function returns a date of some sort, amongst all the possible dates (members called release_date, original_release_date, and recording_date of type eyed3.core.Date).

The order of preference is: 1) date of original release 2) date of this versions release 3) the recording date.

Unless `prefer_recording_date` is `True` in which case the order is 3, 1, 2.

`None` will be returned if no dates are available.

eyed3.utils.**makeUniqueFileName**(*file_path*, *uniq=''*)
The `file_path` is the desired file name, and it is returned if the file does not exist. In the case that it already exists the path is adjusted to be unique. First, the `uniq` string is added, and then a couter is used to find a unique name.

eyed3.utils.**b**(*x*, *encoder=None*)
Converts *x* to a bytes string if not already. :param x: The string. :param encoder: Optional codec encoder to perform the conversion. The default is

codecs.latin_1_encode.

Returns The byte string if conversion was needed.

Submodules

eyed3.core module

Basic core types and utilities.

class eyed3.core.**ArtistOrigin**(*city: str*, *state: str*, *country: str*)
Bases: `object`

id3Encode()

eyed3.core.**load**(*path*, *tag_version=None*)
Loads the file identified by `path` and returns a concrete type of *eyed3.core.AudioFile*. If `path` is not a file an `IOError` is raised. `None` is returned when the file type (i.e. mime-type) is not recognized. The following AudioFile types are supported:

- *eyed3.mp3.Mp3AudioFile* - For mp3 audio files.
- *eyed3.id3.TagFile* - For raw ID3 data files.

If `tag_version` is not None (the default) only a specific version of metadata is loaded. This value must be a version constant specific to the eventual format of the metadata.

class eyed3.core.**AudioInfo**
Bases: `object`

A base container for common audio details.

time_secs = 0.0

size_bytes = 0

class eyed3.core.**Tag**(*title=None*, *artist=None*, *album=None*, *album_artist=None*, *track_num=None*)
Bases: `object`

An abstract interface for audio tag (meta) data (e.g. artist, title, etc.)

read_only = False

property title

property artist

property album

property album_artist

property track_num
Track number property. Must return a 2-tuple of (track-number, total-number-of-tracks). Either tuple value may be `None`.

class eyed3.core.**AudioFile**(*path*)
Bases: object

Abstract base class for audio file types (AudioInfo + Tag)

Construct with a path and invoke _read. All other members are set to None.

rename(*name*, *fsencoding='utf-8'*, *preserve_file_time=False*)
Rename the file to name. The encoding used for the file name is eyed3.LOCAL_FS_ENCODING unless overridden by fsencoding. Note, if the target file already exists, or the full path contains non-existent directories the operation will fail with IOError. File times are not modified when preserve_file_time is True, False is the default.

property info
Returns a concrete implemenation of *eyed3.core.AudioInfo*

property tag
Returns a concrete implemenation of *eyed3.core.Tag*

property path
The absolute path of this file.

class eyed3.core.**Date**(*year*, *month=None*, *day=None*, *hour=None*, *minute=None*, *second=None*)
Bases: object

A class for representing a date and time (optional). This class differs from datetime.datetime in that the default values for month, day, hour, minute, and second is None and not 'January 1, 00:00:00'. This allows for an object that is simply 1987, and not January 1 12AM, for example. But when more resolution is required those vales can be set as well.

TIME_STAMP_FORMATS = ['%Y', '%Y-%m', '%Y-%m-%d', '%Y-%m-%dT%H', '%Y-%m-%dT%H:%M', '%Y-
Valid time stamp formats per ISO 8601 and used by *strptime*.

property year

property month

property day

property hour

property minute

property second

static parse(*s*)
Parses date strings that conform to ISO-8601.

eyed3.core.**parseError**(*ex*)
A function that is invoked when non-fatal parse, format, etc. errors occur. In most cases the invalid values will be ignored or possibly fixed. This function simply logs the error.

eyed3.main module

eyed3.main.**main**(*args*, *config*)

eyed3.main.**profileMain**(*args*, *config*)
This is the main function for profiling http://code.google.com/appengine/kb/commontasks.html#profiling

eyed3.main.**setFileScannerOpts**(*arg_parser*, *default_recursive=False*, *paths_metavar='PATH'*, *paths_help='Files or directory paths'*)

eyed3.main.**makeCmdLineParser**(*subparser=None*)

eyed3.main.**parseCommandLine**(*cmd_line_args=None*)

eyed3.mimetype module

eyed3.mimetype.**guessMimetype**(*filename*)
Return the mime-type for *filename*.

class eyed3.mimetype.**Mp2x**
Bases: filetype.types.base.Type

Implements the MP2.x audio type matcher.

MIME = 'audio/mpeg'

EXTENSION = 'mp3'

match(*buf*)

class eyed3.mimetype.**Mp3Invalids**
Bases: filetype.types.base.Type

Implements a MP3 audio type matcher this is odd or/corrupt mp3.

MIME = 'audio/mpeg'

EXTENSION = 'mp3'

match(*buf*)

class eyed3.mimetype.**Id3Tag**
Bases: filetype.types.base.Type

Implements a MP3 audio type matcher this is odd or/corrupt mp3.

MIME = 'application/x-id3'

EXTENSION = 'id3'

match(*buf*)

class eyed3.mimetype.**Id3TagExt**
Bases: *eyed3.mimetype.Id3Tag*

EXTENSION = 'tag'

class eyed3.mimetype.**M3u**
Bases: filetype.types.base.Type

Implements the m3u playlist matcher.

MIME = 'audio/x-mpegurl'

EXTENSION = 'm3u'

match(*buf*)

Module contents

exception eyed3.**Error**(**args*)
Bases: Exception

Base exception type for all eyed3 errors.

eyed3.**load**(*path*, *tag_version=None*)
Loads the file identified by path and returns a concrete type of *eyed3.core.AudioFile*. If path is not a file an IOError is raised. None is returned when the file type (i.e. mime-type) is not recognized. The following AudioFile types are supported:

- *eyed3.mp3.Mp3AudioFile* - For mp3 audio files.
- *eyed3.id3.TagFile* - For raw ID3 data files.

If tag_version is not None (the default) only a specific version of metadata is loaded. This value must be a version constant specific to the eventual format of the metadata.

4.2.4 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

Types of Contributions

Report Bugs

Report bugs at https://github.com/nicfit/eyeD3/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

Write Documentation

eyeD3 could always use more documentation, whether as part of the official eyeD3 docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/nicfit/eyeD3/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Get Started!

Ready to contribute? Here's how to set up eyeD3 for local development.

1. Fork the *eyeD3* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/eyeD3.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv eyed3
$ cd eyed3/
$ python setup.py develop
```

4. Install all dependencies for development and extra packages. For full test coverage you'll also need some test data.:

```
$ for reqs in $(ls requirements/*.txt); do pip install -U -r $reqs ; done
$ make test-data
```

5. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

6. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ make lint
$ make test
$ make test-all    # Optional, requires multiple versions of Python
```

To get flake8 and tox, just pip install them into your virtualenv.

7. Commit your changes and push your branch to GitHub.:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

8. Submit a pull request through the GitHub website.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, and 3.3, 3.4, 3.5, and for PyPy. Check https://travis-ci.org/nicfit/eyeD3/pulls and make sure that the tests pass for all supported Python versions.

4.2.5 Authors

eyeD3 is written and maintained by:

- Travis Shirk <travis@pobox.com>

and has been contributed to by (ordered by date of first contribution):

- Ryan Finnie <ryan@finnie.org>
- Henning Kiel <henning.kiel@rwth-aachen.de>
- Knight Walker <kwalker@kobran.org>
- Todd Zullinger <tmz@pobox.com>
- Aaron VonderHaar <avh4@users.sourceforge.net>
- Alexander Thomas <dr-lex@dr-lex.34sp.com>
- Michael Schout <mschout@gkg.net>
- Renaud Saint-Gratien <rsg@nerim.net>
- David Grant <davidgrant@gmail.com>
- Gergan Penkov <gergan@gmail.com>
- Stephen Fairchild <sfairchild@bethere.co.uk>
- Ville Skyttä <ville.skytta@iki.fi>
- Ben Isaacs <me@ben-xo.com>
- Neil Schemenauer <nas@arctrix.com>
- Otávio Pontes <otaviobp@gmail.com>

- Nathaniel Clark <nate@misrule.us>
- Hans Meine <hmeine@users.noreply.github.com>
- Hans Petter Jansson <hpj@copyleft.no>
- Sebastian Patschorke <sludgefeast@users.noreply.github.com>
- Bouke Versteegh <info@boukeversteegh.nl>
- mafro <github@mafro.net>
- Gaetano Guerriero <x.guerriero@tin.it>
- Grun Seid <grunseid@gmail.com>
- pyup-bot <github-bot@pyup.io>
- Chris Newton <redshodan@gmail.com>
- deoren@users.noreply.github.com
- chamatht@gmail.com
- Mic92@users.noreply.github.com
- gabrieldiegoteixeira@gmail.com
- guillaume.web@gmail.com
- tim.gates@iress.com
- zhumumu@gmail.com

4.2.6 Release History

v0.9 (2020-01-01) : Favorite Thing

Major Changes

- Dropped support for Python versions 2.7, 3.4, and 3.5.
- File scanning is no longer recursive by default; use *-r / –recursive*.
- Default log-level changed from WARNING to ERROR.

New

- Mime-type detection uses filetype.py (libmagic no longer required)
- setFileScannerOpts function accepts *default_recursive* option.
- A new *jsontag* plugin for converting tags to JSON.
- A new *extract* plugin for extracting tags from media.
- A new *yamltag* plugin for converting tags to YAML.
- A new *mimetypes* plugin for listing file mime-types / measuring performance
- Original artist support (TOPE frame, –orig-artist)
- Added support for Python 3.8 and pypy3.

Changes

- Log warning when ID3 v1.x text truncation occurs. Fixes #299.
- Accept (invalid) date strings for the form YYYYMMDD. Fixes #379
- Adjust replay gain correctly for lame >= 3.95.1 headers.
- Added -r/–recursive argument. eyeD3 is no longer recursive by default (#378)
- Regenerated grako parser.
- New ValueError for _setNum when unknown type/values are passed.
- Moved src/* to top-level repo directory.

Fix

- PRIV data type checking, fixed examples, etc.
- Use tox for *make test*
- ID3 v2.3 to v2.4 date conversion.
- Match mp3 mime-types against all possible mime-types. Specifically, application/x-font-gdos. Fixes #338
- Fix simple typo: titel -> title. <tim.gates@iress.com>
- Fixed: load the right config file in arguments. <zhumumu@gmail.com>
- Fix issue tracker link. Fixes #333.
- Fixed art plugin when *pylast* is not installed.
- Unbound variable for track num/total. Fixes #327.
- Fixed MP3 header search to not false match on BOMs.
- Honor APIC text encoding when description is “”. #200.
- Fixed bug with improper types when re-rendering unique file ID. (#324) <gabrieldiegoteixeira@gmail.com>
- UFID fixes, update (#325) <gabrieldiegoteixeira@gmail.com>

Other

- Deprecation of eyed3.utils.guessMimeType
- Removed ipdb from dev requirements

v0.8.12 (2019-12-27)

Changes

- Accept (invalid) date strings for the form YYYYMMDD. Fixes #379

Other

- Test with py38

v0.8.11 (2019-11-09)

Fix

- ID3 v2.3 to v2.4 date conversion.
- Match mp3 mime-types against all possible mime-types. Specifically, application/x-font-gdos. Fixes #338

v0.8.10 (2019-03-07) : Apples

New

- Log warning when ID3 v1.x text truncation occurs. Fixes #299.

Fix

- Honor APIC text encoding when description is “”. #200.
- Fixed bug with improper types when re-rendering unique file ID. (#324) <gabrieldiegoteixeira@gmail.com>

v0.8.9 (2019-01-12) : Descent Into...

