torlib Documentation

Release

July 14, 2022

Table of Contents

torlib is an open source Python utility package that can help in many scenarios :D

Chapter 1

Installation

The recommended method for installing torlib is by using pip.

1.1 Installing from PyPi

To install ChatterBot from PyPi using pip run the following command in your terminal.

```
pip install torlib
```

1.2 Installing from GitHub

You can also install torlib directly from GitHub using `pip`.

```
pip install git+https://github.com/noppadol-assava/torlib.git@master
```

1.3 Installing from source

1. Clone repository from GitHub. git is needed in this step.

```
git clone https://github.com/noppadol-assava/torlib.git
```

2. Install the package using pip

```
pip install ./torlib
```

1.4 Checking the version of torlib that you have installed

To check the current version of torlib installed on your machine, run the following python code.

```
import torlib
print(torlib.__version__)
```

1.5 Upgrading torlib to the latest version

To upgrade torlib to the latest release, run the following command in your terminal.

```
pip install torlib --upgrade
```

Chapter 2

Crawler

torlib provide the easy way to crawl data from API in just only one function call. The current supported API is GitHub API.

2.1 GitHub Crawler

This provide the function for crawling data from GitHub API. torlib also support multiprocess (by specifying `pc`) to allow faster crawling by utilize all of rate limit of the API request. When the rate limit exceed, the function will wait until the rate limit reset and try sending request to the url again

2.1.1 github_crawler_multipage

`torlib.crawler.github_crawler.`**`github_crawler_multipage`** (*savename, url, GHtoken, retry=3, pc=1, log_file='github_crawler_log.txt', output_dir='', for_test=False, pretty_json=True*)

crawl the github api and save file to json this function will also generate the log file that show the url of api that cannot be crawled

Parameters

- **`savename`** (*`list`*) – contain string of the save file name (need to be same length as url)
- **`url`** (*`list`*) – contain string of url of the target api (need to be same length as savename)
- **`GHtoken`** (*`list`*) – list of github token
- **`retry`** (*`int, optional`*) – number of time to retry crawling the fail case. Defaults to 3.
- **`pc`** (*`int, optional`*) – number of process for multiprocessing. Defaults to 1.
- **`log_file`** (*`str, optional`*) – name of the log file showing the detail of fail case. Defaults to 'github_crawler_log.txt'.
- **`output_dir`** (*`str, optional`*) – output directory. Defaults to ''.
- **`for_test`** (*`boolean, optional`*) – used for testing or not. Defaults to False.
- **`pretty_json`** (*`boolean, optional`*) – make to output json file easier to read. Defaults to True.

Raises

- **`LengthNotMatchError`** – Raised when the length of savename and url is not the same
- **`InputNotStringError`** – Raised when not all of member in savename or url are string
- **`NoTokenError`** – Raised when input list of github token is empty

Example Usage

```
import torlib.crawler.github_crawler as gc

savename = ['test1', 'test2', 'test3']
url = ['github/api/link/1', 'github/api/link/2', 'github/api/link/3']
GHtoken = ['token1','token2']

gc.github_crawler_multipage(savename, url, GHtoken, output_dir='data')
```

The above python code will created *data/test1.json*, *data/test2.json*, *data/test3.json*, and *github_crawler_log.txt*. The result from each of API url will be in separate json file (if successfully crawl) and stored in `output_dir` directory. The log will list the url and error message in `log_file` If there is an error when crawling the url, after trying to crawl it `retry` times, the url will be in `log_file` together with error message.

Listing 2.1. github_crawler_log.txt

```
[('github/api/link/1','Connection Error')]
```

If there are any bug reports or feature requests, feel free to report them on the project's issue tracker on GitHub.

- genindex
- modindex
- search

Index