Python qBittorrent Documentation

Release 0.1

Vikas Yadav

Dec 24, 2019

Contents

Contents:

CHAPTER 1

Getting started

Python wrapper for qBittorrent Web API (for versions above v3.1.x) For qBittorrent clients with earlier versions, use mookfist's python-qbittorrent.

This wrapper is based on the methods described in qBittorrent's Official Web API Documentation

Some methods are only supported in qBittorent's latest version (v3.3.1 when writing), It'll be best if you upgrade your client to a latest version.

1.1 Installation

The best way is to install a stable release from PyPI:

```
$ pip install python-qbittorrent
```

You can also stay on the bleeding edge of the package:

```
$ git clone https://github.com/v1k45/python-qBittorrent.git
$ cd python-qBittorrent
$ python setup.py install
```

1.2 Quick usage guide

```
from qbittorrent import Client

qb = Client('http://127.0.0.1:8080/')

qb.login('admin', 'your-secret-password')
# defaults to admin:admin.
# to use defaults, just do qb.login()
```

(continues on next page)

(continued from previous page)

```
torrents = qb.torrents()

for torrent in torrents:
    print torrent['name']
```

1.3 Overview of API methods

1.3.1 Getting torrents

- Get all `active` torrents:

```
qb.torrents()
```

- Filter torrents:

```
qb.torrents(filter='downloading', category='my category')
# This will return all torrents which are currently
# downloading and are categoried as ``my category``.

qb.torrents(filter='paused', sort='ratio')
# This will return all paused torrents sorted by their Leech:Seed ratio.
```

Refer qBittorents WEB API documentation for all possible filters.

1.3.2 Downloading torrents

- Download torrents by link:

```
magnet_link = "magnet:?xt=urn:btih:e334ab9ddd91c10938a7....."
qb.download_from_link(magnet_link)

# No matter the link is correct or not,
# method will always return empty JSON object.
```

- Download multipe torrents by list of links:

```
link_list = [link1, link2, link3]
qb.download_from_link(link_list)
```

- Downloading torrents by file:

```
torrent_file = open('my-torrent-file.torrent', 'rb')
qb.download_from_file(torrent_file)
```

- Downloading multiple torrents by using files:

```
torrent_file_list = [open('1.torrent', 'rb'), open('2.torrent', 'rb')]
qb.download_from_file(torrent_file_list)
```

- Specifing save path for downloads:

```
dl_path = '/home/user/Downloads/special-dir/'
qb.download_from_file(myfile, savepath=dl_path)

# same for links.
qb.download_from_link(my_magnet_uri, savepath=dl_path)
```

- Applying labels to downloads:

```
qb.download_from_file(myfile, label='secret-files ;) ')

# same for links.
qb.download_from_link(my_magnet_uri, category='anime')
```

1.3.3 Pause / Resume torrents

- Pausing/ Resuming all torrents:

```
qb.pause_all()
qb.resume_all()
```

- Pausing/ Resuming a speicific torrent:

```
info_hash = 'e334ab9ddd....infohash....5d7fff526cb4'
qb.pause(info_hash)
qb.resume(info_hash)
```

- Pausing/ Resuming multiple torrents:

```
info_hash_list = ['e334ab9ddd9......infohash......fff526cb4',
                  'c9dc36f46d9......infohash......90ebebc46',
                  '4c859243615......infohash......8b1f20108']

qb.pause_multiple(info_hash_list)
qb.resume_multipe(info_hash_list)
```

CHAPTER 2

Performing common tasks

This page is an overview of common actions you'll perform when using this wrapper. Some methods are handled diffently and some are only compatible with the latest versions. It is recommended to update your qBittorrent client to the latest version.

Since we dicussed about download and pause methods on the previous page, we'll be skipping them here.

Please refer Full API method list

2.1 Getting version details

- Get version info of qBittorrent client:

```
In [4]: qb.qbittorrent_version
Out[4]: u'v3.3.1'
```

- Get API Min and Max version:

```
In [6]: qb.api_version
Out[6]: 7

In [7]: qb.api_min_version
Out[7]: 2
```

2.2 Handling added torrents

- Add trackers to a torrent:

```
In [11]: tracker = 'udp://my.prvt.site:1337/announce'

In [12]: infohash = '0e6a7....infohash....5db6'
```

(continues on next page)

(continued from previous page)

```
In [13]: qb.add_trackers(infohash, tracker)
Out[13]: {} # No matter if method fails, it always returns {}.

# to add multiple trackers, add a like break ('%0A') b/w trackers.
```

- Deleting torrents:

```
infohash_list = 'A single infohash or a list() of info hashes'

qb.delete(infohash_list) # for deleting entry from qBittorrent.

qb.delete_permanently(infohash_list) # delete it from disk too.
```

- Global speed limit values:

```
In [14]: qb.global_download_limit
Out[14]: 0

In [15]: qb.global_upload_limit
Out[15]: 51200

In [16]: qb.global_upload_limit = 102400

In [17]: qb.global_upload_limit
Out[17]: 102400
```