Changes

- Fixup plugin: -t changed to –type.
- Pin pathlib to latest version 1.0.1 (#304) <github-bot@pyup.io>

Fix

- Force no-color output when stdout is not a terminal (#297) <gaetano.guerriero@gmx.com>
- Requirements.txt: pathlib is only needed for older python versions (#284) <Mic92@users.noreply.github.com>
- Art plugin: Pin pylast to 2.x to preserve Python2 support.

v0.8.8 (2018-11-28) : In Ruins

New

- Follow symlink directories. Fixes #224

Changes

- Eyed3.core.AudioInfo *time_secs* is now a float and non-lossy. Fixes #210
- Removed Python 3.3 support.

Fix

- Better type handling during TLEN [fixup plugin].
- Don't tweak logging by default, only thru *main*. Fixes #243

Other

- Added a separate example for Windows (–add-image <url>) [Addresses the issue #219] (#220) <chamatht@gmail.com>

v0.8.7 (2018-06-11) : Aeon

Fix

- Only use os.fwalk where supported.

v0.8.6 (2018-05-27) : Robot Man

New

- Art plugin can now download album covers from last.fm.

Changes

- Use os.fwalk for its better performance (esp. >= py37) Fixes #166
- TagTemplate *path_friendly* is now a string, namely the delimiter to use.

Fix

- Classic plugin: –write-image will work with –quiet. Fixes #188
- Multiple fixes for display plugin %images% replacements. Fixes #176
- Allow –remove-* options to work when there are no tags. Fixes #183

v0.8.5 (2018-03-27) : 30$ Bag

New

- Mp3AudioFile.initTag now returns the new tag.
- Eyed3.core.EP_MAX_SIZE_HINT.
- Added docs for install devel dependencies and test data.

Changes

- Similarly to TextFrame, fallback to latin1 for invalid encodings.
- Removed paver as a dep.
- Removed fabfile and mkenv.
- Clean pytest_cache.
- Nicfit.py cc update.

Fix

- Handle missing *fcntl* on Windows. Fixes #135.
- In addition to None, "" will now clear dates.
- Update index.rst to reflect the code is in a Git repo, not Mercurial (#164) <deoren@users.noreply.github.com>

Other

- Update pytest from 3.2.2 to 3.5.0 (#175) <github-bot@pyup.io>
- Update twine from 1.9.1 to 1.11.0 (#173) <github-bot@pyup.io>
- Update sphinx from 1.6.5 to 1.7.2 (#174) <github-bot@pyup.io>
- Update sphinxcontrib-paverutils from 1.16.0 to 1.17.0 (#172) <github- bot@pyup.io>
- Update pytest-runner from 3.0 to 4.2 (#171) <github-bot@pyup.io>
- Update nicfit.py from 0.7 to 0.8 (#161) <github-bot@pyup.io>
- Update ipdb from 0.10.3 to 0.11 (#159) <github-bot@pyup.io>
- Update factory-boy from 2.9.2 to 2.10.0 (#150) <github-bot@pyup.io>
- Update pyaml from 17.10.0 to 17.12.1 (#138) <github-bot@pyup.io>
- Update python-magic to 0.4.15 (#130) <github-bot@pyup.io>
- Update pip-tools from 1.10.1 to 1.11.0 (#129) <github-bot@pyup.io>
- Update check-manifest from 0.35 to 0.36 (#125) <github-bot@pyup.io>

v0.8.4 (2017-11-17) : The Cold Vein

New

- Composer (TCOM) support (#123)
- Check for version incompatibilities during version changes.

Changes

- More forgiving of invalid text encoding identifiers (fixes #101)
- More forgiving of bad Unicode in text frames (fixes #105)
- EyeD3 cmd line helper turned not session-scoped fixture.
- Only warn about missing grako when the plugin is used. Fixes #115.

Fix

- Fix python3 setup when system encoding is not utf-8 (#120) <x.guerriero@tin.it>
- Fix bad frames detection in stats plugin for python3 (#113) <x.guerriero@tin.it>
- Script exits with 0 status when called with –version/–help (#109) <x.guerriero@tin.it>
- Help pymagic with poorly encoded filenames.
- [display plugin] Handle comments.
- [display plugin] Handle internal exception types. Fixes #118.
- IOError (nor OSError) have a message attr.

Other

- Set theme jekyll-theme-slate.
- Update pytest to 3.2.5 (#122) <github-bot@pyup.io>
- Update pytest-runner to 3.0 (#108) <github-bot@pyup.io>
- Update sphinx to 1.6.5 (#106) <github-bot@pyup.io>
- Update flake8 to 3.5.0 (#107) <github-bot@pyup.io>

v0.8.3 (2017-10-22) : So Alone

Fix

- Reload and process after tag removals, fixes #102. (PR #103)
- Display incorrectly encoded strings (usually filenames)

Other

- Make the classic output span the actual width of the tty so you can see the actual path with a long file name. (#92) <redshodan@gmail.com>

v0.8.2 (2017-09-23) : Standing At the Station

New

- Pypy and pypy3 support.

Changes

- 'nose' is no longer used/required for testing.

Fix

- Fix for Unicode paths when using Python2. Fixes #56.

v0.8.1 (2017-08-26) : I Can't Talk To You

New

- `make pkg-test-data` target.
- Sample mime-type tests.

Fix

- Added `python-magic` as a dependency for reliable mime-type detection. Fixes #61
- Add pathlib to requirements. Fixes #43.
- [doc] Fixed github URL.

v0.8 (2017-05-13) : I Don't Know My Name

> **Warning:** This release is **NOT** API compatible with 0.7.x. The majority of the command line interface has been preserved although many options have either changed or been removed. Additionally, support for Python 2.6 has been dropped.

New

- Python 3 support (version 2.7 and >= 3.3 supported)
- The Display plugin (-P/–plugin display) enables complete control over tag output. Requires `grako`. If using pip, `pip install eyeD3[display]`. Contributed by Sebastian Patschorke.

- Genre.parse(id3_std=False) (and –non-std-genres) to disable genre # mapping.
- eyed3.load accept pathlib.Path arguments.
- eyed3.core.AudioFile accept pathlib.Path arguments.
- eyed3.utils.walk accept pathlib.Path arguments.
- New manual page. Contributed by Gaetano Guerriero
- `make test-data`

Changes

- Project home from to GitHub: https://github.com/nicfit/eyeD3

Fix

- Lang fixes, and no longer coerce invalids to eng.

Other

- Moved to pytest, although unittest not yet purged.

0.7.11 - 03.12.2017 (Evergreen)

New Features:

- Repo and issue tracker moved to GitHub: https://github.com/nicfit/eyeD3

Bug Fixes:

- 'NoneType' object has no attribute 'year'
- Multiple date related fixes.
- Allow superfluous –no-tagging-ttme-frame option for backward compatibility.
- The –version option now prints a short, version-only, message.
- Allow –year option for backward compatibility. Converts to –release-year.
- Fixes for –user-text-frame with multiple colons and similar fixes.
- ID3 v1.1 encoding fixes.

0.7.10 - 12.10.2016 (Hollow)

Bug Fixes:

- Missing import
- Fix the rendering of default constructed id3.TagHeader
- Fixed Tag.frameiter

0.7.9 - 11.27.2015 (Collapse/Failure)

New Features:

- process files and directories in a sorted fashion. <Hans-Peter Jansen>
- display the ellipsis file name and path, and the file size right justified in printHeader. <Hans-Peter Jansen>
- stating to be unable to find a valid mp3 frame without a hint, where this happened is rather unfortunate. I noticed this from using eyed3.load() calls. <Hans-Peter Jansen>
- [fixup plugin] - Better compilation support.

Bug Fixes:

- Fixed missing 'math' import.
- Replaced invalid Unicode.
- Disabled ANSI codes on Windows
- More friendly logging (as a module)

0.7.8 - 05.25.2015 (Chartsengrafs)

New Features:

- [pymod plugin] – A more procedural plugin interface with modules.
- [art plugin] – Extract tag art to image files, or add images to tags.
- eyed3.utils.art - High level tag art API
- eyed3.id3.frames.ImageFrame.makeFileName produces the file extension .jpg instead of .jpeg for JPEG mime-types.
- Added eyed3.utils.makeUniqueFileName for better reuse.
- [statistics plugin] – Less score deduction for lower bit rates.
- Split example plugins module into discrete plugin modules.
- [fixup plugin] – Added –fix-case for applying `title()` to names
- [fixup plugin] – Detects and optionally removes files determined to be cruft.
- eyed3.id3.Tag – Added `frameiter` method for iterating over tag frames.
- Added optional `preserve_file_time` argument to eyed3.id3.Tag.remove.
- Removed python-magic dependency, it not longer offers any value (AFAICT).

Bug Fixes:

- ashing on –remove-frame PRIV
- rse lameinfo even if crc16 is not correct
- po in docs/installation.rst
- Request to update the GPL License in source files
- Fixes to eyed3.id3.tag.TagTemplate when expanding empty dates.
- eyed3.plugins.Plugin.handleDone return code is not actually used.
- [classic plugin] – Fixed ID3v1 –verbose bug.

- [fixup plugin] – Better date handling, album type, and many bug fixes.

0.7.5 - 09.06.2014 (Nerve Endings)

New Features:

- Support for album artist info. By Cyril Roelandt <tipecaml@gmail.com>
- [fixup plugin] – Custom patterns for file/directory renaming. By Matt Black <https://bitbucket.org/mafrosis>
- API providing simple prompts for plugins to use.
- API and TXXX frame mappings for album type (e.g. various, album, demo, etc.) and artist origin (i.e. where the artist/band is from).
- Lower cases ANSI codes and other console fixes.
- Added the ability to set (remove) tag padding. See *eyeD3 –max-padding* option. By Hans Meine.
- Tag class contains read_only attribute than can be set to `True` to disable the `save` method.
- [classic plugin] – Added `--track-offset` for incrementing/decrementing the track number.
- [fixup plugin] – Check for and fix cover art files.

Bug Fixes:

- Build from pypi when `paver` is not available.
- Disable ANSI color codes when TERM == "dumb"
- Locking around libmagic.
- Work around for zero-padded utf16 strings.
- Safer tempfile usage.
- Better default v1.x genre.

0.7.3 - 07.12.2013 (Harder They Fall)

Bug fixes:

- Allow setup.py to run with having `paver` installed.
- [statistics plugin] Don't crash when 0 files are processed.

0.7.2 - 07.06.2013 (Nevertheless)

New Features:

- Python 2.6 is now supported if `argparse` and `ordereddict` dependencies are installed. Thanks to Bouke Versteegh for much of this.
- More support and bug fixes for ID3 chapters and table-of-contents.
- [classic plugin] `-d/-D` options for setting tag disc number and disc set total.
- Frames are always written in sorted order, so if a tag is rewritten with no values changed the file's checksum remains the same.

- Documentation and examples are now included in source distribution.
- [classic plugin] Removed `-p` for setting publisher since using it when `-P` is intended is destructive.
- [classic plugin] Supports `--no-color` to disable color output. Note, this happens automatically if the output streams is not a TTY.
- `Tag.save` supports preserving the file modification time; and option added to classic plugin.
- [statistics plgin] Added rules for "lint-like" checking of a collection. The rules are not yet configurable.
- ERROR is now the default log level.

