LyricsMaster Documentation

Release 2.8.1

SekouD

Apr 07, 2019

Contents

Contents:

CHAPTER 1

LyricsMaster

LyricsMaster is a library for downloading lyrics from multiple lyrics providers.

The following Lyrics Providers are supported:

- Lyric Wikia
- AzLyrics
- Genius
- Lyrics007
- MusixMatch
- The Original Hip-Hop (Rap) Lyrics Archive - OHHLA.com
- and more to come soon.
- Free software: MIT license
- Documentation: https://lyricsmaster.readthedocs.io.

1.1 Features

- Download Lyrics from LyricWiki, AzLyrics, Genius, Lyrics007, MusixMatch, OHHLA and more.
- Download Lyrics Asynchronously.
- Can make requests over Tor for anonymous downloading of songs.

- Easily save the lyrics on your computer.

CHAPTER 2

Installation

2.1 Stable release

To install LyricsMaster, run this command in your terminal:

```
$ pip install lyricsmaster
```

This is the preferred method to install LyricsMaster, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for LyricsMaster can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/SekouD/lyricsmaster
```

Or download the tarball:

```
$ curl  -OL https://github.com/SekouD/lyricsmaster/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

To use LyricsMaster in a project:

```
from lyricsmaster import LyricWiki, TorController


# Select a provider from the supported Lyrics Providers (LyricWiki, AzLyrics, Genius␣
↪etc..)
# The default Provider is LyricWiki
provider = LyricWiki()

# Fetch all lyrics from 2Pac
discography = provider.get_lyrics('2Pac')

# Discography Objects and Album Objects can be iterated over.
for album in discography:    # album is an Album Object.
    print('Album: ', album.title)
    for song in album:       # song is a Song Object.
        print('Song: ', song.title)
        print('Lyrics: ', song.lyrics)

# New indexing and slicing support of Discography and Album Objects
first_song_of_first_album = discography.albums[0].songs[0]
lat_two_songs_of_first_album = discography.albums[0].songs[-2:]

# Fetch all lyrics from 2pac's album 'All eyez on me'.
album = provider.get_lyrics('2Pac', album='All eyes on me')

# Fetch the lyrics from the song 'California Love' in 2pac's album 'All eyez on me'.
song = provider.get_lyrics('2Pac', album='All eyez on me', song='California Love)

# Once the lyrics are fetched, you can save them on disk.
# The 'save()' method is implemented for Discography, Album and Song objects.
# By default, the lyrics are saved in {user}/Documents/lyricsmaster/
discography.save()
```

(continues on next page)

(continued from previous page)

```
# You can also supply a folder to save the lyrics in.
folder = 'c:\MyFolder'
discography.save(folder)

# For anonymity, you can use a Tor Proxy to make requests.
# The TorController class has the same defaults as a default Tor Install.
provider = LyricWiki(TorController())
discography = provider.get_lyrics('2Pac')

# For enhanced anonymity, the TorController can renew the the Tor ciruit for each␣
↪album dowloaded.
# For this functionnality to work, the Tor ControlPort option must be enabled in your␣
↪torrc config file.
# See https://www.torproject.org/docs/tor-manual.html.en for more information.
provider = LyricWiki(TorController(control_port=9051, password='password'))
discography = provider.get_lyrics('2Pac')
```

To use LyricsMaster from the command line (The default Lyrics Provider is LyricWiki):

```
$ lyricsmaster <artist_name> options
```

Examples:

```
$ lyricsmaster "2Pac"
Anonymous requests disabled. The connexion will not be anonymous.
Downloading 2Pacalypse Now (1991)
2Pacalypse Now (1991) succesfully downloaded
Downloading Strictly 4 My N.I.G.G.A.Z... (1993)
Strictly 4 My N.I.G.G.A.Z... (1993) succesfully downloaded
Downloading Thug Life - Volume 1 (1994)
...


$ lyricsmaster "2Pac" --provider Genius
Anonymous requests disabled. The connexion will not be anonymous.
Downloading The Rose That Grew From Concrete (Book)
The Rose That Grew From Concrete (Book) succesfully downloaded
Downloading Best of 2Pac Part 2: Life
Best of 2Pac Part 2: Life succesfully downloaded
...


$ lyricsmaster "2Pac" --tor 127.0.0.1
Anonymous requests enabled. The Tor circuit will change according to the Tor network␣
↪defaults.
Downloading 2Pacalypse Now (1991)
2Pacalypse Now (1991) succesfully downloaded
Downloading Strictly 4 My N.I.G.G.A.Z... (1993)
Strictly 4 My N.I.G.G.A.Z... (1993) succesfully downloaded
Downloading Thug Life - Volume 1 (1994)
...


$ lyricsmaster "2Pac" --tor 127.0.0.1 --controlport 9051 --password password
Anonymous requests enabled. The Tor circuit will change for each album.
New Tor circuit created
```

