pylistenbrainz

Release unknown

Param Singh

Jun 29, 2022

CONTENTS

1 Features 3

2 Installation 5

3 Getting Started 7

4 Examples 9

5 API Reference 11

6 Exceptions 13

7 Support 15

8 License 17

9 Table Of Contents 19
- 9.1 API Reference . . . 19
- 9.2 Exceptions . . . 23

10 Indices and tables 25

Python Module Index 27

Index 29

pylistenbrainz is a simple Python library for the ListenBrainz Web API. pylistenbrainz should help you start getting data from and submitting data to ListenBrainz very quickly.

Here's an example of getting the listening history of a ListenBrainz user:

```
import pylistenbrainz

client = pylistenbrainz.ListenBrainz()
listens = client.get_listens(username='iliekcomputers')
for listen in listens:
    print("Track name:", listen.track_name)
    print("Artist name:", listen.artist_name)
```

Here's another quick example of how to submit a listen to ListenBrainz:

```
import pylistenbrainz
import time

auth_token = input('Please enter your auth token: ')

listen = pylistenbrainz.Listen(
    track_name="Fade",
    artist_name="Kanye West",
    release_name="The Life of Pablo",
    listened_at=int(time.time()),
)

client = pylistenbrainz.ListenBrainz()
client.set_auth_token(auth_token)
response = client.submit_single_listen(listen)
```

CHAPTER

ONE

FEATURES

pylistenbrainz provides easy access to all ListenBrainz endpoints, handles ratelimits automatically and supports the ListenBrainz authorization flow.

For details on the API endpoints that can be used via pylistenbrainz, take a look at the ListenBrainz API Documentation.

CHAPTER

TWO

INSTALLATION

Install or upgrade pylistenbrainz with:

```
pip install pylistenbrainz --upgrade
```

Or you can get the source code from GitHub at https://github.com/paramsingh/pylistenbrainz.

CHAPTER

THREE

GETTING STARTED

It is easy to get started retrieving data from ListenBrainz using pylistenbrainz. No authentication is required for getting data.

To submit data for a user, pylistenbrainz requires that you have the user's ListenBrainz auth token. Each user has a unique auth token available on their profile page.

You can optionally set an auth token for requests to get data as well.

Here's an example of setting an auth token to a pylistenbrainz client:

```
import pylistenbrainz

auth_token = input('Please enter your auth token: ')
client = pylistenbrainz.ListenBrainz()
pylistenbrainz.set_auth_token(auth_token)
```

By default, the `set_auth_token` method checks for the validity of the auth token by making a request to the ListenBrainz API. You can skip this check using the `check_validity` param. For example:

```
import pylistenbrainz

auth_token = input('Please enter your auth token: ')
client = pylistenbrainz.ListenBrainz()
pylistenbrainz.set_auth_token(auth_token, check_validity=False)
```

CHAPTER

FOUR

EXAMPLES

There are more examples of how to use pylistenbrainz in the examples directory on GitHub.

CHAPTER

FIVE

API REFERENCE

There are more details about the client interface on the *API reference page*.

CHAPTER

SIX

EXCEPTIONS

All exceptions raised by pylistenbrainz should inherit from the base class `pylistenbrainz.errors.ListenBrainzException`.

For a comprehensive list of exceptions that the library can raise, take a look at the *exceptions page*.

CHAPTER

SEVEN

SUPPORT

You can ask questions about how to use pylistenbrainz on IRC (freenode #metabrainz). You can also email me at `iliekcomputers [at] gmail [dot] com`.

If you have found a bug or have a feature request, let me know by opening a GitHub Issue.