Bug fixes:

- Various fixes for PRIV frames, error handling, etc. from Bouke Versteegh
- Convert '/' to '-' in TagTemplate names (i.e. –rename)
- Drop TSIZ frames when converting to ID3 v2.4
- ID3 tag padding size now set correctly.
- Fixes for Unicode paths.
- License clarification in pkg-info.
- The `-b` setup.py argument is now properly supported.
- Magic module *hasattr* fix.
- More robust handling of bogus play count values.
- More robust handling of bogus date values.
- Proper unicode handling of APIC descriptions.
- Proper use of argparse.ArgumentTypeError
- Allow TCMP frames when parsing.
- Accept more invalid frame types (iTunes)
- Documentation fixes.
- Fix for bash completion script.
- Fix for certain mp3 bit rate and play time computations.

0.7.1 - 11.25.2012 (Feel It)

New Features:

- Support for ID3 chapters and table-of-contents frames (i.e.CHAP and CTOC).
- A new plugin for toggling the state of iTunes podcast files. In other words, PCST and WFED support. Additionally, the Apple "extensions" frames TKWD, TDES, and TGID are supported. Run `eyeD3 -P itunes-podcast --help` for more info.
- Native frame type for POPM (Popularity meter). See the *eyed3.id3.tag.Tag.popularities()* accessor method.
- Plugins can deal with traversed directories instead of only file-by-file. Also, *eyed3.plugins.LoaderPlugin* can optionally cache the loaded audio file objects for each callback to `handleDirectory`.

- [classic plugin] New –remove-frame option.
- [statistics plugin] More accurate values and easier to extend.

Bug fixes:

- Fixed a very old bug where certain values of 0 would be written to the tag as '' instead of 'x00'.
- Don't crash on malformed (invalid) UFID frames.
- Handle timestamps that are terminated with 'Z' to show the time is UTC.
- Conversions between ID3 v2.3 and v2.4 date frames fixed.
- [classic plugin] Use the system text encoding (locale) when converting lyrics files to Unicode.

0.7.0 - 11.15.2012 (Be Quiet and Drive)

> **Warning:** This release is **NOT** API compatible with 0.6.x. The majority of the command line interface has been preserved although many options have either changed or been removed.

New Features:

- Command line script `eyeD3` now supports plugins. The default plugin is the classic interface for tag reading and editing.
- Plugins for writing NFO files, displaying lame/xing headers, jabber tunes, and library statistics.
- Module name is now `eyed3` (all lower case) to be more standards conforming.
- New `eyed3.id3.Tag` interface based on properties.
- Improved ID3 date frame support and 2.3<->2.4 conversion, and better conversions, in general.
- Native support for many more ID3 frame types.
- Python Package Index friendly, and installable with 'pip'.
- Improved mime-type detection.
- Improved unicode support.
- Support for config files to contain common options for the command-line tool.

0.6.18 - 11.25.2011 (Nobunny loves you)

New features:

- Support for disc number frames (TPOS). Thanks to Nathaniel Clark <nate@misrule.us>
- Added %Y (year) and %G (genre) substitution variables for file renames. Thanks to Otávio Pontes <otaviobp@gmail.com>
- Improved XML (–jep-118) escaping and a new option (–rfc822) to output in RFC 822 format. Thanks to Neil Schemenauer <nas@arctrix.com>
- –rename will NOT clobber existing files.
- New option –itunes to write only iTunes accepted genres. Thanks to Ben Isaacs <Ben XO me@ben-xo.com>

- If available the 'magic' module will be used to determine mimetypes when the filename is not enough. Thanks to Ville Skyttä <ville.skytta@iki.fi>
- –set-encoding can be used along with a version conversion arg to apply a new encoding to the new tag.
- Increased performance for mp3 header search when malformed GEOB frames are encountered. Thanks to Stephen Fairchild <sfairchild@bethere.co.uk>
- Less crashing when invalid user text frames are encountered.
- Less crashing when invalid BPM values (empty/non-numeric) are encountered.

0.6.17 - 02.01.2009 (The Point of No Return)

Bug fixes:

- Workaround invalid utf16
- Show all genres during –list-genres
- Workaround invalid PLCT frames.
- Show all tracks during –nfo output.

New features:

- Support for URL frames (W??? and WXXX)
- Program exit code for the 'eyeD3' command line tool

0.6.16 - 06.09.2008 (Gimme Danger)

Bug fixes:

- Typo fix of sysnc/unsync data. Thanks to Gergan Penkov <gergan@gmail.com>
- Infinite loop fix when dealing with malformed APIC frames.
- Tag.removeUserTextFrame helper. Thanks to David Grant <davidgrant@gmail.com>

0.6.15 - 03.02.2008 (Doin' The Cockroach)

Bug fixes:

- ID3 v1 comment encoding (latin1) bug fix (Renaud Saint-Gratien <rsg@nerim.net>)
- APIC picture type fix (Michael Schout <mschout@gkg.net>)
- Fixed console Unicode encoding for display.
- Fixed frame de-unsnychronization bugs.
- Round float BPMs to int (per the spec)

0.6.14 - 05.08.2007 (Breakthrough)

Bugs fixes:

- Fixed a nasty corruption of the first mp3 header when writing to files that do not already contain a tag.
- Fixed a bug that would duplicate TYER frames when setting new values.
- Fixed the reading/validation of some odd (i.e.,rare) mp3 headers

New Features:

- Encoding info extracted from Lame mp3 headers [Todd Zullinger]
- Genre names will now support '|' to allow for genres like "Rock|Punk|Pop-Punk" and '!' for "Oi!"

0.6.13 - 04.30.2007 (Undercovers On)

- Numerous write fixes, especially for v2.4 tags. Thanks to Alexander Thomas <dr-lex@dr-lex.34sp.com> for finding these.
- Add –no-zero-padding option to allow disabling of zero padding track numbers
- Add –nfo option to output NFO format files about music directories.
- Time computation fixes when MP3 frames headers were mistakingly found.

0.6.12 - 02.18.2007 (Rid Of Me)

- Handle Mac style line ending in lyrics and display with the proper output encoding. [Todd Zullinger]
- TDTG support and other date frame fixes. [Todd Zullinger]
- Output encoding bug fixes. [Todd Zullinger]

0.6.11 - 11.05.2006 (Disintegration)

- Support for GEOB (General encapsulated object) frames from Aaron VonderHaar <gruen0aermel@gmail.com>
- Decreased memory consumption during tag rewrites/removals.
- Allow the "reserved" mpeg version bits when not in strict mode.
- Solaris packages available via Blastwave - http://www.blastwave.org/packages.php/pyeyed3

0.6.10 - 03.19.2006 (Teh Mesk release)

- Unsynchronized lyrics (USLT) frame support [Todd Zullinger <tmz@pobox.com>]
- UTF16 bug fixes
- More forgiving of invalid User URL frames (WXXX)
- RPM spec file fixes [Knight Walker <kwalker@kobran.org>]
- More details in –verbose display

0.6.9 - 01.08.2005 (The Broken Social Scene Release)

- eyeD3 (the CLI) processes directories more efficiently
- A specific file system encoding can be specified for file renaming, see –fs-encoding (Andrew de Quincey)
- Faster mp3 header search for empty and/or corrupt mp3 files
- Extended header fixes
- Bug fix for saving files with no current tag
- What would a release be without unicode fixes, this time it's unicode filename output and JEP 0118 output.

0.6.8 - 08.29.2005 (The Anal Cunt Release)

- Frame header size bug. A _serious_ bug since writes MAY be affected (note: I've had no problems reported so far).

0.6.7 - 08.28.2005 (The Autopsy Release)

- Beats per minute (TPBM) interface
- Publisher/label (TPUB) interface
- When not in strict mode exceptions for invalid tags are quelled more often
- Support for iTunes ID3 spec violations regarding multiple APIC frames
- Bug fix where lang in CommentFrame was unicode where it MUST be ascii
- Bug fixed for v2.2 frame header sizes
- Bug fixed for v2.2 PIC frames
- File rename bug fixes
- Added -c option as an alias for –comment
- -i/–write-images now takes a destination path arg. Due to optparse non-support for optional arguments the path MUST be specified. This option no longer clobbers existing files.

0.6.6 - 05.15.2005 (The Electric Wizard Release)

- APIC frames can now be removed.
- An interface for TBPM (beats per minute) frames.
- Utf-16 bug fixes and better unicode display/output
- RPM spec file fixes

0.6.5 - 04.16.2005

- Read-only support for ID3 v2.2
- TPOS frame support (disc number in set).
- Bug fixes

0.6.4 - 02.05.2005

- Native support for play count (PCNT), and unique file id (UFID) frames.
- More relaxed genre processing.
- Sync-safe bug fixed when the tag header requests sync-safety and not the frames themselves.
- configure should successfly detect python release candidates and betas.

0.6.3 - 11.23.2004

- Much better unicode support when writing to the tag.
- Added Tag.setEncoding (–set-encoding) and –force-update
- Handle MP3 frames that violate spec when in non-strict mode. (Henning Kiel <henning.kiel@rwth-aachen.de>)
- Fix for Debian bug report #270964
- Various bug fixes.

0.6.2 - 8.29.2004 (Happy Birthday Mom!)

- TagFile.rename and Tag.tagToString (eyeD3 –rename=PATTERN). The latter supports substitution of tag values: %A is artist, %t is title, %a is album, %n is track number, and %N is track total.
- eyeD3 man page.
- User text frame (TXXX) API and –set-user-text-frame.
- Python 2.2/Optik compatibility works now.
- ebuild for Gentoo (http://eyed3.nicfit.net/releases/gentoo/)

0.6.1 - 5/14/2004 (Oz/2 Ohh my!)

- Unicode support - UTF-8, UTF-16, and UTF-16BE
- Adding images (APIC frames) is supported (–add-image, Tag.addImage(), etc.)
- Added a –relaxed option to be much more forgiving about tags that violate the spec. Quite useful for removing such tags.
- Added Tag.setTextFrame (–set-text-frame=FID:TEXT)
- Added –remove-comments.
- Now requires Python 2.3. Sorry, but I like cutting-edge python features.
- Better handling and conversion (2.3 <=> 2.4) of the multiple date frames.

- Output format per JEP 0118: User Tune, excluding xsd:duration format for <length/> (http://www.jabber.org/jeps/jep-0118.html)
- Lot's of bug fixes.
- Added a mailing list. Subscribe by sending a message to eyed3-devel-subscribe@nicfit.net

0.5.1 - 7/17/2003 (It's Too Damn Hot to Paint Release)

- Temporary files created during ID3 saving are now properly cleaned up.
- Fixed a "bug" when date frames are present but contain empty strings.
- Added a –no-color option to the eyeD3 driver.
- Workaround invalid tag sizes by implyied padding.
- Updated README

0.5.0 - 6/7/2003 (The Long Time Coming Release)

- ID3 v2.x saving.
- The eyeD3 driver/sample program is much more complete, allowing for most common tag operations such as tag display, editing, removal, etc. Optik is required to use this program. See the README.
- Complete access to all artist and title frames (i.e. TPE* and TIT*)
- Full v2.4 date support (i.e. TDRC).
- Case insensitive genres and compression fixes. (Gary Shao)
- ExtendedHeader support, including CRC checksums.
- Frame groups now supported.
- Syncsafe integer conversion bug fixes.
- Bug fixes related to data length indicator bytes.
- Genre and lot's of other bug fixes.

0.4.0 - 11/11/2002 (The Anniversary Release)

- Added the ability to save tags in ID v1.x format, including when the linked file was IDv2. Original backups are created by default for the time being...
- Added deleting of v1 and v2 frames from the file.
- Zlib frame data decompression is now working.
- bin/eyeD3 now displays user text frames, mp3 copyright and originality, URLs, all comments, and images. Using the –write-images arg will write each APIC image data to disk.
- Added eyeD3.isMp3File(), Tag.clear(), Tag.getImages(), Tag.getURLs(), Tag.getCDID(), FrameSet.removeFrame(), Tag.save(), ImageFrame.writeFile(), etc...
- Modified bin/eyeD3 to grok non Mp3 files. This allows testing with files containing only tag data and lays some groundwork for future OGG support.
- Fixed ImageFrame mime type problem.
- Fixed picture type scoping problems.

