lazyscraper documentation

Release 0.1.3

Ivan Begtin

Oct 12, 2022

Contents

Lazyscraper is a simple command line tool and library, a swiss knife for scraper writers. It's created to work only from command line and to make easier scraper writing for very simple tasks like extraction of external urls or simple table.

CHAPTER 1

Supported patterns

- simpleul - Extracts list of urls with pattern ul/li/a. Returns array of urls with “_text” and “href” fields
- simpleopt - Extracts list of options values with pattern select/option. Returns array: “_text”, “value”
- exturls - Extracts list of urls that leads to external websites. Returns array of urls with “_text” and “href” fields
- getforms - Extracts all forms from website. Returns complex JSON data with each form on the page

CHAPTER 2

Command-line tool

Usage: lazyscraper.py [OPTIONS] COMMAND [ARGS]...

Options:

--help Show this message and exit.

Commands: * extract Extract data with xpath * gettable Extracts table with data from html * use Uses predefined pattern to extract page data

CHAPTER 3

Examples

Extracts list of photos and names of Russian government ministers and outputs it to “gov_persons.csv”

> python lscraper.py extract –url http://government.ru/en/gov/persons/ –xpath “//img[@class=’photo’]” –fieldnames src,srcset,alt –absolutize True –output gov_persons.csv –format csv

Extracts list of ministries from Russian government website using pattern “simpleul” and from UL tag with class “departments col col__wide” and outputs absolutized urls.

> python lscraper.py use –pattern simpleul –nodeclass “departments col col__wide” –url http://government.ru/en/ministries –absolutize True

Extracts list of territorial organizations urls from Russian tax service website using pattern “simpleopt”.

> python lscraper.py use –pattern simpleopt –url http://nalog.ru

Extracts all forms from Russian tax service website using pattern “getforms”. Returns JSON with each form and each button, input and select

> python lscraper.py use –pattern getforms –url http://nalog.ru

Extracts list of websites urls of Russian Federal Treasury and uses awk to extract domains.

> python lscraper.py extract –url http://roskazna.ru –xpath “//ul[@class=’site-list’]/li/a” –fieldnames href | awk -F/ ‘{print $3}’

CHAPTER 4

How to use library

Extracts all urls with fields: src, alt, href and _text from gov.uk website

```
>>> from lazyscraper import extract_data_xpath
>>> extract_data_xpath('http://gov.uk', xpath='//a', fieldnames='src,alt,href,_
↪text', absolutize=True)
```

Run pattern 'simpleopt' against Russian federal treasury website

```
>>> from lazyscraper import use_pattern
>>> use_pattern('http://roskazna.ru', 'simpleopt')
```

CHAPTER 5

Requirements

- Python3 https://www.python.org
- click https://github.com/pallets/click
- lxml http://lxml.de/

CHAPTER 6

About Lazyscraper

Lazyscraper is a simple command line tool and library, a swiss knife for scraper writers. It's created to work only from command line and to make easier scraper writing for very simple tasks like extraction of external urls or simple table.

CHAPTER 7

Supported patterns

- simpleul - Extracts list of urls with pattern ul/li/a. Returns array of urls with “_text” and “href” fields
- simpleopt - Extracts list of options values with pattern select/option. Returns array: “_text”, “value”
- exturls - Extracts list of urls that leads to external websites. Returns array of urls with “_text” and “href” fields
- getforms - Extracts all forms from website. Returns complex JSON data with each form on the page

CHAPTER 8

Command-line tool

Usage: lazyscraper.py [OPTIONS] COMMAND [ARGS]...

Options:

--help Show this message and exit.

Commands: extract Extract data with xpath gettable Extracts table with data from html use Uses predefined pattern to extract page data

CHAPTER 9

Examples

Extracts list of photos and names of Russian government ministers and outputs it to “gov_persons.csv”

```
>>> python lscraper.py extract --url http://government.ru/en/gov/persons/ --xpath "//
↪img[@class='photo']" --fieldnames src,srcset,alt --absolutize True --output gov_
↪persons.csv --format csv
```

Extracts list of ministries from Russian government website using pattern “simpleul” and from UL tag with class “departments col col__wide” and outputs absolutized urls.

```
>>> python lscraper.py use --pattern simpleul --nodeclass "departments col col__wide"␣
↪--url http://government.ru/en/ministries  --absolutize True
```

Extracts list of territorial organizations urls from Russian tax service website using pattern “simpleopt”.

```
>>> python lscraper.py use --pattern simpleopt --url http://nalog.ru
```

Extracts all forms from Russian tax service website using pattern “getforms”. Returns JSON with each form and each button, input and select

```
>>> python lscraper.py use --pattern getforms --url http://nalog.ru
```

Extracts list of websites urls of Russian Federal Treasury and uses awk to extract domains.

```
>>> python lscraper.py extract --url http://roskazna.ru --xpath "//ul[@class='site-
↪list']/li/a" --fieldnames href | awk -F/ '{print $3}'
```

CHAPTER 10

How to use library

Extracts all urls with fields: src, alt, href and _text from gov.uk website

```
>>> from lazyscraper import extract_data_xpath
>>> extract_data_xpath('http://gov.uk', xpath='//a', fieldnames='src,alt,href,_
↪text', absolutize=True)
```

Run pattern 'simpleopt' against Russian federal treasury website

```
>>> from lazyscraper import use_pattern
>>> use_pattern('http://roskazna.ru', 'simpleopt')
```

CHAPTER 11

Requirements

- Python3 https://www.python.org
- click https://github.com/pallets/click
- lxml http://lxml.de/

CHAPTER 12

Documentation

Contents:

12.1 Installation

At the command line:

```
$ pip install lazyscraper
```

Or, if you don't have pip installed:

```
$ easy_install lazyscraper
```

If you want to install from the latest sources, you can do:

```
$ git clone https://github.com/ivbeg/lazyscraper.git
$ cd lazyscraper
$ python setup.py lazyscraper
```

12.2 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:

12.2.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/ivbeg/lazyscraper/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 thrgough the GitHub issues for bugs. Anything taged with "bug" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it. We encourage you to add new languages to existing stack.

Write Documentation

Lazyscraper could always use more documentation, whether as part of the official lazyscraper 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/ivbeg/lazyscraper/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 contributions are welcome :)

12.2.2 Get Started!

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

1. Fork the *lazyscraper* repo on GitHub.
2. Clone your fork locally:
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 lazyscraper
$ cd lazyscraper/
$ 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:

```
$ pip install -r tests/requirements.txt # install test dependencies
$ flake8 lazyscraper tests
$ nosetests
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv. (Note that we use `max-line-length = 100` for flake8, this is configured in `setup.cfg` file.)

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.

12.2.3 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. Check https://travis-ci.org/ivbeg/lazyscraper/pull_requests and make sure that the tests pass for all supported Python versions.
4. Follow the core developers' advice which aim to ensure code's consistency regardless of variety of approaches used by many contributors.
5. In case you are unable to continue working on a PR, please leave a short comment to notify us. We will be pleased to make any changes required to get it done.

12.3 Credits

12.3.1 Committers

- Ivan Begtin

12.4 History

12.4.1 0.1.1 (2020-04-26)

- Minor fixes

12.4.2 0.1.0 (2018-01-14)

- First public release on PyPI and updated github code

CHAPTER 13

Indices and tables

- genindex
- modindex
- search