requests$_{d}ownloader Documentation$

Release 0.4.1

Hrishikesh Terdalkar

May 17, 2022

Contents:

Python package to download files

- Free software: GNU General Public License v3
- Documentation: https://requests_downloader.readthedocs.io.

CHAPTER 1

Features

- Hassle-free download using `requests`
- Download from Drive, Dropbox, Archive or direct URLs
- No need to specify a name for the file to be downloaded
- Command Line Interface to download
- External `requests.Session` object can be passed

1.1 requests_downloader

Python package to download files

- Free software: GNU General Public License v3
- Documentation: https://requests_downloader.readthedocs.io.

1.1.1 Features

- Hassle-free download using `requests`
- Download from Drive, Dropbox, Archive or direct URLs
- No need to specify a name for the file to be downloaded

- Command Line Interface to download
- External `requests.Session` object can be passed

1.1.2 Usage

Use in a Project

Get multiple download options (e.g. for `archive.org` links):

```
from requests_downloader import downloader
download_urls, default_idx = downloader.handle_url('<url>')
```

Download a file:

```
from requests_downloader import downloader
downloader.download('<download_url>')
```

Use Console Interface

```
usage: smart-dl [-h] [--download_dir DOWNLOAD_DIR] [--download_file DOWNLOAD_FILE]
                [--download_path DOWNLOAD_PATH] [--block BLOCK] [--timeout TIMEOUT]
                [--resume] [--progress] [--checksum CHECKSUM] [--verbose] [--debug]
                [--version] url

positional arguments:
url                   Download URL

optional arguments:
-h, --help            show this help message and exit
--download_dir DOWNLOAD_DIR
                        Specify downloads directory
--download_file DOWNLOAD_FILE
                        Specify filename
--download_path DOWNLOAD_PATH
                        Specify path (ignores _dir or _file arguments)
--block BLOCK         Block size while writing the file, in bytes
--timeout TIMEOUT     Timeout in seconds
--resume              Try to resume the download, if supported
--progress            Show download progressbar
--checksum CHECKSUM   Checksum to verify integrity of the download
--verbose             Enable verbose output
--debug               Enable debug information
--version             show program's version number and exit
```

1.1.3 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

1.2 Installation

1.2.1 Stable release

To install requests_downloader, run this command in your terminal:

```
$ pip install requests_downloader
```

This is the preferred method to install requests_downloader, 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.

1.2.2 From sources

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

You can either clone the public repository:

```
$ git clone git://github.com/hrishikeshrt/requests_downloader
```

Or download the tarball:

```
$ curl -OJL https://github.com/hrishikeshrt/requests_downloader/tarball/master
```

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

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

1.3 Usage

1.3.1 Use in a Project

Get multiple download options (e.g. for `archive.org` links):

```
from requests_downloader import downloader
download_urls, default_idx = downloader.handle_url('<url>')
```

Download a file:

```
from requests_downloader import downloader
downloader.download('<download_url>')
```

1.3.2 Use Console Interface

```
usage: smart-dl [-h] [--download_dir DOWNLOAD_DIR] [--download_file DOWNLOAD_FILE]
                [--download_path DOWNLOAD_PATH] [--block BLOCK] [--timeout TIMEOUT]
                [--resume] [--progress] [--checksum CHECKSUM] [--verbose] [--debug]
                [--version] url

positional arguments:
url                   Download URL
```

(continues on next page)

(continued from previous page)

```
optional arguments:
-h, --help            show this help message and exit
--download_dir DOWNLOAD_DIR
                      Specify downloads directory
--download_file DOWNLOAD_FILE
                      Specify filename
--download_path DOWNLOAD_PATH
                      Specify path (ignores _dir or _file arguments)
--block BLOCK         Block size while writing the file, in bytes
--timeout TIMEOUT     Timeout in seconds
--resume              Try to resume the download, if supported
--progress            Show download progressbar
--checksum CHECKSUM   Checksum to verify integrity of the download
--verbose             Enable verbose output
--debug               Enable debug information
--version             show program's version number and exit
```