0.3.1 - 10/24/2002

- RPM packages added.
- Fixed a bug related to ID3 v1.1 track numbers. (Aubin Paul)
- Mp3AudioFile matches `*.mp3` and `*.MP3`. (Aubin Paul)

0.3.0 - 10/21/2002

- Added a higher level class called Mp3AudioFile.
- MP3 frame (including Xing) decoding for obtaining bit rate, play time, etc.
- Added APIC frame support (eyeD3.frames.Image).
- BUG FIX: Tag unsynchronization and deunsynchronization now works correctly and is ID3 v2.4 compliant.
- Tags can be linked with file names or file objects.
- More tag structure abstractions (TagHeader, Frame, FrameSet, etc.).
- BUG FIX: GenreExceptions were not being caught in eyeD3 driver.

0.2.0 - 8/15/2002

- ID3_Tag was renamed to Tag.
- Added Genre and GenreMap (eyeD3.genres is defined as the latter type)
- Added support of ID3 v1 and v2 comments.
- The ID3v2Frame file was renamed ID3v2 and refactoring work has started with the addition of TagHeader.

0.1.0 - 7/31/2002

- Initial release.

4.3 ChangeLog

Changes made to eyeD3 and the project's release history can be found in the *Release History*.

4.4 References

- ID3 v1.x Specification
- ID3 v2.4 Structure and Frames
- ID3 v2.3 Specification
- ID3 v2.2 Specification
- ISO 8601 Date and Time
- ISO 639-2 Language Codes
- MusicBrainz Tag Mappings

- MP3 Headers

4.5 Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

e

eyed3, 72
eyed3.core, 69
eyed3.id3, 44
eyed3.id3.apple, 32
eyed3.id3.frames, 33
eyed3.id3.headers, 38
eyed3.id3.tag, 40
eyed3.main, 71
eyed3.mimetype, 71
eyed3.mp3, 47
eyed3.mp3.headers, 45
eyed3.plugins, 62
eyed3.plugins.art, 47
eyed3.plugins.classic, 48
eyed3.plugins.display, 49
eyed3.plugins.fixup, 56
eyed3.plugins.genres, 57
eyed3.plugins.itunes, 57
eyed3.plugins.lameinfo, 57
eyed3.plugins.nfo, 58
eyed3.plugins.pymod, 58
eyed3.plugins.stats, 59
eyed3.plugins.xep_118, 61
eyed3.utils, 67
eyed3.utils.art, 63
eyed3.utils.binfuncs, 63
eyed3.utils.console, 64
eyed3.utils.log, 66
eyed3.utils.prompt, 67

INDEX

A

AbstractDateTagPattern (*class in eyed3.plugins.display*), 51
AccessorBase (*class in eyed3.id3.tag*), 42
album() (*eyed3.core.Tag property*), 69
album_artist() (*eyed3.core.Tag property*), 69
album_type() (*eyed3.id3.tag.Tag property*), 42
AlbumArtistTagPattern (*class in eyed3.plugins.display*), 49
AlbumTagPattern (*class in eyed3.plugins.display*), 49
AllCommentsTagPattern (*class in eyed3.plugins.display*), 51
AnsiBack (*class in eyed3.utils.console*), 64
AnsiCodes (*class in eyed3.utils.console*), 64
AnsiFore (*class in eyed3.utils.console*), 64
AnsiStyle (*class in eyed3.utils.console*), 64
ArgumentParser (*class in eyed3.utils*), 68
ArtFile (*class in eyed3.plugins.art*), 47
ARTIST (*eyed3.id3.frames.ImageFrame attribute*), 34
ARTIST (*in module eyed3.utils.art*), 63
artist() (*eyed3.core.Tag property*), 69
artist_origin() (*eyed3.id3.tag.Tag property*), 42
artist_url() (*eyed3.id3.tag.Tag property*), 41
ArtistOrigin (*class in eyed3.core*), 69
ArtistTagPattern (*class in eyed3.plugins.display*), 49
ArtistURLTagPattern (*class in eyed3.plugins.display*), 52
ArtPlugin (*class in eyed3.plugins.art*), 47
ArtworkRule (*class in eyed3.plugins.stats*), 59
audio_file_url() (*eyed3.id3.tag.Tag property*), 41
audio_source_url() (*eyed3.id3.tag.Tag property*), 41
AudioFile (*class in eyed3.core*), 70
AudioFileURLTagPattern (*class in eyed3.plugins.display*), 53
AudioInfo (*class in eyed3.core*), 69
AudioSourceURLTagPattern (*class in eyed3.plugins.display*), 52
AudioStat (*class in eyed3.plugins.stats*), 59

B

b() (*in module eyed3.utils*), 69
BACK_COVER (*eyed3.id3.frames.ImageFrame attribute*), 34
BACK_COVER (*in module eyed3.utils.art*), 63
BAND (*eyed3.id3.frames.ImageFrame attribute*), 34
BAND_LOGO (*eyed3.id3.frames.ImageFrame attribute*), 35
best_release_date() (*eyed3.id3.tag.Tag property*), 40
bin2bytes() (*in module eyed3.utils.binfuncs*), 63
bin2dec() (*in module eyed3.utils.binfuncs*), 63
bin2synchsafe() (*in module eyed3.utils.binfuncs*), 63
bit_rate_str() (*eyed3.mp3.Mp3AudioInfo property*), 47
BITRATE_DEDUCTIONS (*eyed3.plugins.stats.BitrateRule attribute*), 59
BitrateCounter (*class in eyed3.plugins.stats*), 60
BitrateRule (*class in eyed3.plugins.stats*), 59
BLINK_FAST (*eyed3.utils.console.AnsiStyle attribute*), 65
BLINK_SLOW (*eyed3.utils.console.AnsiStyle attribute*), 65
BLUE (*eyed3.utils.console.AnsiBack attribute*), 64
BLUE (*eyed3.utils.console.AnsiFore attribute*), 64
boldText() (*in module eyed3.utils.console*), 66
bpm() (*eyed3.id3.tag.Tag property*), 40
BPMTagPattern (*class in eyed3.plugins.display*), 52
BRIGHT (*eyed3.utils.console.AnsiStyle attribute*), 64
BRIGHT_COLORED_FISH (*eyed3.id3.frames.ImageFrame attribute*), 34
bytes2bin() (*in module eyed3.utils.binfuncs*), 63
bytes2dec() (*in module eyed3.utils.binfuncs*), 63

C

cd_id() (*eyed3.id3.tag.Tag property*), 40
cformat() (*in module eyed3.utils.console*), 66
ChapterFrame (*class in eyed3.id3.frames*), 37
chapters() (*eyed3.id3.tag.Tag property*), 41

ChaptersAccessor (*class in eyed3.id3.tag*), 43
chunkCopy() (*in module eyed3.utils*), 68
ClassicPlugin (*class in eyed3.plugins.classic*), 48
clear() (*eyed3.id3.headers.TagHeader method*), 38
clear() (*eyed3.id3.tag.Tag method*), 40
CommentFrame (*class in eyed3.id3.frames*), 36
comments() (*eyed3.id3.tag.Tag property*), 40
CommentsAccessor (*class in eyed3.id3.tag*), 42
CommentTagPattern (*class in eyed3.plugins.display*), 51
commercial_url() (*eyed3.id3.tag.Tag property*), 41
CommercialURLTagPattern (*class in eyed3.plugins.display*), 53
ComplexPattern (*class in eyed3.plugins.display*), 49
ComplexPattern.ExpectedParameter (*class in eyed3.plugins.display*), 49
ComplexPattern.Parameter (*class in eyed3.plugins.display*), 49
COMPOSER (*eyed3.id3.frames.ImageFrame attribute*), 34
composer() (*eyed3.id3.tag.Tag property*), 40
ComposerTagPattern (*class in eyed3.plugins.display*), 50
compress() (*eyed3.id3.frames.Frame static method*), 33
COMPRESSED (*eyed3.id3.headers.FrameHeader attribute*), 39
compressed() (*eyed3.id3.headers.FrameHeader property*), 39
compute() (*eyed3.plugins.stats.AudioStat method*), 59
compute() (*eyed3.plugins.stats.Stat method*), 59
compute_time_per_frame() (*in module eyed3.mp3.headers*), 45
CONDUCTOR (*eyed3.id3.frames.ImageFrame attribute*), 34
copyFlags() (*eyed3.id3.headers.FrameHeader method*), 39
copyright_url() (*eyed3.id3.tag.Tag property*), 41
CopyrightTagPattern (*class in eyed3.plugins.display*), 53
count() (*eyed3.id3.frames.PopularityFrame property*), 36
cprint() (*in module eyed3.utils.console*), 66
crc() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
crc_bit() (*eyed3.id3.headers.ExtendedTagHeader property*), 38
createFrame() (*in module eyed3.id3.frames*), 37
CYAN (*eyed3.utils.console.AnsiBack attribute*), 64
CYAN (*eyed3.utils.console.AnsiFore attribute*), 64

D

DATA_LEN (*eyed3.id3.headers.FrameHeader attribute*), 39
data_length_indicator() (*eyed3.id3.headers.FrameHeader property*), 40
Date (*class in eyed3.core*), 70
date() (*eyed3.id3.frames.DateFrame property*), 34
DateFrame (*class in eyed3.id3.frames*), 34
datePicker() (*in module eyed3.utils*), 68
day() (*eyed3.core.Date property*), 70
dec2bin() (*in module eyed3.utils.binfuncs*), 63
dec2bytes() (*in module eyed3.utils.binfuncs*), 63
decode() (*eyed3.mp3.headers.LameHeader method*), 46
decode() (*eyed3.mp3.headers.Mp3Header method*), 45
decode() (*eyed3.mp3.headers.VbriHeader method*), 45
decode() (*eyed3.mp3.headers.XingHeader method*), 45
decodeUnicode() (*in module eyed3.id3.frames*), 37
decompress() (*eyed3.id3.frames.Frame static method*), 33
decrypt() (*eyed3.id3.frames.Frame static method*), 33
DescriptableTagPattern (*class in eyed3.plugins.display*), 51
DESCRIPTION (*eyed3.plugins.art.ArtPlugin attribute*), 47
DESCRIPTION (*eyed3.plugins.classic.ClassicPlugin attribute*), 48
DESCRIPTION (*eyed3.plugins.display.AlbumArtistTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.AlbumTagPattern attribute*), 49
DESCRIPTION (*eyed3.plugins.display.AllCommentsTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.ArtistTagPattern attribute*), 49
DESCRIPTION (*eyed3.plugins.display.ArtistURLTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.AudioFileURLTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.AudioSourceURLTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.BPMTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.CommentTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.CommercialURLTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.ComplexPattern attribute*), 49
DESCRIPTION (*eyed3.plugins.display.ComposerTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.CopyrightTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.DiscTagPattern attribute*), 50

