
metalparser

Release 0.6.9b1

Luca Ballore

Jan 24, 2020

HANDBOOK

1	User guide	3
1.1	Description	3
1.2	Installation	3
1.3	Documentation	3
1.3.1	Some examples	4
1.3.1.1	Retrieve the lyrics given a song and the corresponding artist	4
1.3.1.2	Get all the songs of a specific album	4
1.3.1.3	Get all the albums of a specific artist	4
1.4	Support	4
1.5	Thanks to	5
2	Package <i>metaparser</i>	7
2.1	Subpackages	7
2.1.1	Package <i>metaparser.common</i>	7
2.1.1.1	Subpackages	7
2.1.1.1.1	Package <i>metaparser.common.resources</i>	7
2.1.1.2	Module <i>metaparser.common.exceptions</i>	7
2.1.1.3	Module <i>metaparser.common.logger</i>	7
2.1.1.4	Module <i>metaparser.common.scraping</i>	8
2.1.2	Package <i>metaparser.libs</i>	8
2.1.2.1	Module <i>metaparser.libs.darklyrics_utils</i>	8
2.2	Module <i>metaparser.darklyrics</i>	10
3	Indices and tables	13
	Python Module Index	15
	Index	17

“A Python library for heavy metal song lyrics, albums, song titles and other info.”

USER GUIDE

1.1 Description

metaparser is a Python API for obtaining song lyrics from diverse lyrics websites. At the moment there is only one supported website, which is [DarkLyrics](#), an online database of lyrics for heavy metal music.

This library scrapes the corresponding website for the lyrics and returns results according to the used API. Kindly read the [disclaimer](#) to ensure that your use complies with it.

1.2 Installation

metaparser is distributed as a Python package, freely available on [PyPI](#) and can easily be installed via pip. Given that you are using `python >= 3.5`:

```
pip install metaparser
```

Alternatively, it can be manually installed by cloning this project on your local computer:

```
git clone https://github.com/lucone83/metal-parser.git
cd metal-parser
pip install .
```

1.3 Documentation

The library comes (at the moment) with 6 APIs:

- `get_artists_list()`
- `get_albums_info()`
- `get_songs_info()`
- `get_album_info_and_lyrics()`
- `get_albums_info_and_lyrics_by_artist()`
- `get_song_info_and_lyrics()`

1.3.1 Some examples

I recommend not to change the default settings regarding requests rate per minute and the wait time (3 secs) after each request. DarkLyrics does not have a robots.txt, so they don't really like scraping. Be gentle! :)

```
from metaparser.darklyrics import DarkLyricsApi

api = DarkLyricsApi()
```

1.3.1.1 Retrieve the lyrics given a song and the corresponding artist

```
song = 'under grey skies'
artist = 'kamelot'
lyrics = api.get_song_info_and_lyrics(song=song, artist=artist, lyrics_only=True)

print(lyrics)
```

1.3.1.2 Get all the songs of a specific album

```
artist = 'pantera'
album = 'vulgar display of power'
songs_list = api.get_songs_info(artist, album=album, title_only=True)

print(songs_list)
```

1.3.1.3 Get all the albums of a specific artist

```
artist = 'iron maiden'
albums_list = api.get_albums_info(artist=artist, title_only=True)

print(albums_list)
```

1.4 Support

Currently the following python versions are supported:

- 3.4.*
- 3.5.*
- 3.6.*
- 3.7.*
- 3.8.*

1.5 Thanks to

- res0nance and his [darklyrics](#) project for inspiration;

PACKAGE METALPARSER

2.1 Subpackages

2.1.1 Package *metaparser.common*

2.1.1.1 Subpackages

2.1.1.1.1 Package *metaparser.common.resources*

2.1.1.2 Module *metaparser.common.exceptions*

exception *metaparser.common.exceptions.ArtistNotFoundException* (*message='Error'*)
Bases: *metaparser.common.exceptions.MetalParserException*

exception *metaparser.common.exceptions.LyricsNotFoundException* (*message='Error'*)
Bases: *metaparser.common.exceptions.MetalParserException*

exception *metaparser.common.exceptions.MetalParserException* (*message='Error'*)
Bases: *Exception*

exception *metaparser.common.exceptions.SongsNotFoundException* (*message='Error'*)
Bases: *metaparser.common.exceptions.MetalParserException*

2.1.1.3 Module *metaparser.common.logger*

class *metaparser.common.logger.MetalParserLogger* (*debug_mode*)
Bases: *object*

Instantiate a logging.Logger object.