(continues on next page)

(continued from previous page)

```
Downloading 2Pacalypse Now (1991)
2Pacalypse Now (1991) succesfully downloaded
New Tor circuit created
Downloading Strictly 4 My N.I.G.G.A.Z... (1993)
Strictly 4 My N.I.G.G.A.Z... (1993) succesfully downloaded
New Tor circuit created
Downloading Thug Life - Volume 1 (1994)
...
```

CHAPTER 4

Package Api Documentation

4.1 API Reference for classes in lyricsmaster.providers

Main module.

This module defines the Api interface for the various Lyrics providers. All lyrics providers inherit from the base class LyricsProvider.

class lyricsmaster.providers.**LyricsProvider**(*tor_controller=None*)

Bases: object

This is the base class for all Lyrics Providers. If you wish to subclass this class, you must implement all the methods defined in this class to be compatible with the LyricsMaster API. Requests to fetch songs are executed asynchronously for better performance. Tor anonymisation is provided if tor is installed on the system and a TorController is passed at instance creation.

Parameters **tor_controller** – TorController Object.

get_albums(*raw_artist_page*)

Must be implemented by children classes conforming to the LyricsMaster API.

Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)

Must be implemented by children classes conforming to the LyricsMaster API.

Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)

Must be implemented by children classes conforming to the LyricsMaster API.

Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Must be implemented by children classes conforming to the LyricsMaster API.

Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Must be implemented by children classes conforming to the LyricsMaster API.

Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string or None. Formatted lyrics.

extract_writers(*lyrics_page*)
Must be implemented by children classes conforming to the LyricsMaster API.

Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string or None. Song writers.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.

- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

class lyricsmaster.providers.**LyricWiki**(*tor_controller=None*)
Bases: *lyricsmaster.providers.LyricsProvider*

Class interfacing with http://lyrics.wikia.com . This class is used to retrieve lyrics from LyricWiki.

get_album_page(*artist*, *album*)
Fetches the album page for the supplied artist and album.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.

Returns string or None. Album's raw html page. None if the album page was not found.

get_albums(*raw_artist_page*)
Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)
Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)
Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string or None. Formatted lyrics.

extract_writers(*lyrics_page*)
Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string or None. Song writers.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

class lyricsmaster.providers.**AzLyrics**(*tor_controller=None*)
Bases: *lyricsmaster.providers.LyricsProvider*

Class interfacing with https://azlyrics.com . This class is used to retrieve lyrics from AzLyrics.

search(*artist*)
Searches for the artist in the supplier's database.

Parameters **artist** – Artist's name.

Returns url or None. Url to the artist's page if found. None if not Found.

get_albums(*raw_artist_page*)
Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)
Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)
Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Formatted lyrics.

extract_writers(*lyrics_page*)
Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string or None. Song writers or None.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

class lyricsmaster.providers.**Genius**(*tor_controller=None*)
Bases: *lyricsmaster.providers.LyricsProvider*

Class interfacing with https://genius.com . This class is used to retrieve lyrics from Genius.

get_albums(*raw_artist_page*)
Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)
Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)
Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Formatted lyrics.

extract_writers(*lyrics_page*)
Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Song writers or None.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

class lyricsmaster.providers.**Lyrics007**(*tor_controller=None*)
Bases: *lyricsmaster.providers.LyricsProvider*

Class interfacing with https://www.lyrics007.com . This class is used to retrieve lyrics from Lyrics007.

search(*artist*)
Searches for the artist in the supplier's database.

Parameters **artist** – string. Artist's name.