DESCRIPTION (*eyed3.plugins.display.DiscTotalTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.DisplayPlugin attribute*), 56
DESCRIPTION (*eyed3.plugins.display.EncodingDateTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.FunctionAudioModePattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionBitRatePattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionFilenamePattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionFilesizePattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionFormatPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.FunctionLengthPattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionMPEGVersionPattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionNotEmptyPattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionNumberPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.FunctionRepeatPattern attribute*), 56
DESCRIPTION (*eyed3.plugins.display.FunctionSampleFrequencyPattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.FunctionTagVersionPattern attribute*), 55
DESCRIPTION (*eyed3.plugins.display.GenreIdTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.GenreTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.ImagesTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.ImageURLsTagPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.InternetRadioURLTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.LyricsTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.MusicCDIdTagPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.ObjectsTagPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.OriginalReleaseDateTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.PaymentURLTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.PlayCountTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.PopularitiesTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.PrivatesTagPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.PublisherTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.PublisherURLTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.RecordingDateTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.ReleaseDateTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.TaggingDateTagPattern attribute*), 51
DESCRIPTION (*eyed3.plugins.display.TermsOfUseTagPattern attribute*), 54
DESCRIPTION (*eyed3.plugins.display.TextsTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.TitleTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.TrackTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.TrackTotalTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.display.UniqueFileIDTagPattern attribute*), 52
DESCRIPTION (*eyed3.plugins.display.UserURLsTagPattern attribute*), 53
DESCRIPTION (*eyed3.plugins.display.YearTagPattern attribute*), 50
DESCRIPTION (*eyed3.plugins.fixup.FixupPlugin attribute*), 56
DESCRIPTION (*eyed3.plugins.genres.GenreListPlugin attribute*), 57
DESCRIPTION (*eyed3.plugins.lameinfo.LameInfoPlugin attribute*), 57
DESCRIPTION (*eyed3.plugins.nfo.NfoPlugin attribute*), 58
DESCRIPTION (*eyed3.plugins.Plugin attribute*), 62
DESCRIPTION (*eyed3.plugins.pymod.PyModulePlugin attribute*), 58
description() (*eyed3.id3.frames.DescriptionLangTextFrame property*), 36
description() (*eyed3.id3.frames.ImageFrame property*), 35
description() (*eyed3.id3.frames.ObjectFrame property*), 35
description() (*eyed3.id3.frames.UserTextFrame property*), 33
description() (*eyed3.id3.frames.UserUrlFrame property*), 34
DescriptionLangTextFrame (*class in eyed3.id3.frames*), 36
deunsyncData() (*in module eyed3.id3.frames*), 37
DIM (*eyed3.utils.console.AnsiStyle attribute*), 64
dirDate() (*in module eyed3.plugins.fixup*), 56

DISABLE_PROMPT (*in module eyed3.utils.prompt*), 67
disc_num() (*eyed3.id3.tag.Tag property*), 41
DiscTagPattern (*class in eyed3.plugins.display*), 50
DiscTotalTagPattern (*class in eyed3.plugins.display*), 50
DisplayException, 56
DisplayPlugin (*class in eyed3.plugins.display*), 56
DltAccessor (*class in eyed3.id3.tag*), 42
DURING_PERFORMANCE (*eyed3.id3.frames.ImageFrame attribute*), 34
DURING_RECORDING (*eyed3.id3.frames.ImageFrame attribute*), 34

E

email() (*eyed3.id3.frames.PopularityFrame property*), 36
ENCODER_FLAGS (*eyed3.mp3.headers.LameHeader attribute*), 46
encoding() (*eyed3.id3.frames.Frame property*), 33
encoding_date() (*eyed3.id3.tag.Tag property*), 40
EncodingDateTagPattern (*class in eyed3.plugins.display*), 51
encrypt() (*eyed3.id3.frames.Frame static method*), 33
ENCRYPTED (*eyed3.id3.headers.FrameHeader attribute*), 39
encrypted() (*eyed3.id3.headers.FrameHeader property*), 39
end() (*eyed3.id3.frames.StartEndTuple property*), 37
Error, 72
ERROR_COLOR() (*in module eyed3.utils.console*), 65
ExtendedTagHeader (*class in eyed3.id3.headers*), 38
EXTENSION (*eyed3.mimetype.Id3Tag attribute*), 71
EXTENSION (*eyed3.mimetype.Id3TagExt attribute*), 71
EXTENSION (*eyed3.mimetype.M3u attribute*), 71
EXTENSION (*eyed3.mimetype.Mp2x attribute*), 71
EXTENSION (*eyed3.mimetype.Mp3Invalids attribute*), 71
eyed3 (*module*), 72
eyed3.core (*module*), 69
eyed3.id3 (*module*), 44
eyed3.id3.apple (*module*), 32
eyed3.id3.frames (*module*), 33
eyed3.id3.headers (*module*), 38
eyed3.id3.tag (*module*), 40
eyed3.main (*module*), 71
eyed3.mimetype (*module*), 71
eyed3.mp3 (*module*), 47
eyed3.mp3.headers (*module*), 45
eyed3.plugins (*module*), 62
eyed3.plugins.art (*module*), 47
eyed3.plugins.classic (*module*), 48
eyed3.plugins.display (*module*), 49
eyed3.plugins.fixup (*module*), 56
eyed3.plugins.genres (*module*), 57
eyed3.plugins.itunes (*module*), 57
eyed3.plugins.lameinfo (*module*), 57
eyed3.plugins.nfo (*module*), 58
eyed3.plugins.pymod (*module*), 58
eyed3.plugins.stats (*module*), 59
eyed3.plugins.xep_118 (*module*), 61
eyed3.utils (*module*), 67
eyed3.utils.art (*module*), 63
eyed3.utils.binfuncs (*module*), 63
eyed3.utils.console (*module*), 64
eyed3.utils.log (*module*), 66
eyed3.utils.prompt (*module*), 67

F

FILE_ALTER (*eyed3.id3.headers.FrameHeader attribute*), 39
file_alter() (*eyed3.id3.headers.FrameHeader property*), 39
FileCounterStat (*class in eyed3.plugins.stats*), 60
FileHandler (*class in eyed3.utils*), 67
FileInfo (*class in eyed3.id3.tag*), 42
filename() (*eyed3.id3.frames.ObjectFrame property*), 35
FILENAMES (*in module eyed3.utils.art*), 63
FileRule (*class in eyed3.plugins.stats*), 59
findHeader() (*in module eyed3.mp3.headers*), 45
FixupPlugin (*class in eyed3.plugins.fixup*), 56
formatSize() (*in module eyed3.utils*), 68
formatText() (*in module eyed3.utils.console*), 66
formatTime() (*in module eyed3.utils*), 68
formatTimeDelta() (*in module eyed3.utils*), 68
Frame (*class in eyed3.id3.frames*), 33
FrameException, 33
FrameHeader (*class in eyed3.id3.headers*), 39
frameiter() (*eyed3.id3.tag.Tag method*), 42
FrameSet (*class in eyed3.id3.frames*), 37
FROM_ID3_ART_TYPES (*in module eyed3.utils.art*), 63
FRONT_COVER (*eyed3.id3.frames.ImageFrame attribute*), 34
FRONT_COVER (*in module eyed3.utils.art*), 63
FunctionAudioModePattern (*class in eyed3.plugins.display*), 55
FunctionBitRatePattern (*class in eyed3.plugins.display*), 55
FunctionFilenamePattern (*class in eyed3.plugins.display*), 54
FunctionFilesizePattern (*class in eyed3.plugins.display*), 55
FunctionFormatPattern (*class in eyed3.plugins.display*), 54
FunctionLengthPattern (*class in eyed3.plugins.display*), 55
FunctionMPEGVersionPattern (*class in eyed3.plugins.display*), 55

FunctionNotEmptyPattern (*class in eyed3.plugins.display*), 55
FunctionNumberPattern (*class in eyed3.plugins.display*), 54
FunctionPattern (*class in eyed3.plugins.display*), 54
FunctionRepeatPattern (*class in eyed3.plugins.display*), 55
FunctionSampleFrequencePattern (*class in eyed3.plugins.display*), 55
FunctionTagVersionPattern (*class in eyed3.plugins.display*), 55

G

Genre (*class in eyed3.id3*), 44
genre() (*eyed3.id3.tag.Tag property*), 41
GENRE_MAX (*eyed3.id3.GenreMap attribute*), 44
GENRE_MIN (*eyed3.id3.GenreMap attribute*), 44
GenreException, 44
GenreIdTagPattern (*class in eyed3.plugins.display*), 50
GenreListPlugin (*class in eyed3.plugins.genres*), 57
GenreMap (*class in eyed3.id3*), 44
GenreTagPattern (*class in eyed3.plugins.display*), 50
get() (*eyed3.id3.tag.AccessorBase method*), 42
get() (*eyed3.id3.tag.ChaptersAccessor method*), 43
get() (*eyed3.id3.tag.DltAccessor method*), 42
get() (*eyed3.id3.tag.ImagesAccessor method*), 42
get() (*eyed3.id3.tag.ObjectsAccessor method*), 42
get() (*eyed3.id3.tag.PopularitiesAccessor method*), 43
get() (*eyed3.id3.tag.PrivatesAccessor method*), 43
get() (*eyed3.id3.tag.TocAccessor method*), 43
get() (*eyed3.id3.tag.UniqueFileIdAccessor method*), 43
get() (*eyed3.id3.tag.UserTextsAccessor method*), 43
get() (*eyed3.id3.tag.UserUrlsAccessor method*), 43
getAllFrames() (*eyed3.id3.frames.FrameSet method*), 37
getArtFromTag() (*in module eyed3.utils.art*), 63
getBestDate() (*eyed3.id3.tag.Tag method*), 40
getLogger() (*in module eyed3.utils.log*), 66
getTextFrame() (*eyed3.id3.tag.Tag method*), 40
getTtySize() (*in module eyed3.utils.console*), 66
getXML() (*eyed3.plugins.xep_118.Xep118Plugin method*), 62
GREEN (*eyed3.utils.console.AnsiBack attribute*), 64
GREEN (*eyed3.utils.console.AnsiFore attribute*), 64
GREY (*eyed3.utils.console.AnsiBack attribute*), 64
GREY (*eyed3.utils.console.AnsiFore attribute*), 64
GROUPED (*eyed3.id3.headers.FrameHeader attribute*), 39
grouped() (*eyed3.id3.headers.FrameHeader property*), 40
guessMimetype() (*in module eyed3.mimetype*), 71
guessMimetype() (*in module eyed3.utils*), 67

H

handleDirectory() (*eyed3.plugins.art.ArtPlugin method*), 48
handleDirectory() (*eyed3.plugins.fixup.FixupPlugin method*), 57
handleDirectory() (*eyed3.plugins.LoaderPlugin method*), 62
handleDirectory() (*eyed3.plugins.pymod.PyModulePlugin method*), 58
handleDirectory() (*eyed3.utils.FileHandler method*), 68
handleDone() (*eyed3.plugins.art.ArtPlugin method*), 48
handleDone() (*eyed3.plugins.display.DisplayPlugin method*), 56
handleDone() (*eyed3.plugins.fixup.FixupPlugin method*), 57
handleDone() (*eyed3.plugins.LoaderPlugin method*), 62
handleDone() (*eyed3.plugins.nfo.NfoPlugin method*), 58
handleDone() (*eyed3.plugins.Plugin method*), 62
handleDone() (*eyed3.plugins.pymod.PyModulePlugin method*), 58
handleDone() (*eyed3.plugins.stats.StatisticsPlugin method*), 61
handleDone() (*eyed3.utils.FileHandler method*), 68
handleEdits() (*eyed3.plugins.classic.ClassicPlugin method*), 48
handleFile() (*eyed3.plugins.classic.ClassicPlugin method*), 48
handleFile() (*eyed3.plugins.display.DisplayPlugin method*), 56
handleFile() (*eyed3.plugins.fixup.FixupPlugin method*), 56
handleFile() (*eyed3.plugins.itunes.Podcast method*), 57
handleFile() (*eyed3.plugins.lameinfo.LameInfoPlugin method*), 57
handleFile() (*eyed3.plugins.LoaderPlugin method*), 62
handleFile() (*eyed3.plugins.nfo.NfoPlugin method*), 58
handleFile() (*eyed3.plugins.Plugin method*), 62
handleFile() (*eyed3.plugins.pymod.PyModulePlugin method*), 58
handleFile() (*eyed3.plugins.stats.StatisticsPlugin method*), 61

handleFile() (*eyed3.plugins.xep_118.Xep118Plugin method*), 61
handleFile() (*eyed3.utils.FileHandler method*), 68
handlePadding() (*eyed3.plugins.classic.ClassicPlugin method*), 48
handleRemoves() (*eyed3.plugins.classic.ClassicPlugin method*), 48
header() (*eyed3.id3.frames.Frame property*), 33
HEADER_COLOR() (*in module eyed3.utils.console*), 65
HIDDEN_FILES (*eyed3.plugins.stats.FileCounterStat attribute*), 60
hour() (*eyed3.core.Date property*), 70

