Pywordapi

Release 1.0.0

Jul 21, 2019

Contents

CHAPTER 1

Overview

Python Wordpress Api Library

> **Pywordapi** allows a simple way to get data in and out of WordPress over HTTP, using python and Wordpress REST API.

1.1 Installation

To install the Pywordapi package:

```
pip install pywordapi
```

1.2 Usage

Get posts

```
from pywordapi import Pywordapi
api_url = "https://demo.wp-api.org/"
api = Pywordapi.WpRest(api_url)
results = api.get_posts()
```

Get categories

```
from pywordapi import Pywordapi
api_url = "https://demo.wp-api.org/"
api = Pywordapi.WpRest(api_url)
results = api.get_categories()
```

Variable **results** will return instance of type list (of dict).

Using proxy

```
from pywordapi import Pywordapi
api_url = "https://demo.wp-api.org/"
proxy_url = "http://username:password@IP_ADDRESS:PORT"
api = Pywordapi.WpRest(api_url, proxy_url)
results = api.get_posts()
```

1.3 Documentation

https://pywordapi.readthedocs.io/

CHAPTER 2

Reference

2.1 pywordapi

class pywordapi.**WpRest**(*api_url*, *headers=None*, *proxy=None*)
make api call and retrieve postdata from wordpress sites

static is_empty(*any_structure*)
check is data structure empty

CHAPTER 3

Development

To run the all tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

Windows	`set PYTEST_ADDOPTS=--cov-append` `tox`
Other	`PYTEST_ADDOPTS=--cov-append tox`

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

4.1 Bug reports

When 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.

4.2 Documentation improvements

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

4.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/clchangnet/pywordapi/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 code contributions are welcome :)

4.4 Development

To set up *pywordapi* for local development:

1. Fork pywordapi (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/pywordapi.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. 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
```

6. Submit a pull request through the GitHub website.

4.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

4.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 5

Authors

- Allan Chang - https://clchang.net/

CHAPTER 6

Changelog

6.1 1.0.0 (2019-06-21)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index