1.4 requests_downloader

1.4.1 requests_downloader package

Submodules

requests_downloader.cli module

Command Line Interface for requests_downloader

requests_downloader.cli.**main**()
CLI for requests_downloader

requests_downloader.downloader module

Main module containing download function

requests_downloader.downloader.**download**(*url*, *download_dir=''*, *download_file=None*, *download_path=None*, *headers={}*, *session=None*, *block_size=1024*, *timeout=60*, *resume=True*, *show_progress=True*, *show_progress_desc=True*, *max_desc_length=35*, *checksum=None*, *smart=True*, *url_handler=None*)

Download a file

Parameters

- **url** (*str*) – URL to download.
- **download_dir** (*str, optional*) – Path of the directory to download the file in. The default is '' (i.e. current directory).
- **download_file** (*str, optional*) – Name for the downloaded file. If None, the function will infer it from URL and Content-Disposition The default is None.

- **download_path** (*str, optional*) – Full path where the downloaded file should be saved. If None, the function will save it in *download_dir/download_file* If provided, *download_dir* and *download_file* arguments are ignored. The default is None.
- **headers** (*dict, optional*) – Headers to be sent. The default is {}. Note:
  - These headers are merged with a default set of headers.
  - In case of a conflict the user-provided values are used.
  - This behaviour is inherited from *requests.Session()*
- **session** (*object, optional*) – A valid *requests.Session* object. This is useful when download url requires authentication. In such a case, authentication can be handled independently in session. The default is None.
- **block_size** (*int, optional*) – Block size, in bytes, to stream the downloadable content. The default is 1024.
- **timeout** (*float, optional*) – Timeout, in seconds The default is 60.
- **resume** (*bool, optional*) – Try to resume download. The default is True.
- **show_progress** (*bool, optional*) – Show progressbar. The default is True.
- **show_progress_desc** (*str or bool, optional*) – Show the description to the left of progressbar. If False or None, no description is shown. If True, the name of file being downloaded is shown. Otherwise, the *str()* of the provided value is shown. The default is True.
- **max_desc_length** (*int, optional*) – If length of the description is more, abbreviate it by showing the first and last parts connected by three dots. The default is 35.
- **checksum** (*str, optional*) – Value of md5 checksum of the file to be downloaded. If provided, the downloaded file will be verified using the checksum. The default is None.
- **smart** (*bool, optional*) – Use url_handler for special case URLs The default is True.
- **url_handler** (*function, optional*) – Handler function for special cases of download URLs The function should return a list of (TAG, URL) pairs and default index

Returns **download_path** – If download was successful, full *download_path* otherwise, None

Return type str or None

Module contents

requests_downloader

Python package to download files

1.5 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:

1.5.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/hrishikeshrt/requests_downloader/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.

Fix Bugs

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

Implement Features

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

Write Documentation

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

Submit Feedback

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

1.5.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/requests_downloader.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 requests_downloader
$ cd requests_downloader/
$ 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 requests_downloader tests
$ python setup.py test or pytest
$ 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.

1.5.3 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 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/hrishikeshrt/requests_downloader/pull_requests and make sure that the tests pass for all supported Python versions.

1.5.4 Tips

To run a subset of tests:

```
$ pytest tests.test_requests_downloader
```

1.5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

1.6 Credits

1.6.1 Development Lead

- Hrishikesh Terdalkar <hrishikeshrt@linuxmail.org>

1.6.2 Contributors

None yet. Why not be the first?

1.7 History

1.7.1 0.4.0 (2022-04-28)

- Option to show filename to the left of progressbar
- Modularize code
- Style: Black
- Fix bugs

1.7.2 0.3.0 (2022-02-11)

- Several documentation updates
- Code refactoring
- Change console command from clunky *requests_downloader* to *smart-dl*

1.7.3 0.2.0 (2021-06-08)

- Improvements in content disposition and content length handling
- Logging improvements

1.7.4 0.1.7 (2021-06-07)

- Minor bugfixes and spelling corrections.

1.7.5 0.1.5 (2020-11-03)

- Support for Google Docs (Document, Spreadsheets, Presentations).

1.7.6 0.1.4 (2020-09-07)

- Bugfixes.
- First Alpha release.

1.7.7 0.1.0 (2020-09-05)

- First release on PyPI.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

r

Index

D

M

R