I

ICON (*eyed3.id3.frames.ImageFrame attribute*), 34
id() (*eyed3.id3.Genre property*), 44
ID3_GENRE_MAX (*eyed3.id3.GenreMap attribute*), 44
ID3_GENRE_MIN (*eyed3.id3.GenreMap attribute*), 44
id3Encode() (*eyed3.core.ArtistOrigin method*), 69
id3EncodingToString() (*in module eyed3.id3.frames*), 37
Id3FrameCounter (*class in eyed3.plugins.stats*), 60
Id3FrameRules (*class in eyed3.plugins.stats*), 59
Id3ImageTypeCounter (*class in eyed3.plugins.stats*), 61
Id3Tag (*class in eyed3.mimetype*), 71
Id3TagExt (*class in eyed3.mimetype*), 71
Id3TagRules (*class in eyed3.plugins.stats*), 59
Id3VersionCounter (*class in eyed3.plugins.stats*), 60
idpattern (*eyed3.id3.tag.TagTemplate attribute*), 44
ILLUSTRATION (*eyed3.id3.frames.ImageFrame attribute*), 35
image_data() (*eyed3.plugins.art.ArtFile property*), 47
image_enc_restriction() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
image_enc_restriction_description() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
image_size_restriction() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
image_size_restriction_description() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
ImageFrame (*class in eyed3.id3.frames*), 34
images() (*eyed3.id3.tag.Tag property*), 40
ImagesAccessor (*class in eyed3.id3.tag*), 42
ImagesTagPattern (*class in eyed3.plugins.display*), 53
ImageURLsTagPattern (*class in eyed3.plugins.display*), 53
info() (*eyed3.core.AudioFile property*), 70
init() (*eyed3.utils.console.AnsiCodes class method*), 64
initLogging() (*in module eyed3.utils.log*), 66
initStatTimes() (*eyed3.id3.tag.FileInfo method*), 42
initTag() (*eyed3.id3.TagFile method*), 45
initTag() (*eyed3.mp3.Mp3AudioFile method*), 47
internet_radio_url() (*eyed3.id3.tag.Tag property*), 41
InternetRadioURLTagPattern (*class in eyed3.plugins.display*), 53
INVERSE (*eyed3.utils.console.AnsiStyle attribute*), 65
isV1() (*eyed3.id3.tag.Tag method*), 40
isV2() (*eyed3.id3.tag.Tag method*), 40
isValidHeader() (*in module eyed3.mp3.headers*), 45
isValidVersion() (*in module eyed3.id3*), 44
ITALICS (*eyed3.utils.console.AnsiStyle attribute*), 64

L

LameHeader (*class in eyed3.mp3.headers*), 45
LameInfoPlugin (*class in eyed3.plugins.lameinfo*), 57
lamevercmp() (*in module eyed3.mp3.headers*), 47
lang() (*eyed3.id3.frames.LanguageCodeMixin property*), 36
LanguageCodeMixin (*class in eyed3.id3.frames*), 36
LEAD_ARTIST (*eyed3.id3.frames.ImageFrame attribute*), 34
LEAFLET (*eyed3.id3.frames.ImageFrame attribute*), 34
LIVE (*in module eyed3.utils.art*), 63
load() (*in module eyed3*), 72
load() (*in module eyed3.core*), 69
load() (*in module eyed3.plugins*), 62
LoaderPlugin (*class in eyed3.plugins*), 62
Logger (*class in eyed3.utils.log*), 66
LoggingAction (*class in eyed3.utils*), 68
LOGO (*in module eyed3.utils.art*), 63
LYRICIST (*eyed3.id3.frames.ImageFrame attribute*), 34
lyrics() (*eyed3.id3.tag.Tag property*), 41
LyricsAccessor (*class in eyed3.id3.tag*), 42
LyricsFrame (*class in eyed3.id3.frames*), 36
LyricsTagPattern (*class in eyed3.plugins.display*), 52

M

M3u (*class in eyed3.mimetype*), 71
MAGENTA (*eyed3.utils.console.AnsiBack attribute*), 64
MAGENTA (*eyed3.utils.console.AnsiFore attribute*), 64
main() (*in module eyed3.main*), 71
major_version() (*eyed3.id3.headers.FrameHeader property*), 39

major_version() (*eyed3.id3.headers.TagHeader property*), 38
makeCmdLineParser() (*in module eyed3.main*), 71
makeFileName() (*eyed3.id3.frames.ImageFrame method*), 35
makeUniqueFileName() (*in module eyed3.utils*), 68
map() (*eyed3.utils.console.ProgressBar class method*), 66
map2_2FrameId() (*in module eyed3.id3.frames*), 38
match() (*eyed3.mimetype.Id3Tag method*), 71
match() (*eyed3.mimetype.M3u method*), 71
match() (*eyed3.mimetype.Mp2x method*), 71
match() (*eyed3.mimetype.Mp3Invalids method*), 71
matchArtFile() (*in module eyed3.utils.art*), 63
MAX_TYPE (*eyed3.id3.frames.ImageFrame attribute*), 35
md5Data() (*in module eyed3.plugins.art*), 48
md5File() (*in module eyed3.plugins.art*), 48
MEDIA (*eyed3.id3.frames.ImageFrame attribute*), 34
message() (*eyed3.plugins.display.DisplayException property*), 56
message() (*eyed3.plugins.display.PatternCompileException property*), 56
MIME (*eyed3.mimetype.Id3Tag attribute*), 71
MIME (*eyed3.mimetype.M3u attribute*), 71
MIME (*eyed3.mimetype.Mp2x attribute*), 71
MIME (*eyed3.mimetype.Mp3Invalids attribute*), 71
mime_type() (*eyed3.id3.frames.ImageFrame property*), 35
mime_type() (*eyed3.id3.frames.ObjectFrame property*), 35
mime_type() (*eyed3.plugins.art.ArtFile property*), 47
MimeTypeStat (*class in eyed3.plugins.stats*), 60
MIN_TYPE (*eyed3.id3.frames.ImageFrame attribute*), 35
minor_version() (*eyed3.id3.headers.FrameHeader property*), 39
minor_version() (*eyed3.id3.headers.TagHeader property*), 38
minute() (*eyed3.core.Date property*), 70
MISC_COVER (*in module eyed3.utils.art*), 63
month() (*eyed3.core.Date property*), 70
Mp2x (*class in eyed3.mimetype*), 71
Mp3AudioFile (*class in eyed3.mp3*), 47
Mp3AudioInfo (*class in eyed3.mp3*), 47
Mp3Exception, 47
Mp3Header (*class in eyed3.mp3.headers*), 45
Mp3Invalids (*class in eyed3.mimetype*), 71
MusicCDIdFrame (*class in eyed3.id3.frames*), 35
MusicCDIdTagPattern (*class in eyed3.plugins.display*), 54

N

name() (*eyed3.id3.Genre property*), 44
name() (*eyed3.plugins.display.ComplexPattern property*), 49
NAMES (*eyed3.plugins.art.ArtPlugin attribute*), 48
NAMES (*eyed3.plugins.classic.ClassicPlugin attribute*), 48
NAMES (*eyed3.plugins.display.AlbumArtistTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.AlbumTagPattern attribute*), 49
NAMES (*eyed3.plugins.display.AllCommentsTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.ArtistTagPattern attribute*), 49
NAMES (*eyed3.plugins.display.ArtistURLTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.AudioFileURLTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.AudioSourceURLTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.BPMTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.CommentTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.CommercialURLTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.ComplexPattern attribute*), 49
NAMES (*eyed3.plugins.display.ComposerTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.CopyrightTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.DiscTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.DiscTotalTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.DisplayPlugin attribute*), 56
NAMES (*eyed3.plugins.display.EncodingDateTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.FunctionAudioModePattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionBitRatePattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionFilenamePattern attribute*), 54
NAMES (*eyed3.plugins.display.FunctionFilesizePattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionFormatPattern attribute*), 54
NAMES (*eyed3.plugins.display.FunctionLengthPattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionMPEGVersionPattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionNotEmptyPattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionNumberPattern*

attribute), 54
NAMES (*eyed3.plugins.display.FunctionRepeatPattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionSampleFrequencePattern attribute*), 55
NAMES (*eyed3.plugins.display.FunctionTagVersionPattern attribute*), 55
NAMES (*eyed3.plugins.display.GenreIdTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.GenreTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.ImagesTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.ImageURLsTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.InternetRadioURLTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.LyricsTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.MusicCDIdTagPattern attribute*), 54
NAMES (*eyed3.plugins.display.ObjectsTagPattern attribute*), 54
NAMES (*eyed3.plugins.display.OriginalReleaseDateTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.PaymentURLTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.PlayCountTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.PopularitiesTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.PrivatesTagPattern attribute*), 54
NAMES (*eyed3.plugins.display.PublisherTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.PublisherURLTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.RecordingDateTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.ReleaseDateTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.TaggingDateTagPattern attribute*), 51
NAMES (*eyed3.plugins.display.TermsOfUseTagPattern attribute*), 54
NAMES (*eyed3.plugins.display.TextsTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.TitleTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.TrackTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.TrackTotalTagPattern attribute*), 50
NAMES (*eyed3.plugins.display.UniqueFileIDTagPattern attribute*), 52
NAMES (*eyed3.plugins.display.UserURLsTagPattern attribute*), 53
NAMES (*eyed3.plugins.display.YearTagPattern attribute*), 50
NAMES (*eyed3.plugins.fixup.FixupPlugin attribute*), 56
NAMES (*eyed3.plugins.genres.GenreListPlugin attribute*), 57
NAMES (*eyed3.plugins.itunes.Podcast attribute*), 57
NAMES (*eyed3.plugins.lameinfo.LameInfoPlugin attribute*), 57
NAMES (*eyed3.plugins.nfo.NfoPlugin attribute*), 58
NAMES (*eyed3.plugins.Plugin attribute*), 62
NAMES (*eyed3.plugins.pymod.PyModulePlugin attribute*), 58
NAMES (*eyed3.plugins.stats.StatisticsPlugin attribute*), 61
NAMES (*eyed3.plugins.xep_118.Xep118Plugin attribute*), 61
next() (*eyed3.utils.console.ProgressBar method*), 65
NfoPlugin (*class in eyed3.plugins.nfo*), 58
NO_OFFSET (*eyed3.id3.frames.ChapterFrame attribute*), 37
non_std_genre() (*eyed3.id3.tag.Tag property*), 41
normalizeVersion() (*in module eyed3.id3*), 44