Parameters *debug_mode* (*bool*) – Boolean defining if the logging level (DEBUG if True, ERROR if False).

logger
The logging.Logger object initialized.

Type *Logger*

get_logger (*self*)
Returns the logger attribute.

get_logger ()
Returns the logger attribute.

Returns [*Logger*] – The logging.Logger object initialized.

2.1.1.4 Module *metaparser.common.scraping*

```
class metaparser.common.scraping.ScrapingAgent (use_cache=True)
    Bases: object

    Instantiate an object with cached and uncached web crawling functions.

        Parameters use_cache (bool) – Boolean defining if a cached session will be created or not

    cache_expires_after
        Expiring time for cached contents

        Type int

    cached_session
        Object instantiating a cached session for requests

        Type CachedSession

    get_page_from_url (self, url)
        Returns a DarkLyrics.com page related to an artist in form of a BeautifulSoup object.

    get_cached_session (self)
        Returns the cached_session attribute.

    get_last_response (self)
        Returns the last Response object corresponding to the last request made by the ScrapingAgent.

    get_cached_session ()
        Returns the cached_session attribute.

        Returns [CachedSession or None] – The CachedSession object instantiated when initializing
        the object class.

    get_last_response ()
        Returns the last Response object corresponding to the last request made by the ScrapingAgent.

        Returns [Response or None] – The Response object corresponding to the last request made by
        the ScrapingAgent.

    get_page_from_url (url)
        Returns a DarkLyrics.com page related to an artist in form of a BeautifulSoup object.

        Parameters {str} -- A string containing an URL (url) –

        Returns [BeautifulSoup] – An HTML page related to the specified URL in form of a Beautiful-
        Soup object
```

2.1.2 Package *metaparser.libs*

2.1.2.1 Module *metaparser.libs.darklyrics_utils*

```
class metaparser.libs.darklyrics_utils.DarkLyricsHelper (use_cache)
    Bases: object

    A class with helpers for DarkLyricsApi

    BASE_URL
        DarkLyrics.com base URL

        Type str
```

scraping_agent

The agent taking hand of HTTP requests

Type *ScrapingAgent*

get_base_url (*self*)

Returns DarkLyrics.com base URL.

get_artist_page (*self*, *artist*)

Returns a DarkLyrics.com page related to an artist in form of a BeautifulSoup object.

get_songs_links_from_artist (*self*, *artist*, *album=None*)

Returns a links list containing all the lyrics URLs related to an artist or an album.

get_albums_info_from_artist_page (*self*, *artist_page*, *all_info=False*):

Given the artist page, returns infos about the albums.

get_albums_info_from_url (*self*, *url*):

Returns album info given the album's URL.

get_lyrics_url_by_song (*self*, *song*, *artist*)

Given a song title and the artist, returns the link related to the lyrics.

get_lyrics_url_by_tag (*self*, *link_tag*)

Given an <a> HTML tag related to a song's lyrics, returns the related URL.

get_lyrics_by_url (*self*, *url*)

Given an URL related to a song, returns the lyrics.

get_albums_info_from_artist_page (*artist_page*, *title_only=False*)

Given the artist page, retrieve infos about the albums.

Parameters {BeautifulSoup} -- The artist page in BeautifulSoup format. (*artist_page*) –

Keyword Arguments {bool} -- Flag to determinate if returning all albums info or title only (default (*all_info*) – {False})

Returns [list] – List of albums (str list or dict list, depending on *all_info*)

get_albums_info_from_url (*url*)

Returns album info given the album's URL.

Parameters {str} -- The album's URL (*url*) –

Returns title, release year and type (album, EP).

Return type [dict] – A dict with the following album info

get_artist_page (*artist*)

Returns a DarkLyrics.com page related to an artist in form of a BeautifulSoup object.