- Preferences:

```
qb.preferences() # for getting dictionary of setting vaulues

qb.preferences['setting_name'] # for fetching a particular setting

qb.preferences['setting_name'] = 'setting-value' # for changing a setting value.

# example

In [20]: prefs = qb.preferences()

In [21]: prefs['autorun_enabled']
Out[21]: True

In [22]: prefs['autorun_enabled'] = False

In [23]: prefs['autorun_enabled']
Out[23]: False

# changing multiple settings at once:

qb.set_preferences(setting_name1=setting_value1, setting_name2=setting_value2,
                   setting_nameN=setting_valueN)
```

- Misc:

```
qb.shutdown() # shutdown qbittorrent
```

(continues on next page)

(continued from previous page)

```
qb.toggle_sequential_download() # as it says

qb.logout() # logs out of current session.
```

This page was just for important methods, Please refer Full API method list

CHAPTER 3

API methods

class qbittorrent.client.**Client**(*url*)
class to interact with qBittorrent WEB API

add_trackers(*infohash*, *trackers*)
Add trackers to a torrent.

Parameters

- **infohash** – INFO HASH of torrent.
- **trackers** – Trackers.

:note %0A (aka LF newline) between trackers. Ampersand in tracker urls MUST be escaped.

alternative_speed_status
Get Alternative speed limits. (1/0)

api_version
Get WEB API version.

create_category(*category*)
Create a new category :param category: category to create

decrease_priority(*infohash_list*)
Decrease priority of torrents.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

delete(*infohash_list*)
Delete torrents. Does not remove files.

Parameters **infohash_list** – Single or list() of infohashes.

delete_all()
Delete all torrents. Does not remove files.

delete_all_permanently()
Permanently delete torrents.

delete_permanently (*infohash_list*)
Permanently delete torrents. Removes files.

Parameters **infohash_list** – Single or list() of infohashes.

download_from_file (*file_buffer*, ***kwargs*)
Download torrent using a file.

Parameters

- **file_buffer** – Single file() buffer or list of.
- **save_path** – Path to download the torrent.
- **label** – Label of the torrent(s).

Returns Empty JSON data.

download_from_link (*link*, ***kwargs*)
Download torrent using a link.

Parameters

- **link** – URL Link or list of.
- **savepath** – Path to download the torrent.
- **category** – Label or Category of the torrent(s).

Returns Empty JSON data.

force_start (*infohash_list*, *value*)
Force start selected torrents.

Parameters

- **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.
- **value** – Force start value (bool)

get_alternative_speed_status ()
Get Alternative speed limits. (1/0)

get_default_save_path ()
Get default save path.

get_global_download_limit ()
Get global download speed limit.

get_global_upload_limit ()
Get global upload speed limit.

get_log (***params*)
Returns a list of log entries matching the supplied params.

Parameters

- **normal** – Include normal messages (default: true).
- **info** – Include info messages (default: true).
- **warning** – Include warning messages (default: true).
- **critical** – Include critical messages (default: true).
- **last_known_id** – Exclude messages with "message id" <= last_known_id (default: -1).

Returns list().

For example: qb.get_log(normal='true', info='true')

get_torrent (*infohash*)
Get details of the torrent.

Parameters **infohash** – INFO HASH of the torrent.

get_torrent_download_limit (*infohash_list*)
Get download speed limit of the supplied torrents.

Parameters **infohash_list** – Single or list() of infohashes.

get_torrent_files (*infohash*)
Get list of files for the torrent.

Parameters **infohash** – INFO HASH of the torrent.

get_torrent_piece_hashes (*infohash*)
Get list of all hashes (in order) of a specific torrent.

Parameters **infohash** – INFO HASH of the torrent.

Returns array of hashes (strings).

get_torrent_piece_states (*infohash*)
Get list of all pieces (in order) of a specific torrent.

Parameters **infohash** – INFO HASH of the torrent.

Returns array of states (integers).

get_torrent_trackers (*infohash*)
Get trackers for the torrent.

Parameters **infohash** – INFO HASH of the torrent.

get_torrent_upload_limit (*infohash_list*)
Get upoload speed limit of the supplied torrents.

Parameters **infohash_list** – Single or list() of infohashes.

get_torrent_webseeds (*infohash*)
Get webseeds for the torrent.

Parameters **infohash** – INFO HASH of the torrent.

global_download_limit
Get global download speed limit.

global_transfer_info

Returns dict{} of the global transfer info of qBittorrent.

global_upload_limit
Get global upload speed limit.

increase_priority (*infohash_list*)
Increase priority of torrents.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

login (*username='admin'*, *password='admin'*)
Method to authenticate the qBittorrent Client.

Declares a class attribute named `session` which stores the authenticated session if the login is correct. Else, shows the login error.

Parameters

- **username** – Username.
- **password** – Password.

Returns Response to login request to the API.

logout()
Logout the current session.

pause(*infohash*)
Pause a torrent.