O

ObjectFrame (*class in eyed3.id3.frames*), 35
objects() (*eyed3.id3.tag.Tag property*), 41
ObjectsAccessor (*class in eyed3.id3.tag*), 42
ObjectsTagPattern (*class in eyed3.plugins.display*), 54
ORDERED_FLAG_BIT (*eyed3.id3.frames.TocFrame attribute*), 37
original_artist() (*eyed3.id3.tag.Tag property*), 42
original_release_date() (*eyed3.id3.tag.Tag property*), 41
OriginalReleaseDateTagPattern (*class in eyed3.plugins.display*), 51
os_walk() (*in module eyed3.utils*), 67
os_walk_unpack() (*in module eyed3.utils*), 67
OTHER (*eyed3.id3.frames.ImageFrame attribute*), 34
OTHER_FILES (*eyed3.plugins.stats.FileCounterStat attribute*), 60
OTHER_ICON (*eyed3.id3.frames.ImageFrame attribute*), 34
output_for() (*eyed3.plugins.display.ComplexPattern method*), 49
output_for() (*eyed3.plugins.display.Pattern method*), 49
output_for() (*eyed3.plugins.display.TextPattern method*), 49

owner_id() (*eyed3.id3.frames.UniqueFileIDFrame property*), 36

P

PARAMETERS (*eyed3.plugins.display.AllCommentsTagPattern attribute*), 51
PARAMETERS (*eyed3.plugins.display.CommentTagPattern attribute*), 51
PARAMETERS (*eyed3.plugins.display.ComplexPattern attribute*), 49
PARAMETERS (*eyed3.plugins.display.DescriptableTagPattern attribute*), 51
PARAMETERS (*eyed3.plugins.display.FunctionFilenamePattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.FunctionFormatPattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.FunctionMPEGVersionPattern attribute*), 55
PARAMETERS (*eyed3.plugins.display.FunctionNotEmptyPattern attribute*), 55
PARAMETERS (*eyed3.plugins.display.FunctionNumberPattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.FunctionRepeatPattern attribute*), 56
PARAMETERS (*eyed3.plugins.display.ImagesTagPattern attribute*), 53
PARAMETERS (*eyed3.plugins.display.ImageURLsTagPattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.LyricsTagPattern attribute*), 52
PARAMETERS (*eyed3.plugins.display.ObjectsTagPattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.PopularitiesTagPattern attribute*), 52
PARAMETERS (*eyed3.plugins.display.PrivatesTagPattern attribute*), 54
PARAMETERS (*eyed3.plugins.display.TextsTagPattern attribute*), 52
PARAMETERS (*eyed3.plugins.display.UniqueFileIDTagPattern attribute*), 52
PARAMETERS (*eyed3.plugins.display.UserURLsTagPattern attribute*), 53
parameters() (*eyed3.plugins.display.ComplexPattern property*), 49
parse() (*eyed3.core.Date static method*), 70
parse() (*eyed3.id3.frames.ChapterFrame method*), 37
parse() (*eyed3.id3.frames.DateFrame method*), 34
parse() (*eyed3.id3.frames.DescriptionLangTextFrame method*), 36
parse() (*eyed3.id3.frames.Frame method*), 33
parse() (*eyed3.id3.frames.FrameSet method*), 37
parse() (*eyed3.id3.frames.ImageFrame method*), 35
parse() (*eyed3.id3.frames.MusicCDIdFrame method*), 35
parse() (*eyed3.id3.frames.ObjectFrame method*), 35
parse() (*eyed3.id3.frames.PlayCountFrame method*), 36
parse() (*eyed3.id3.frames.PopularityFrame method*), 36
parse() (*eyed3.id3.frames.PrivateFrame method*), 35
parse() (*eyed3.id3.frames.TermsOfUseFrame method*), 36
parse() (*eyed3.id3.frames.TextFrame method*), 33
parse() (*eyed3.id3.frames.TocFrame method*), 37
parse() (*eyed3.id3.frames.UniqueFileIDFrame method*), 36
parse() (*eyed3.id3.frames.UrlFrame method*), 34
parse() (*eyed3.id3.frames.UserTextFrame method*), 34
parse() (*eyed3.id3.frames.UserUrlFrame method*), 34
parse() (*eyed3.id3.Genre static method*), 44
parse() (*eyed3.id3.headers.ExtendedTagHeader method*), 39
parse() (*eyed3.id3.headers.FrameHeader static method*), 40
parse() (*eyed3.id3.headers.TagHeader method*), 38
parse() (*eyed3.id3.tag.Tag method*), 40
parseCommandLine() (*in module eyed3.main*), 71
parseError() (*in module eyed3.core*), 70
parseIntList() (*in module eyed3.utils.prompt*), 67
path() (*eyed3.core.AudioFile property*), 70
Pattern (*class in eyed3.plugins.display*), 49
pattern (*eyed3.id3.tag.TagTemplate attribute*), 44
pattern_class_parameters() (*eyed3.plugins.display.Pattern static method*), 49
PatternCompileException, 56
payment_url() (*eyed3.id3.tag.Tag property*), 41
PaymentURLTagPattern (*class in eyed3.plugins.display*), 53
PCST (*class in eyed3.id3.apple*), 32
percent() (*eyed3.plugins.stats.Stat method*), 59
picture_type() (*eyed3.id3.frames.ImageFrame property*), 35
picTypeToString() (*eyed3.id3.frames.ImageFrame static method*), 35
pilImage() (*in module eyed3.plugins.art*), 48
pilImageDetails() (*in module eyed3.plugins.art*), 48
PlaceholderUsagePattern (*class in eyed3.plugins.display*), 49
play_count() (*eyed3.id3.tag.Tag property*), 40
PlayCountFrame (*class in eyed3.id3.frames*), 35
PlayCountTagPattern (*class in eyed3.plugins.display*), 51
Plugin (*class in eyed3.plugins*), 62
Podcast (*class in eyed3.plugins.itunes*), 57
popularities() (*eyed3.id3.tag.Tag property*), 41
PopularitiesAccessor (*class in eyed3.id3.tag*), 43

PopularitiesTagPattern (*class in eyed3.plugins.display*), 52
PopularityFrame (*class in eyed3.id3.frames*), 36
PRESETS (*eyed3.mp3.headers.LameHeader attribute*), 46
printAudioInfo() (*eyed3.plugins.classic.ClassicPlugin method*), 48
printError() (*in module eyed3.utils.console*), 66
printHeader() (*eyed3.plugins.classic.ClassicPlugin method*), 48
printHeader() (*eyed3.plugins.lameinfo.LameInfoPlugin method*), 57
printHeader() (*in module eyed3.utils.console*), 66
printMsg() (*in module eyed3.utils.console*), 66
printTag() (*eyed3.plugins.classic.ClassicPlugin method*), 48
printWarning() (*in module eyed3.utils.console*), 66
PrivateFrame (*class in eyed3.id3.frames*), 35
privates() (*eyed3.id3.tag.Tag property*), 41
PrivatesAccessor (*class in eyed3.id3.tag*), 42
PrivatesTagPattern (*class in eyed3.plugins.display*), 54
profileMain() (*in module eyed3.main*), 71
ProgressBar (*class in eyed3.utils.console*), 65
prompt() (*in module eyed3.utils.prompt*), 67
PromptExit, 67
publisher() (*eyed3.id3.tag.Tag property*), 40
PUBLISHER_LOGO (*eyed3.id3.frames.ImageFrame attribute*), 35
publisher_url() (*eyed3.id3.tag.Tag property*), 41
PublisherTagPattern (*class in eyed3.plugins.display*), 52
PublisherURLTagPattern (*class in eyed3.plugins.display*), 53
PyModulePlugin (*class in eyed3.plugins.pymod*), 58

R

rating() (*eyed3.id3.frames.PopularityFrame property*), 36
read_only (*eyed3.core.Tag attribute*), 69
READ_ONLY (*eyed3.id3.headers.FrameHeader attribute*), 39
read_only() (*eyed3.id3.headers.FrameHeader property*), 39
recording_date() (*eyed3.id3.tag.Tag property*), 41
RECORDING_LOCATION (*eyed3.id3.frames.ImageFrame attribute*), 34
RecordingDateTagPattern (*class in eyed3.plugins.display*), 51
RED (*eyed3.utils.console.AnsiBack attribute*), 64
RED (*eyed3.utils.console.AnsiFore attribute*), 64
release_date() (*eyed3.id3.tag.Tag property*), 41
ReleaseDateTagPattern (*class in eyed3.plugins.display*), 51
remove() (*eyed3.id3.tag.AccessorBase method*), 42
remove() (*eyed3.id3.tag.ChaptersAccessor method*), 43
remove() (*eyed3.id3.tag.DltAccessor method*), 42
remove() (*eyed3.id3.tag.ImagesAccessor method*), 42
remove() (*eyed3.id3.tag.ObjectsAccessor method*), 42
remove() (*eyed3.id3.tag.PopularitiesAccessor method*), 43
remove() (*eyed3.id3.tag.PrivatesAccessor method*), 43
remove() (*eyed3.id3.tag.Tag static method*), 41
remove() (*eyed3.id3.tag.TocAccessor method*), 43
remove() (*eyed3.id3.tag.UniqueFileIdAccessor method*), 43
remove() (*eyed3.id3.tag.UserTextsAccessor method*), 43
remove() (*eyed3.id3.tag.UserUrlsAccessor method*), 43
rename() (*eyed3.core.AudioFile method*), 70
render() (*eyed3.id3.apple.PCST method*), 33
render() (*eyed3.id3.frames.ChapterFrame method*), 37
render() (*eyed3.id3.frames.DescriptionLangTextFrame method*), 36
render() (*eyed3.id3.frames.Frame method*), 33
render() (*eyed3.id3.frames.ImageFrame method*), 35
render() (*eyed3.id3.frames.ObjectFrame method*), 35
render() (*eyed3.id3.frames.PlayCountFrame method*), 36
render() (*eyed3.id3.frames.PopularityFrame method*), 36
render() (*eyed3.id3.frames.PrivateFrame method*), 35
render() (*eyed3.id3.frames.TermsOfUseFrame method*), 36
render() (*eyed3.id3.frames.TextFrame method*), 33
render() (*eyed3.id3.frames.TocFrame method*), 37
render() (*eyed3.id3.frames.UniqueFileIDFrame method*), 36
render() (*eyed3.id3.frames.UrlFrame method*), 34
render() (*eyed3.id3.frames.UserTextFrame method*), 34
render() (*eyed3.id3.frames.UserUrlFrame method*), 34
render() (*eyed3.id3.headers.ExtendedTagHeader method*), 39
render() (*eyed3.id3.headers.FrameHeader method*), 40
render() (*eyed3.id3.headers.TagHeader method*), 38
REPLAYGAIN_NAME (*eyed3.mp3.headers.LameHeader attribute*), 46
REPLAYGAIN_ORIGINATOR (*eyed3.mp3.headers.LameHeader attribute*), 46
report() (*eyed3.plugins.stats.Stat method*), 59

requireBytes() (*in module eyed3.utils*), 68
requireUnicode() (*in module eyed3.utils*), 68
RESET (*eyed3.utils.console.AnsiBack attribute*), 64
RESET (*eyed3.utils.console.AnsiFore attribute*), 64
RESET_ALL (*eyed3.utils.console.AnsiStyle attribute*), 64
RESET_BLINK_FAST (*eyed3.utils.console.AnsiStyle attribute*), 65
RESET_BLINK_SLOW (*eyed3.utils.console.AnsiStyle attribute*), 65
RESET_BRIGHT (*eyed3.utils.console.AnsiStyle attribute*), 64
RESET_DIM (*eyed3.utils.console.AnsiStyle attribute*), 64
RESET_INVERSE (*eyed3.utils.console.AnsiStyle attribute*), 65
RESET_ITALICS (*eyed3.utils.console.AnsiStyle attribute*), 64
RESET_STRIKE_THRU (*eyed3.utils.console.AnsiStyle attribute*), 65
RESET_UNDERLINE (*eyed3.utils.console.AnsiStyle attribute*), 65
RESTRICT_IMG_ENC_NONE (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_IMG_ENC_PNG_JPG (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_IMG_SZ_256 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_IMG_SZ_64 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_IMG_SZ_64_EXACT (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_IMG_SZ_NONE (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TAG_SZ_LARGE (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TAG_SZ_MED (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TAG_SZ_SMALL (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TAG_SZ_TINY (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_ENC_NONE (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_ENC_UTF8 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_LEN_1024 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_LEN_128 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_LEN_30 (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
RESTRICT_TEXT_LEN_NONE (*eyed3.id3.headers.ExtendedTagHeader attribute*), 38
restrictions_bit() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
rev_version() (*eyed3.id3.headers.TagHeader property*), 38
Rule (*class in eyed3.plugins.stats*), 59
RuleViolationStat (*class in eyed3.plugins.stats*), 61