Parameters {str} -- The artist's name (*artist*) –

Raises *ArtistNotFoundException* – Exception raised when the URL is not found on DarkLyrics.com

Returns [BeautifulSoup] – Page related to an artist in form of a BeautifulSoup object

get_base_url ()

Returns DarkLyrics.com base URL.

Returns [str] – DarkLyrics.com base URL

get_lyrics_by_url (*url*)

Given an URL related to a song, returns the lyrics.

Parameters {str} -- URL leading to the lyrics of a certain song
(url) –

Raises *LyricsNotFoundException* – Exception raised when no lyrics div is found

Returns [str] – A string with the lyrics related to the specified URL

get_lyrics_url_by_song (song, artist)

Given a song title and the artist, returns the link related to the lyrics.

Parameters

- {str} -- The title of the song (song) –
- {str} -- The artist's name (artist) –

Raises *LyricsNotFoundException* – Exception raised when no link is found

Returns [str] – The link related to the lyrics of the specified song

get_lyrics_url_by_tag (link_tag)

Given an <a> HTML tag related to a song's lyrics, returns the related URL.

Parameters {BeautifulSoup} -- <a> tag which is supposed to contain an URL related to lyrics (link_tag) –

Raises *LyricsNotFoundException* – Exception raised when no link or invalid link is found

Returns [str] – URL string contained in the specified <a> tag, leading to lyrics.

get_songs_links_from_artist (artist, album=None)

Returns a links list containing all the lyrics URLs related to an artist or an album.

Parameters {str} -- The artist's name (artist) –

Keyword Arguments {str} -- The title of the album (album) – {None}}

Raises *SongsNotFoundException* – Exception raised when no songs related to an artist or album are found

Returns [list] – List of strings containing all the lyrics URLs related to an artist or an album

2.2 Module *metaparser.darklyrics*

class metaparser.darklyrics.**DarkLyricsApi** (use_cache=True, debug_mode=False)

Bases: object

A class with APIs for scraping DarkLyrics.com website.

Parameters

- **use_cache** (bool) – Boolean defining if a cached session will be created or not.
- **debug_mode** (bool) – Boolean defining when to save debug info on a log file.

helper

Object containing helpers for DarkLyrics.com APIs.

Type *DarkLyricsHelper*

get_artists_list (self, initial_letter=None)

Returns a list with all the artists registered on DarkLyrics.com. When specified, it returns a list of artists starting with an initial.

get_albums_info (*self, artist, title_only=False*)

Returns a list containing all the albums titles related to an artist.

get_songs_info (*self, artist, album=None, title_only=False*)

Returns a list containing the songs titles (and other info when specified) related to a single artist or album (when specified).

get_album_info_and_lyrics (*self, album, artist*)

Returns a list of dict containing name, title, album, track number and lyrics of all the songs related to an album on DarkLyrics.com.

get_albums_info_and_lyrics_by_artist (*self, artist*)

Returns a list of dict containing name, title, album, track number and lyrics of all the songs related to an artist on DarkLyrics.com.

def get_song_info_and_lyrics(self, song, artist)

Returns a str containing the lyrics of the specified song.

get_album_info_and_lyrics (*album, artist, lyrics_only=False*)

Returns a list of dict containing info and lyrics of all the songs related to an album on DarkLyrics.com.

Parameters

- {str} -- The title of the album (*album*) –
- {str} -- The artist's name (*artist*) –

Returns

[list] – A list of dict containing info and lyrics about of all the songs related to the specified album or a list of str containing only the lyrics of the specified album, depending on the lyrics_only flag.

get_albums_info (*artist, title_only=False*)

Returns a list containing all the albums titles related to an artist.

Parameters {str} -- The artist's name (*artist*) –

Returns [list] – A list of str containing all the albums titles related to an artist

get_albums_info_and_lyrics_by_artist (*artist*)

Returns a list of dict containing name, title, album, track number and lyrics of all the songs related to an artist on DarkLyrics.com.

Parameters {str} -- The artist's name (*artist*) –

Returns [list] – A list of dict containing info and lyrics of all the songs related to the specified artist.

get_artists_list (*initial_letter=None*)

Returns a list with all the artists registered on DarkLyrics.com. When specified, it returns a list of artists starting with an initial.