Returns string or None. Artist's url page.

get_albums(*raw_artist_page*)
Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)
Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)
Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Formatted lyrics.

extract_writers(*lyrics_page*)
Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Song writers or None.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

class lyricsmaster.providers.**MusixMatch**(*tor_controller=None*)
Bases: *lyricsmaster.providers.LyricsProvider*

Class interfacing with https://www.musixmatch.com . This class is used to retrieve lyrics from MusixMatch.

get_albums(*raw_artist_page*)
Fetches the albums section in the supplied html page.

Parameters **raw_artist_page** – Artist's raw html page.

Returns list. List of BeautifulSoup objects.

get_album_infos(*tag*)
Extracts the Album informations from the tag

Parameters **tag** – BeautifulSoup object.

Returns tuple(string, string). Album title and release date.

get_songs(*album*)
Fetches the links to the songs of the supplied album.

Parameters **album** – BeautifulSoup object.

Returns List of BeautifulSoup Link objects.

create_song(*link*, *artist*, *album_title*)
Creates a Song object.

Parameters

- **link** – BeautifulSoup Link object.
- **artist** – string.
- **album_title** – string.

Returns models.Song object or None.

extract_lyrics(*lyrics_page*)
Extracts the lyrics from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Formatted lyrics.

extract_writers(*lyrics_page*)
Extracts the writers from the lyrics page of the supplied song.

Parameters **lyrics_page** – BeautifulSoup Object. BeautifulSoup lyrics page.

Returns string. Song writers or None.

get_artist_page(*artist*)
Fetches the web page for the supplied artist.

Parameters **artist** – string. Artist name.

Returns string or None. Artist's raw html page. None if the artist page was not found.

get_lyrics(*artist*, *album=None*, *song=None*)
This is the main method of this class. Connects to the Lyrics Provider and downloads lyrics for all the albums of the supplied artist and songs. Returns a Discography Object or None if the artist was not found on the Lyrics Provider.

Parameters

- **artist** – string. Artist name.
- **album** – string. Album title.
- **song** – string. Song title.

Returns models.Discography object or None.

get_lyrics_page(*url*)
Fetches the web page containing the lyrics at the supplied url.

Parameters **url** – string. Lyrics url.

Returns string or None. Lyrics's raw html page. None if the lyrics page was not found.

get_page(*url*)
Fetches the supplied url and returns a request object.

Parameters **url** – string.

Returns urllib3.response.HTTPResponse Object or None.

4.2 API Reference for classes in lyricsmaster.models

Models the domain objects.

Defines classes for Song, Album and Discography.

class lyricsmaster.models.**Song**(*title*, *album*, *artist*, *lyrics=None*, *writers=None*)
Song class.

Parameters

- **title** – string. Song title.
- **album** – string. Album title.
- **artist** – string. Author name.
- **lyrics** – string. Lyrics of the song.
- **writers** – string. List of the song's writers.

save(*folder=None*)
Saves the lyrics of the song in the supplied folder. If no folder is supplied, 'folder' is set to {user}/Documents/lyricsmaster/ The lyrics of a song are saved in the folder /artist/album/song_title.txt

Parameters **folder** – string. path to save folder.

class lyricsmaster.models.**Album**(*title*, *artist*, *songs*, *release_date='Unknown'*)
Album Class. The Album class follows the Iterable protocol and can be iterated over the songs.

Parameters

- **title** – string. Album title.
- **artist** – string. Artist name.
- **release_date** – string. Release date.
- **songs** – list. List of Songs objects.

save(*folder=None*)
Saves the album in the supplied folder.

Parameters **folder** – string. path to save folder.

class lyricsmaster.models.**Discography**(*artist*, *albums*)
Discography Class. The Discography class follows the Iterable protocol and can be iterated over the albums.

Parameters

- **artist** – string. Artist name.
- **albums** – list. List of Album objects.

save(*folder=None*)
Saves Discography in the supplied folder.

Parameters **folder** – string. Path to save folder.

4.3 API Reference for classes in lyricsmaster.utils

class lyricsmaster.utils.**TorController**(*ip='127.0.0.1'*, *socksport=9050*, *controlport=None*, *password=''*)
Controller class for Tor client.

Allows the Api to make requests over the Tor network. If 'controlport' is None, the library will use the default timing to renew the Tor circuit. If 'controlport' is passed as an argument, the library will create a new Tor circuit for each new album downloaded. See https://www.torproject.org/docs/tor-manual.html.en for more information on how Tor works.

