GEO Python Documentation

Release 0.1.2

Jerry Lau

Oct 31, 2017

Contents

Contents:

CHAPTER 1

GEO Python

Simple GEO library based on Redis GEO commands

- Free software: MIT license
- Documentation: https://geo-python.readthedocs.io.

From version 3.2, Redis contains a set of very wonderful commands: the GEO commands (https://redis.io/commands#geo)

With these commands, we can easily develop LBS or GEO application.

Unfortunately, these features are not in redis-py(https://github.com/andymccurdy/redis-py) released packages, so we can only use its development version.

Live Demo

https://github.com/JerryLeooo/geo_example

Installation

Now to use geo_python, you have to install the development version of redis-py first, and I am trysng to fix this...

```
$ pip install https://github.com/andymccurdy/redis-py/archive/master.zip#egg=redis

$ pip install geo_python
```

Get Started

```
In [1]: from geo_python import Point
In [2]: class MyPoint(Point):
   ...:     __key__ = 'my_point'
   ...:


In [3]: point = MyPoint.create(120, 40, 'my point 1')
In [4]: MyPoint.query_by_pos(point.longitude, point.latitude)
Out[4]: [<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108␣
↪member:my point 1>]


In [5]: MyPoint.query_by_member(point.member)
Out[5]: [<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108␣
↪member:my point 1>]


In [6]: point.update(member='my point 2')
In [7]: print point.member
my point 2

In [8]: another_point = MyPoint.get_by_member(point.member)

In [9]: print another_point
<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108 member:my␣
↪point 2>

In [10]: print MyPoint.dist(point, another_point)
0.0

In [11]: point.geo_hash()
Out[11]: 'wxj7d9v2fs0'
```

Credits

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

CHAPTER 2

Installation

Now to use geo_python, you have to install the development version of redis-py first, and I am trysng to fix this...

```
$ pip install https://github.com/andymccurdy/redis-py/archive/master.zip#egg=redis
```

Stable release

To install GEO Python, run this command in your terminal:

```
$ pip install geo_python
```

This is the preferred method to install GEO Python, 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.

From sources

The sources for GEO Python can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/jerryleooo/geo_python
```

Or download the tarball:

```
$ curl  -OL https://github.com/jerryleooo/geo_python/tarball/master
```

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

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

CHAPTER 3

Usage

To use GEO Python in a project:

```
In [1]: from geo_python import Point
In [2]: class MyPoint(Point):
   ...:     __key__ = 'my_point'
   ...:


In [3]: point = MyPoint.create(120, 40, 'my point 1')
In [4]: MyPoint.query_by_pos(point.longitude, point.latitude)
Out[4]: [<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108␣
↪member:my point 1>]


In [5]: MyPoint.query_by_member(point.member)
Out[5]: [<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108␣
↪member:my point 1>]


In [6]: point.update(member='my point 2')
In [7]: print point.member
my point 2

In [8]: another_point = MyPoint.get_by_member(point.member)

In [9]: print another_point
<MyPoint __key__:my_point longitude:120.000000894 latitude:39.9999999108 member:my␣
↪point 2>

In [10]: print MyPoint.dist(point, another_point)
0.0

In [11]: point.geo_hash()
Out[11]: 'wxj7d9v2fs0'
```

CHAPTER 4

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:

Types of Contributions

Report Bugs

Report bugs at https://github.com/jerryleooo/geo_python/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

GEO Python could always use more documentation, whether as part of the official GEO Python 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/jerryleooo/geo_python/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 :)

Get Started!

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

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

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

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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/jerryleooo/geo_python/pull_requests and make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

```
$ python -m unittest tests.test_geo_python
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search