Keyword Arguments {str} -- The initial letter of The artist's name (*initial_letter*) – {None}

Raises ValueError – Exception raised when the argument initial_letter is longer than 1 (when specified)

Returns [list] – An alphabetically ordered list of str containing all the artists found according to the arguments

get_song_info_and_lyrics (*song, artist, lyrics_only=False*)

Returns a str containing the lyrics of the specified song.

Parameters

- **{str}** -- The title of the song (*song*) –
- **{str}** -- The artist's name (*artist*) –

Returns

[dict or str] – A dict containing info and lyrics about a song of a certain artist or a str containing only the lyrics of the specified song, depending on the `lyrics_only` flag.

get_songs_info (*artist*, *album=None*, *title_only=False*)

Returns a list containing the songs titles related to a single artist or album (when specified).

Parameters **{str}** -- The artist's name (*artist*) –

Keyword Arguments

- **{bool}** -- (*songs_only*) –
- **{str}** -- The album name (*album*) – {None}

Returns **[list]** – A list of str containing the songs titles related to a single artist or album (when specified)

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

m

- `metaparser`, [7](#)
- `metaparser.common`, [7](#)
- `metaparser.common.exceptions`, [7](#)
- `metaparser.common.logger`, [7](#)
- `metaparser.common.resources`, [7](#)
- `metaparser.common.scraping`, [8](#)
- `metaparser.darklyrics`, [10](#)
- `metaparser.libs`, [8](#)
- `metaparser.libs.darklyrics_utils`, [8](#)

INDEX

A

ArtistNotFoundException, 7

B

BASE_URL (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* attribute), 8

C

cache_expires_after (*metaparser.common.scraping.ScrapingAgent* attribute), 8

cached_session (*metaparser.common.scraping.ScrapingAgent* attribute), 8

D

DarkLyricsApi (*class in metaparser.darklyrics*), 10

DarkLyricsHelper (*class in metaparser.libs.darklyrics_utils*), 8

G

get_album_info_and_lyrics() (*metaparser.darklyrics.DarkLyricsApi* method), 11

get_albums_info() (*metaparser.darklyrics.DarkLyricsApi* method), 10, 11

get_albums_info_and_lyrics_by_artist() (*metaparser.darklyrics.DarkLyricsApi* method), 11

get_albums_info_from_artist_page() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9

get_albums_info_from_url() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9

get_artist_page() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9

get_artists_list() (*metaparser.darklyrics.DarkLyricsApi* method), 10, 11

get_base_url() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9

get_cached_session() (*metaparser.common.scraping.ScrapingAgent* method), 8

get_last_response() (*metaparser.common.scraping.ScrapingAgent* method), 8

get_logger() (*metaparser.common.logger.MetalParserLogger* method), 7

get_lyrics_by_url() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9

get_lyrics_url_by_song() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9, 10

get_lyrics_url_by_tag() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9, 10

get_page_from_url() (*metaparser.common.scraping.ScrapingAgent* method), 8

get_song_info_and_lyrics() (*metaparser.darklyrics.DarkLyricsApi* method), 11

get_songs_info() (*metaparser.darklyrics.DarkLyricsApi* method), 11, 12

get_songs_links_from_artist() (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* method), 9, 10

H

helper (*metaparser.darklyrics.DarkLyricsApi* attribute), 10

L

logger (*metaparser.common.logger.MetalParserLogger* attribute), 7

LyricsNotFoundException, 7

M

metaparser (*module*), 7
metaparser.common (*module*), 7
metaparser.common.exceptions (*module*), 7
metaparser.common.logger (*module*), 7
metaparser.common.resources (*module*), 7
metaparser.common.scraping (*module*), 8
metaparser.darklyrics (*module*), 10
metaparser.libs (*module*), 8
metaparser.libs.darklyrics_utils (*module*), 8
MetalParserException, 7
MetalParserLogger (*class* in *metaparser.common.logger*), 7

S

scraping_agent (*metaparser.libs.darklyrics_utils.DarkLyricsHelper* attribute), 8
ScrapingAgent (*class* in *metaparser.common.scraping*), 8
SongsNotFoundException, 7