CHAPTER

EIGHT

LICENSE

https://github.com/paramsingh/pylistenbrainz/blob/master/LICENSE

CHAPTER

NINE

TABLE OF CONTENTS

9.1 API Reference

9.1.1 ListenBrainz client

The `ListenBrainz` class is the main interface provided by pylistenbrainz. It can be used to interact with the ListenBrainz API.

class pylistenbrainz.client.**ListenBrainz**

Bases: `object`

get_listens(*username*, *max_ts=None*, *min_ts=None*, *count=None*)

Get listens for user *username*

If none of the optional arguments are given, this function will return the 25 most recent listens. The optional *max_ts* and *min_ts* UNIX epoch timestamps control at which point in time to start returning listens. You may specify max_ts or min_ts, but not both in one call.

Parameters

- **username** (*str*) – the username of the user whose data is to be fetched
- **max_ts** (*int, optional*) – If you specify a max_ts timestamp, listens with listened_at less than (but not including) this value will be returned.
- **min_ts** (*int, optional*) – If you specify a min_ts timestamp, listens with listened_at greater than (but not including) this value will be returned.
- **count** (*int, optional*) – the number of listens to return. Defaults to 25, maximum is 100.

Returns A list of listens for the user *username*

Return type List[*pylistenbrainz.Listen*]

Raises ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code

get_playing_now(*username*)

Get the listen being played right now for user *username*.

Parameters **username** (*str*) – the username of the user whose data is to be fetched

Returns A single listen if the user is playing something currently, else None

Return type *pylistenbrainz.Listen* or None

Raises ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code

get_user_artists(*username*, *count=25*, *offset=0*, *time_range='all_time'*)
Get artists for user 'username', sorted in descending order of listen count.

Parameters

- **username** (*str*) – the username of the user whose artists are to be fetched.
- **count** (*int, optional*) – the number of artists to fetch, defaults to 25, maximum is 100.
- **offset** (*int, optional*) – the number of artists to skip from the beginning, for pagination, defaults to 0.
- **time_range** (*str*) – the time range, can be 'all_time', 'month', 'week' or 'year'

Returns the artists listened to by the user in the time range with listen counts and other data in the same format as the API response

Return type dict

get_user_listen_count(*username*)
Get total number of listens for user

Parameters **username** (*str*) – The username of the user whose listens are to be fetched

Returns Number of listens returned by the Listenbrainz API

Return type int

get_user_recommendation_recordings(*username*, *artist_type='top'*, *count=25*, *offset=0*)
Get recommended recordings for a user.

Parameters

- **username** (*str*) – the username of the user whose recommended tracks are to be fetched.
- **artist_type** (*str*) – The type of filtering applied to the recommended tracks. 'top' for filtering by top artists or 'similar' for filtering by similar artists 'raw' for no filtering
- **count** (*int, optional*) – the number of recordings to fetch, defaults to 25, maximum is 100.
- **offset** (*int, optional*) – the number of releases to skip from the beginning, for pagination, defaults to 0.

Returns the recommended recordings as other data returned by the API

Return type dict

get_user_recordings(*username*, *count=25*, *offset=0*, *time_range='all_time'*)
Get recordings for user 'username', sorted in descending order of listen count.

Parameters

- **username** (*str*) – the username of the user whose artists are to be fetched.
- **count** (*int, optional*) – the number of recordings to fetch, defaults to 25, maximum is 100.
- **offset** (*int, optional*) – the number of recordings to skip from the beginning, for pagination, defaults to 0.
- **time_range** (*str*) – the time range, can be 'all_time', 'month', 'week' or 'year'

Returns the recordings listened to by the user in the time range with listen counts and other data, in the same format as the API response

Return type dict

get_user_releases(*username*, *count=25*, *offset=0*, *time_range='all_time'*)
Get releases for user 'username', sorted in descending order of listen count.

Parameters

- **username** (*str*) – the username of the user whose releases are to be fetched.
- **count** (*int, optional*) – the number of releases to fetch, defaults to 25, maximum is 100.
- **offset** (*int, optional*) – the number of releases to skip from the beginning, for pagination, defaults to 0.
- **time_range** (*str*) – the time range, can be 'all_time', 'month', 'week' or 'year'

Returns the releases listened to by the user in the time range with listen counts and other data

Return type dict

is_token_valid(*token*)
Check if the specified ListenBrainz auth token is valid using the `/1/validate-token` endpoint.