Parameters **infohash** – INFO HASH of torrent.

pause_all()
Pause all torrents.

pause_multiple(*infohash_list*)
Pause multiple torrents.

Parameters **infohash_list** – Single or list() of infohashes.

preferences
Get the current qBittorrent preferences. Can also be used to assign individual preferences. For setting multiple preferences at once, see `set_preferences` method.

Note: Even if this is a `property`, to fetch the current preferences dict, you are required to call it like a bound method.

Wrong:

```
qb.preferences
```

Right:

```
qb.preferences()
```

qbittorrent_version
Get qBittorrent version.

reannounce(*infohash_list*)
Recheck all torrents.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

recheck(*infohash_list*)
Recheck torrents.

Parameters **infohash_list** – Single or list() of infohashes.

recheck_all()
Recheck all torrents.

remove_category(*categories*)

Remove categories

param categories can contain multiple cateogies separated by

(%0A urlencoded).

resume(*infohash*)
Resume a paused torrent.

Parameters **infohash** – INFO HASH of torrent.

resume_all()
Resume all torrents.

resume_multiple(*infohash_list*)
Resume multiple paused torrents.

Parameters **infohash_list** – Single or list() of infohashes.

set_automatic_torrent_management(*infohash_list*, *enable='false'*)
Set the category on multiple torrents.

Parameters

- **infohash_list** – Single or list() of infohashes.
- **enable** – is a boolean, affects the torrents listed in infohash_list, default is 'false'

set_category(*infohash_list*, *category*)
Set the category on multiple torrents.

The category must exist before using set_category. As of v2.1.0,the API returns a 409 Client Error for any valid category name that doesn't already exist.

Parameters

- **infohash_list** – Single or list() of infohashes.
- **category** – If category is set to empty string '',

the torrent(s) specified is/are removed from all categories.

set_file_priority(*infohash*, *file_id*, *priority*)
Set file of a torrent to a supplied priority level.

Parameters

- **infohash** – INFO HASH of torrent.
- **file_id** – ID of the file to set priority.
- **priority** – Priority level of the file.

:note priority 4 is no priority set

set_global_download_limit(*limit*)
Set global download speed limit.

Parameters **limit** – Speed limit in bytes.

set_global_upload_limit(*limit*)
Set global upload speed limit.

Parameters **limit** – Speed limit in bytes.

set_max_priority(*infohash_list*)
Set torrents to maximum priority level.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

set_min_priority(*infohash_list*)
Set torrents to minimum priority level.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

set_preferences(***kwargs*)
Set preferences of qBittorrent. Read all possible preferences @ https://git.io/fx2Y9

Parameters **kwargs** – set preferences in kwargs form.

set_super_seeding(*infohash_list*, *value*)
Set super seeding for selected torrents.

Parameters

- **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.
- **value** – Force start value (bool)

set_torrent_download_limit(*infohash_list*, *limit*)
Set download speed limit of the supplied torrents.

Parameters

- **infohash_list** – Single or list() of infohashes.
- **limit** – Speed limit in bytes.

set_torrent_location(*infohash_list*, *location*)
Set the location for the torrent

Parameters

- **infohash** – INFO HASH of torrent.
- **location** – /mnt/nfs/media.

set_torrent_name(*infohash*, *name*)
Set the name for the torrent

Parameters

- **infohash** – INFO HASH of torrent.
- **name** – Whatever_name_you_want.

set_torrent_upload_limit(*infohash_list*, *limit*)
Set upload speed limit of the supplied torrents.

Parameters

- **infohash_list** – Single or list() of infohashes.
- **limit** – Speed limit in bytes.

shutdown()
Shutdown qBittorrent.

sync_main_data(*rid=0*)
Sync the torrents main data by supplied LAST RESPONSE ID. Read more @ https://git.io/fxgB8

Parameters **rid** – Response ID of last request.

sync_peers_data(*infohash*, *rid=0*)
Sync the torrent peers data by supplied LAST RESPONSE ID. Read more @ https://git.io/fxgBg

Parameters

- **infohash** – INFO HASH of torrent.
- **rid** – Response ID of last request.

toggle_alternative_speed()
Toggle alternative speed limits.

toggle_first_last_piece_priority(*infohash_list*)
Toggle first/last piece priority of supplied torrents.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

toggle_sequential_download(*infohash_list*)
Toggle sequential download in supplied torrents.

Parameters **infohash_list** – Single or list() of infohashes; pass 'all' for all torrents.

torrents(***filters*)
Returns a list of torrents matching the supplied filters.

Parameters

- **filter** – Current status of the torrents.
- **category** – Fetch all torrents with the supplied label.
- **sort** – Sort torrents by.
- **reverse** – Enable reverse sorting.
- **limit** – Limit the number of torrents returned.
- **offset** – Set offset (if less than 0, offset from end).

Returns list() of torrent with matching filter.

For example: qb.torrents(filter='downloading', sort='ratio').

exception qbittorrent.client.**LoginRequired**

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

q

Index