Parameters

- **ip** – string. The IP adress of the Tor proxy.
- **socksport** – integer. The SOCKSPORT port number for Tor.
- **controlport** – integer or string. The CONTROLPORT port number for Tor or the unix path to the CONTROLPATH.
- **password** – string. The password or control_auth_cookie to authenticate on the Tor CONTROLPORT.

get_tor_session()

Configures and create the session to use a Tor Socks proxy.

Returns urllib3.SOCKSProxyManager object.

renew_tor_circuit()

Renews the Tor circuit. Sends a NEWNYM message to the Tor network to create a new circuit.

Returns bool. Whether a new tor ciruit was created.

CHAPTER 5

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:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/SekouD/lyricsmaster/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.

5.1.2 Fix Bugs

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

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

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

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/SekouD/lyricsmaster/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 :)

5.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/lyricsmaster.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 lyricsmaster
$ cd lyricsmaster/
$ python setup.py develop
```

4. Create a branch for local development:

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

   Now you can make your changes locally.

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

```
$ flake8 lyricsmaster tests
$ python setup.py test or py.test
$ tox
```

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

6. 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
```

7. Submit a pull request through the GitHub website.

5.3 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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/SekouD/lyricsmaster/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_lyricsmaster
```

CHAPTER 6

Credits

6.1 Development Lead

- SekouD <sekoud.python@gmail.com>
- GPG key ID: B51D1046EF63C50B

https://github.com/SekouD/

6.2 Contributors

- The logo was designed by estribiyo.

CHAPTER 7

History

7.1 2.8.1 (2019-04-07)

- Implemented more fully the Python Data Model for the Discography and Album classes.
- Now individual albums or songs in a Discography object can be individually accessed by indexing or slicing on top of previously being iterable.
- For example Discography.albums[0].songs[0] or Discography.albums[0].songs[2:5]
- Updated dependencies.

7.2 2.8 (2019-03-31)

- Enhanced support for utf-8 characters even when the html encoding has wrong information. (see https://github.com/SekouD/lyricsmaster/issues/211)
- Fixed AzLyrics bug when Artist had song but no album on the service.
- Updated dependencies.

7.3 2.7.25 (2019-03-23)

- Enhanced support for utf-8 characters even when the html encoding has wrong information. (see https://github.com/SekouD/lyricsmaster/issues/211)
- Fixed MusixMatch bug when only the first sentence of some lyrics was parsed.
- updated logger configuration to avoid repeated logs when lyricsmaster was used as a library instead of standalone.
- Updated dependencies.

7.4 2.7.24 (2019-03-16)

- Fixed bug when trying to download lyrics from urls containing unicode characters. (see https://github.com/SekouD/lyricsmaster/issues/211)
- Replaced use of print() with Python logging facilities.
- Updated dependencies.

7.5 2.7.23 (2019-02-28)

- Updated lyricsmaster to reflect changes in MusixMatch and Lyrics007 APIs.
- Updated dependencies.

7.6 2.7.22 (2018-11-18)

- Updated dependencies.

7.7 2.7.21 (2018-11-04)

- Updated to latest tor version.
- Updated dependencies.

7.8 2.7.20 (2018-07-29)

- Updated to latest tor version.
- Updated documentation.

7.9 2.7.19 (2018-07-16)

- Catch exceptions when the release date of the album is not in the title tag for all providers.

7.10 2.7.17 (2018-07-08)

- Improved Tests.
- Updated documentation.

7.11 2.7.16 (2017-09-27)

- General improvements.

7.12 2.7.0 (2017-09-27)

- Added Command Line Interface.

7.13 2.6.0 (2017-09-26)

- Added Genius provider.

7.14 2.5.0 (2017-09-26)

- Added python 2.7 compatibility

7.15 2.4.0 (2017-09-24)

- Added AzLyrics provider.

7.16 2.3.0 (2017-09-21)

- Added full documentation.
- Corrected asynchronous requests bug when renewing Tor circuit.

7.17 2.2.0 (2017-09-20)

- Added save method to Discography, Album, Song objects.

7.18 2.1.0 (2017-09-20)

- Added Asynchronous Requests.

7.19 2.0.0 (2017-09-19)

- Added Tor Anonymisation.

7.20 1.0.0 (2017-09-17)

- Added LyricWiki provider.

7.21 0.1.0 (2017-09-11)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

l

Index

L

M

R

S

T