zippyshare-downloader

Release 0.3.0

Rahman Yusuf

Jan 03, 2022

CONTENTS

Download file from Zippyshare directly from Python

CHAPTER

ONE

KEY FEATURES:

In `zippyshare_downloader` you can:

- You can download file from Zippyshare (Yes of course).
- Extract filename, date uploaded, file size, and downloadable url information from given url.
- Fast Download, allows you to download in 2 connections at same time simultaneously.

1.1 Command Line Interface (CLI)

1.1.1 Command Line Interface (CLI) Usage

App names

There is few app names in zippyshare-downloader:

- `zippyshare-dl`
- `zippyshare-downloader`

Note: If none of above doesn't work use this

```
# For Windows
py -3 -m zippyshare_downloader

# For Linux
python3 -m zippyshare_downloader
```

Options

Global options

- `ZIPPYSHARE_URL or FILE` Zippyshare URL or file containing zippyshare urls
- `--no-download` No download file
- `--verbose`, `-v` Enable verbose
- `--replace`, `-r` Replace file if exist
- `--silent` No output

File / Folder related

- `--folder FOLDER` Store downloaded file in given folder
- `--filename FILENAME` Rewrite filename. Will be ignored if using multiple urls

Zip and unzip

- `--zip FILENAME`, `-z FILENAME` Zip all downloaded files
- `--unzip`, `-uz` Unzip all downloaded files, one by one

Warning: Option `--zip` or `-z` will only work if you are using multiple zippyshare urls

For example:

```
# This will work
$ zippyshare-dl "urls.txt" --zip

# This will NOT work
$ zippyshare-dl "https://www.zippyshare.com/v/..." --zip
```

Download related

- `--async` Run zippyshare-downloader in asynchronous mode
- `--fast` Enable fast download

Note: When you using `--async` option, the download is faster than without using `--async` one.

Warning: `--fast` option require `--async`. It will throw error if you specified `--fast` without `--async` option.

For example:

```
# This will work
$ zippyshare-dl "https://www.zippyshare.com/v/..." --async --fast

# This will NOT work
$ zippyshare-dl "https://www.zippyshare.com/v/..." --fast
```

Pipe to media player (etc: vlc)

- `-pipe` Download to stdout, useful for piping media content to media player (like vlc)

Example Usage

```
# Let's say you want watching videos with vlc from zippyshare
# this can be done with piping the stdout from zippyshare-dl
$ zippyshare-dl "insert zippyshare url here" -pipe | vlc -

# or (for Linux / Mac OS)
$ python3 -m zippyshare_downloader "insert zippyshare url here" -pipe | vlc -

# or (for Windows)
$ py -3 -m zippyshar_downloader "insert zippyshare url here" -pipe | vlc -
```

Proxy related

- `--proxy PROXY` Set a http/socks proxy, all network in this app will be re-routed from this proxy.
- `--proxy-trust-env` Enable http/socks proxy from environments (`http_proxy`, `https_proxy`).

Warning: If you specified `--proxy` and `--proxy-trust-env`, option `--proxy-trust-env` will be ignored.

1.2 Embedding (API)

1.2.1 API Documentation

zippyshare_downloader.**extract_info**(*url*, *download=True*, *unzip=False*, ***kwargs*)
Extract all informations in Zippyshare url.

Parameters

- **url** (str) – Zippyshare url.
- **download** (bool) – Download given zippyshare url if `True`, default to `True`.
- **unzip** (bool) – Unzip downloaded file once finished (if given file is zip or tar format extract it, otherwise ignore it), default to `False`.
- ****kwargs** – These parameters will be passed to *File.download()*

Returns Zippyshare file

Return type *File*

zippyshare_downloader.**download**(**urls*, *zip=None*, *unzip=False*, ***kwargs*)
Download multiple zippyshare urls

Parameters

- ***urls** – Zippyshare urls.

- **zip** (str) – Zip all downloaded files once finished. Zip filename will be taken from zip parameter, default to None. NOTE: You can't mix zip and unzip options together with value True, it will raise error.
- **unzip** (bool) – Unzip all downloaded files once finished (if given file is zip format extract it, otherwise ignore it), default to False. NOTE: You can't mix zip and unzip options together with value True, it will raise error.
- ****kwargs** – These parameters will be passed to *File.download()*, except for parameter filename.

Returns a list of Zippyshare files

Return type List[*File*]

async zippyshare_downloader.**extract_info_coro**(*url*, *download=True*, *unzip=False*, ***kwargs*)
"Coroutine Function"

Extract all informations in Zippyshare url.