S

safe_substitute() (*eyed3.id3.tag.TagTemplate method*), 44
SAMPLE_FREQUENCIES (*eyed3.mp3.headers.LameHeader attribute*), 46
save() (*eyed3.id3.tag.Tag method*), 41
second() (*eyed3.core.Date property*), 70
set() (*eyed3.id3.tag.ChaptersAccessor method*), 43
set() (*eyed3.id3.tag.DltAccessor method*), 42
set() (*eyed3.id3.tag.ImagesAccessor method*), 42
set() (*eyed3.id3.tag.ObjectsAccessor method*), 42
set() (*eyed3.id3.tag.PopularitiesAccessor method*), 43
set() (*eyed3.id3.tag.PrivatesAccessor method*), 43
set() (*eyed3.id3.tag.TocAccessor method*), 43
set() (*eyed3.id3.tag.UniqueFileIdAccessor method*), 43
set() (*eyed3.id3.tag.UserTextsAccessor method*), 43
set() (*eyed3.id3.tag.UserUrlsAccessor method*), 43
setFileScannerOpts() (*in module eyed3.main*), 71
setTextFrame() (*eyed3.id3.frames.FrameSet method*), 37
setTextFrame() (*eyed3.id3.tag.Tag method*), 40
SIZE (*eyed3.id3.headers.TagHeader attribute*), 38
size_bytes (*eyed3.core.AudioInfo attribute*), 69
SPECIAL_CHARACTERS (*eyed3.plugins.display.TextPattern attribute*), 49
SPECIAL_CHARACTERS_DESCRIPTIONS (*eyed3.plugins.display.TextPattern attribute*), 49
Spinner (*class in eyed3.utils.console*), 65

splitUnicode() (*in module eyed3.id3.frames*), 37
start() (*eyed3.id3.frames.StartEndTuple property*), 37
start() (*eyed3.plugins.art.ArtPlugin method*), 48
start() (*eyed3.plugins.display.DisplayPlugin method*), 56
start() (*eyed3.plugins.fixup.FixupPlugin method*), 56
start() (*eyed3.plugins.genres.GenreListPlugin method*), 57
start() (*eyed3.plugins.Plugin method*), 62
start() (*eyed3.plugins.pymod.PyModulePlugin method*), 58
StartEndTuple (*class in eyed3.id3.frames*), 37
Stat (*class in eyed3.plugins.stats*), 59
StatisticsPlugin (*class in eyed3.plugins.stats*), 61
STEREO_MODES (*eyed3.mp3.headers.LameHeader attribute*), 46
STRIKE_THRU (*eyed3.utils.console.AnsiStyle attribute*), 65
stringToEncoding() (*in module eyed3.id3.frames*), 38
stringToPicType() (*eyed3.id3.frames.ImageFrame static method*), 35
sub_pattern_classes() (*eyed3.plugins.display.Pattern static method*), 49
sub_patterns() (*eyed3.plugins.display.Pattern property*), 49
substitute() (*eyed3.id3.tag.TagTemplate method*), 44
subtitle() (*eyed3.id3.frames.ChapterFrame property*), 37
SUMMARY (*eyed3.plugins.art.ArtPlugin attribute*), 47
SUMMARY (*eyed3.plugins.classic.ClassicPlugin attribute*), 48
SUMMARY (*eyed3.plugins.display.DisplayPlugin attribute*), 56
SUMMARY (*eyed3.plugins.fixup.FixupPlugin attribute*), 56
SUMMARY (*eyed3.plugins.genres.GenreListPlugin attribute*), 57
SUMMARY (*eyed3.plugins.itunes.Podcast attribute*), 57
SUMMARY (*eyed3.plugins.lameinfo.LameInfoPlugin attribute*), 57
SUMMARY (*eyed3.plugins.nfo.NfoPlugin attribute*), 58
SUMMARY (*eyed3.plugins.Plugin attribute*), 62
SUMMARY (*eyed3.plugins.pymod.PyModulePlugin attribute*), 58
SUMMARY (*eyed3.plugins.stats.StatisticsPlugin attribute*), 61
SUMMARY (*eyed3.plugins.xep_118.Xep118Plugin attribute*), 61
SUPPORTED_AUDIO (*eyed3.plugins.stats.FileCounterStat attribute*), 60
SURROUND_INFO (*eyed3.mp3.headers.LameHeader attribute*), 46

T

table_of_contents() (*eyed3.id3.tag.Tag property*), 42
Tag (*class in eyed3.core*), 69
Tag (*class in eyed3.id3.tag*), 40
tag() (*eyed3.core.AudioFile property*), 70
tag() (*eyed3.mp3.Mp3AudioFile property*), 47
TAG_ALTER (*eyed3.id3.headers.FrameHeader attribute*), 39
tag_alter() (*eyed3.id3.headers.FrameHeader property*), 39
tag_size_restriction() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
tag_size_restriction_description() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
TagException, 40
TagFile (*class in eyed3.id3*), 45
tagging_date() (*eyed3.id3.tag.Tag property*), 41
TaggingDateTagPattern (*class in eyed3.plugins.display*), 51
TagHeader (*class in eyed3.id3.headers*), 38
TagPattern (*class in eyed3.plugins.display*), 49
TagTemplate (*class in eyed3.id3.tag*), 43
TDES (*class in eyed3.id3.apple*), 33
terms_of_use() (*eyed3.id3.tag.Tag property*), 41
TermsOfUseFrame (*class in eyed3.id3.frames*), 36
TermsOfUseTagPattern (*class in eyed3.plugins.display*), 54
test() (*eyed3.plugins.stats.ArtworkRule method*), 59
test() (*eyed3.plugins.stats.BitrateRule method*), 59
test() (*eyed3.plugins.stats.FileRule method*), 59
test() (*eyed3.plugins.stats.Id3FrameRules method*), 59
test() (*eyed3.plugins.stats.Id3TagRules method*), 59
test() (*eyed3.plugins.stats.Rule method*), 59
text() (*eyed3.id3.frames.DescriptionLangTextFrame property*), 36
text() (*eyed3.id3.frames.TermsOfUseFrame property*), 36
text() (*eyed3.id3.frames.TextFrame property*), 33
text_delim() (*eyed3.id3.frames.Frame property*), 33
text_enc_restriction() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
text_enc_restriction_description() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
text_length_restriction() (*eyed3.id3.headers.ExtendedTagHeader property*), 39

text_length_restriction_description() (*eyed3.id3.headers.ExtendedTagHeader property*), 39
TextFrame (*class in eyed3.id3.frames*), 33
TextPattern (*class in eyed3.plugins.display*), 49
TextsTagPattern (*class in eyed3.plugins.display*), 52
TGID (*class in eyed3.id3.apple*), 33
time_secs (*eyed3.core.AudioInfo attribute*), 69
TIME_STAMP_FORMATS (*eyed3.core.Date attribute*), 70
timePerFrame() (*in module eyed3.mp3.headers*), 45
title() (*eyed3.core.Tag property*), 69
title() (*eyed3.id3.frames.ChapterFrame property*), 37
TitleTagPattern (*class in eyed3.plugins.display*), 50
TKWD (*class in eyed3.id3.apple*), 33
TO_ID3_ART_TYPES (*in module eyed3.utils.art*), 63
toc() (*eyed3.id3.frames.MusicCDIdFrame property*), 35
TocAccessor (*class in eyed3.id3.tag*), 43
TocFrame (*class in eyed3.id3.frames*), 36
TOP_LEVEL_FLAG_BIT (*eyed3.id3.frames.TocFrame attribute*), 37
TOTAL (*eyed3.plugins.stats.Stat attribute*), 59
touch() (*eyed3.id3.tag.FileInfo method*), 42
track_num() (*eyed3.core.Tag property*), 69
TrackTagPattern (*class in eyed3.plugins.display*), 50
TrackTotalTagPattern (*class in eyed3.plugins.display*), 50
TYPE (*eyed3.plugins.display.ComplexPattern attribute*), 49
TYPE (*eyed3.plugins.display.FunctionPattern attribute*), 54
TYPE (*eyed3.plugins.display.TagPattern attribute*), 49

U

UNDERLINE (*eyed3.utils.console.AnsiStyle attribute*), 65
uniq_id() (*eyed3.id3.frames.UniqueFileIDFrame property*), 36
unique_file_ids() (*eyed3.id3.tag.Tag property*), 41
UniqueFileIdAccessor (*class in eyed3.id3.tag*), 43
UniqueFileIDFrame (*class in eyed3.id3.frames*), 36
UniqueFileIDTagPattern (*class in eyed3.plugins.display*), 52
UNSUPPORTED_AUDIO (*eyed3.plugins.stats.FileCounterStat attribute*), 60
UNSYNC (*eyed3.id3.headers.FrameHeader attribute*), 39
unsync() (*eyed3.id3.headers.FrameHeader property*), 40
update() (*eyed3.utils.console.ProgressBar method*), 65
update_bit() (*eyed3.id3.headers.ExtendedTagHeader property*), 38
url() (*eyed3.id3.frames.UrlFrame property*), 34
URL_MIME_TYPE (*eyed3.id3.frames.ImageFrame attribute*), 35
URL_MIME_TYPE_STR (*eyed3.id3.frames.ImageFrame attribute*), 35
URL_MIME_TYPE_VALUES (*eyed3.id3.frames.ImageFrame attribute*), 35
UrlFrame (*class in eyed3.id3.frames*), 34
user_text_frames() (*eyed3.id3.tag.Tag property*), 41
user_url() (*eyed3.id3.frames.ChapterFrame property*), 37
user_url_frames() (*eyed3.id3.tag.Tag property*), 41
UserTextFrame (*class in eyed3.id3.frames*), 33
UserTextsAccessor (*class in eyed3.id3.tag*), 43
UserUrlFrame (*class in eyed3.id3.frames*), 34
UserUrlsAccessor (*class in eyed3.id3.tag*), 43
UserURLsTagPattern (*class in eyed3.plugins.display*), 53

V

VBR_METHODS (*eyed3.mp3.headers.LameHeader attribute*), 46
VbriHeader (*class in eyed3.mp3.headers*), 45
verbose() (*eyed3.utils.log.Logger method*), 66
version() (*eyed3.id3.headers.FrameHeader property*), 39
version() (*eyed3.id3.headers.TagHeader property*), 38
version() (*eyed3.id3.tag.Tag property*), 40
versionToString() (*in module eyed3.id3*), 44
VIDEO (*eyed3.id3.frames.ImageFrame attribute*), 34

W

walk() (*in module eyed3.utils*), 67
WARNING_COLOR() (*in module eyed3.utils.console*), 65
WFED (*class in eyed3.id3.apple*), 33
WHITE (*eyed3.utils.console.AnsiBack attribute*), 64
WHITE (*eyed3.utils.console.AnsiFore attribute*), 64
WINAMP_GENRE_MAX (*eyed3.id3.GenreMap attribute*), 45
WINAMP_GENRE_MIN (*eyed3.id3.GenreMap attribute*), 44

X

Xep118Plugin (*class in eyed3.plugins.xep_118*), 61
XingHeader (*class in eyed3.mp3.headers*), 45

Y

year() (*eyed3.core.Date property*), 70
YearTagPattern (*class in eyed3.plugins.display*), 50
YELLOW (*eyed3.utils.console.AnsiBack attribute*), 64
YELLOW (*eyed3.utils.console.AnsiFore attribute*), 64