Parameters **token** (*str*) – the auth token that needs to be checked for validity

Raises ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code

set_auth_token(*auth_token*, *check_validity=True*)
Give the client an auth_token to use for future requests. This is required if the client wishes to submit listens. Each user has a unique auth token and the auth token is used to identify the user whose data is being submitted.

Parameters

- **auth_token** (*str*) – auth token
- **check_validity** (*bool, optional*) – specify whether we should check the validity of the auth token by making a request to ListenBrainz before setting it (defaults to True)

Raises

- ***InvalidAuthTokenException*** – if ListenBrainz tells us that the token is invalid
- ***ListenBrainzAPIException*** – if there is an error with the validity check API call

submit_multiple_listens(*listens*)
Submit a list of listens to ListenBrainz.

Requires that the auth token for the user whose listens are being submitted has been set.

Parameters **listens** (*List[pylistenbrainz.Listen]*) – the list of listens to be submitted

Raises

- ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code
- ***InvalidSubmitListensPayloadException*** – if the listens sent are invalid, see exception message for details

submit_playing_now(*listen*)
Submit a playing now notification to ListenBrainz.

Requires that the auth token for the user whose data is being submitted has been set.

Parameters **listen** (pylistenbrainz.Listen) – the listen to be submitted, the listen should NOT have a *listened_at* attribute

Raises

- ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code
- ***InvalidSubmitListensPayloadException*** – if the listen being sent is invalid, see exception message for details

submit_single_listen(*listen*)
Submit a single listen to ListenBrainz.

Requires that the auth token for the user whose data is being submitted has been set.

Parameters **listen** (pylistenbrainz.Listen) – the listen to be submitted

Raises

- ***ListenBrainzAPIException*** – if the ListenBrainz API returns a non 2xx return code
- ***InvalidSubmitListensPayloadException*** – if the listen being sent is invalid, see exception message for details

9.1.2 class Listen

The Listen class represents a Listen from ListenBrainz.

class pylistenbrainz.**Listen**(*track_name*, *artist_name*, *listened_at=None*, *release_name=None*, *recording_mbid=None*, *artist_mbids=None*, *release_mbid=None*, *tags=None*, *release_group_mbid=None*, *work_mbids=None*, *tracknumber=None*, *spotify_id=None*, *listening_from=None*, *isrc=None*, *additional_info=None*, *username=None*)

Bases: object

__init__(*track_name*, *artist_name*, *listened_at=None*, *release_name=None*, *recording_mbid=None*, *artist_mbids=None*, *release_mbid=None*, *tags=None*, *release_group_mbid=None*, *work_mbids=None*, *tracknumber=None*, *spotify_id=None*, *listening_from=None*, *isrc=None*, *additional_info=None*, *username=None*)
Creates a Listen.

Needs at least a track name and an artist name.

Parameters

- **track_name** (*str*) – the name of the track
- **artist_name** (*str*) – the name of the artist
- **listened_at** (*int, optional*) – the unix timestamp at which the user listened to this listen
- **release_name** (*str, optional*) – the name of the MusicBrainz release the track is a part of
- **recording_mbid** (*str, optional*) – the MusicBrainz ID of this listen's recording

- **artist_mbids** (*List[str], optional*) – the MusicBrainz IDs of this listen's artists
- **release_mbid** (*str, optional*) – the MusicBrainz ID of this listen's release
- **tags** (*List[str], optional*) – a list of user defined tags for this recording, each listen can only have at most 50 tags and each tag must be shorter than 64 characters.
- **release_group_mbid** (*str, optional*) – A MusicBrainz Release Group ID of the release group this recording was played from.
- **work_mbids** (*List[str], optional*) – A list of MusicBrainz Work IDs that may be associated with this recording.
- **tracknumber** (*int, optional*) – The tracknumber of the recording. This first recording on a release is tracknumber 1.
- **spotify_id** (*str, optional*) – The Spotify track URL associated with this recording. e.g.: http://open.spotify.com/track/1rrgWMXGCGHru5bIRxGFV0
- **listening_from** (*str, optional*) – the source of the listen, for example: 'spotify' or 'vlc',
- **isrc** (*str, optional*) – The ISRC code associated with the recording.
- **additional_info** (*dict, optional*) – a dict containing any additional fields that should be submitted with the listen.
- **username** (*str, optional*) – the username of the user to whom this listen belongs