Parameters

- **url** (str) – Zippyshare url.
- **download** (bool) – Download given zippyshare url if True, default to True.
- **unzip** (bool) – Unzip downloaded file once finished (if given file is zip or tar format extract it, otherwise ignore it), default to False.
- ****kwargs** – These parameters will be passed to *File.download_coro()*

Returns Zippyshare file

Return type *File*

async zippyshare_downloader.**download_coro**(**urls*, *zip=None*, *unzip=False*, ***kwargs*)
"Coroutine Function"

Download multiple zippyshare urls

Parameters

- ***urls** (str) – Zippyshare urls.
- **zip** (str) – Zip all downloaded files once finished. Zip filename will be taken from zip, default to None. NOTE: You can't mix zip and unzip options together with value True, it will raise error.
- **unzip** (bool) – Unzip all downloaded files once finished (if given file is zip format extract it, otherwise ignore it), default to False. NOTE: You can't mix zip and unzip options together with value True, it will raise error.
- ****kwargs** – These parameters will be passed to *File.download_coro()*, except for parameter filename.

Returns a list of Zippyshare files

Return type List[*File*]

zippyshare_downloader.**download_stdout**(*url*)
Extract zippyshare download url and then download its content to stdout

Warning: This will print all its content to stdout, if you are not intend to use this for piping the content to media player (like vlc), then DO NOT DO THIS.

Example usage (Command-line)

```
# Let's say you want watching videos with vlc from zippyshare
# this can be done with piping the stdout from zippyshare-dl
$ zippyshare-dl "insert zippyshare url here" -pipe | vlc -

# or (for Linux / Mac OS)
$ python3 -m zippyshare_downloader "insert zippyshare url here" -pipe | vlc -

# or (for Windows)
$ py -3 -m zippyshar_downloader "insert zippyshare url here" -pipe | vlc -
```

class zippyshare_downloader.**File**(*data*)

property date_uploaded
Return date that this file uploaded.

Type datetime.datetime

property date_uploaded_fmt
Return formatted date that this file uploaded.

Type str

download(*progress_bar=True*, *replace=False*, *folder=None*, *filename=None*)
Download this file

Parameters

- **progress_bar** (bool) – Enable/Disable progress bar, default to *True*
- **replace** (bool) – Replace file if exist, default to *False*
- **folder** (str) – Set a folder where to store downloaded file, default to *None*.
- **filename** (str) – Set a replacement filename, default to *None*.

Returns Zippyshare file downloaded

Return type pathlib.Path

async download_coro(*progress_bar=True*, *replace=False*, *folder=None*, *filename=None*, *fast=False*)
Same like *File.download()* but for asynchronous process

Parameters

- **progress_bar** (bool) – Enable/Disable progress bar, default to *True*
- **replace** (bool) – Replace file if exist, default to *False*
- **folder** (str) – Set a folder where to store downloaded file, default to *None*.
- **filename** (str) – Set a replacement filename, default to *None*.
- **fast** (bool) – Enable Fast download, default to False

Returns Zippyshare file downloaded

Return type pathlib.Path

property download_url
Return downloadable url

Type *download_url*

property name
Return name of the file

Type str

property size
Return size of the file, in bytes.

Type float

property size_fmt
Return formatted size of the file

Type str

to_JSON()
Return all zippyshare informations in JSON

to_dict()
Return all zippyshare informations in dict

property url
Return origin url

Type str

Proxy related

Usage:

```
from zippyshare_downloader.network import Net, set_proxy, clear_proxy

# Http proxy
Net.set_proxy('http://user:pass@address:port')

# socks
Net.set_proxy('socks5://user:pass@address:port')

# Shortcut (if you're lazy af)
set_proxy('http://user:pass@address:port')
set_proxy('socks5://user:pass@address:port')
```

class zippyshare_downloader.**NetworkObject**(*proxy=None*, *trust_env=False*)

property aiohttp
Return proxied aiohttp (if configured)

clear_proxy()
Remove all proxy from aiohttp/requests

close()
Close requests session only

async close_async()
Close aiohttp & requests session

is_proxied()
Return `True` if requests/aiohttp from *NetworkObject* are configured using proxy.

property proxy
Return HTTP/SOCKS proxy, return `None` if not configured

property requests
Return proxied requests (if configured)

set_proxy(*proxy*)
Setup HTTP/SOCKS proxy for aiohttp/requests

property trust_env
Return `True` if http/socks proxy are grabbed from env

zippyshare_downloader.**set_proxy**(*proxy*)
Setup HTTP/SOCKS proxy for aiohttp/requests

This is shortcut for *NetworkObject.set_proxy()*.

zippyshare_downloader.**clear_proxy**()
Remove all proxy from aiohttp/requests

This is shortcut for *NetworkObject.clear_proxy()*.

INDEX