Returns A listen object with the passed properties

Return type *Listen*

9.1.3 Statistics (beta)

ListenBrainz has started exposing statistics endpoints. The following classes are related to those endpoints.

9.2 Exceptions

exception pylistenbrainz.errors.**AuthTokenRequiredException**
Bases: *pylistenbrainz.errors.ListenBrainzException*

exception pylistenbrainz.errors.**EmptyPayloadException**
Bases: *pylistenbrainz.errors.InvalidSubmitListensPayloadException*

exception pylistenbrainz.errors.**InvalidAuthTokenException**
Bases: *pylistenbrainz.errors.ListenBrainzException*

exception pylistenbrainz.errors.**InvalidSubmitListensPayloadException**
Bases: *pylistenbrainz.errors.ListenBrainzException*

exception pylistenbrainz.errors.**ListenBrainzAPIException**(*status_code*, *message=None*)
Bases: *pylistenbrainz.errors.ListenBrainzException*

exception pylistenbrainz.errors.**ListenBrainzException**
Bases: Exception

exception pylistenbrainz.errors.**ListenedAtInPlayingNowException**
Bases: *pylistenbrainz.errors.InvalidSubmitListensPayloadException*

exception pylistenbrainz.errors.**TooManyListensException**
Bases: *pylistenbrainz.errors.InvalidSubmitListensPayloadException*

exception pylistenbrainz.errors.**UnknownListenTypeException**
Bases: *pylistenbrainz.errors.InvalidSubmitListensPayloadException*

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

pylistenbrainz.client, 19
pylistenbrainz.errors, 23

INDEX

Symbols

__init__() (*pylistenbrainz.Listen method*), 22

A

AuthTokenRequiredException, 23

E

EmptyPayloadException, 23

G

get_listens() (*pylistenbrainz.client.ListenBrainz method*), 19
get_playing_now() (*pylistenbrainz.client.ListenBrainz method*), 19
get_user_artists() (*pylistenbrainz.client.ListenBrainz method*), 19
get_user_listen_count() (*pylistenbrainz.client.ListenBrainz method*), 20
get_user_recommendation_recordings() (*pylistenbrainz.client.ListenBrainz method*), 20
get_user_recordings() (*pylistenbrainz.client.ListenBrainz method*), 20
get_user_releases() (*pylistenbrainz.client.ListenBrainz method*), 21

I

InvalidAuthTokenException, 23
InvalidSubmitListensPayloadException, 23
is_token_valid() (*pylistenbrainz.client.ListenBrainz method*), 21

L

Listen (*class in pylistenbrainz*), 22
ListenBrainz (*class in pylistenbrainz.client*), 19
ListenBrainzAPIException, 23
ListenBrainzException, 23
ListenedAtInPlayingNowException, 23

M

module
    pylistenbrainz.client, 19
    pylistenbrainz.errors, 23

P

pylistenbrainz.client
    module, 19
pylistenbrainz.errors
    module, 23

S

set_auth_token() (*pylistenbrainz.client.ListenBrainz method*), 21
submit_multiple_listens() (*pylistenbrainz.client.ListenBrainz method*), 21
submit_playing_now() (*pylistenbrainz.client.ListenBrainz method*), 21
submit_single_listen() (*pylistenbrainz.client.ListenBrainz method*), 22

T

TooManyListensException, 24

U

UnknownListenTypeException, 24