pystrava Documentation

Release 0.2.2

Oriol Fabregas

Feb 17, 2019

Contents

Contents:

Introduction

Transparent authentication for Strava API. This library has been inspired by spotifylib as it uses a very similar authentication flow.

The idea is to use all features and methods from stravalib but handling the authentication transparently just by providing all required details.

It's currently written for Strava V3 API

Read more on USAGE.rst or Read the docs or check the code for substantial docstrings.

1.1 Features

- Same features as Stravalib's library but with transparent authentication
- Renew's the token transparently

CHAPTER 2

Installation

At the command line:

```
$ pip install pystrava
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv pystrava
$ pip install pystrava
```

Or, if you are using pipenv:

```
$ pipenv install pystrava
```

Usage

To use pystrava in a project:

First off you'll have to create your application on your Strava profile. Give it a name and as website you can just use `http://localhost.local/callback` and in terms of Authorization Callback Domain, `localhost.local` will suffice.

Once created, you should have a `client_id` and a `client_secret`.

You will also need to use a `scope` to get a token that has access to the resources. Read more about scopes here

Scopes can be appended by using a comma. Let's assume we will use `activity:write,profile:read_all` as `scope`.

```
$ pip install pystrava
```

```python
from pystrava import Strava
import logging
import os

logging.basicConfig(level=logging.INFO)
strava = Strava(client_id=os.environ['CLIENT_ID'],
                client_secret=os.environ['SECRET'],
                callback=os.environ['CALLBACK_URL'],
                scope=os.environ['SCOPE'],
                email=os.environ['EMAIL'],
                password=os.environ['PASSWORD'])
athlete = strava.get_athlete()
print(athlete)
```

To read more on what available methods and features `stravalib` has, go to this link.

Contributing

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

4.1 Submit Feedback

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.

4.1.1 Get Started!

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

1. Clone your fork locally:

```
$ git clone https://github.com/wefner/pystrava
```

2. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your clone for local development:

```
$ mkvirtualenv pystrava
$ cd pystrava/
$ python setup.py develop
```

3. Create a branch for local development:

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

Now you can make your changes locally.

4. Commit your changes and push your branch to the server:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

5. Submit a merge request

pystrava

5.1 pystrava package

5.1.1 Submodules

5.1.2 pystrava.constants module

Main code for pystrava

class `pystrava.constants.`**`Token`**(*access_token*, *token_type*, *expires_at*, *expires_in*, *refresh_token*)
    Bases: `tuple`

    `access_token`
        Alias for field number 0

    `expires_at`
        Alias for field number 2

    `expires_in`
        Alias for field number 3

    `refresh_token`
        Alias for field number 4

    `token_type`
        Alias for field number 1

class `pystrava.constants.`**`User`**(*client_id*, *client_secret*, *email*, *password*)
    Bases: `tuple`

    `client_id`
        Alias for field number 0

    `client_secret`
        Alias for field number 1

email
> Alias for field number 2

password
> Alias for field number 3

5.1.3 pystrava.pystrava module

Main code for pystrava

class `pystrava.pystrava.`**`Strava`**
> Bases: `object`

class `pystrava.pystrava.`**`StravaAuthenticator`**(*client_id*, *client_secret*, *callback*, *scope*, *email*, *password*)
> Bases: `object`

> This class handles Strava V3API OAuth authorization transparently.

> More details can be found on https://developers.strava.com/docs/authentication

> **token**
> > Token namedtuple

> > Returns: namedtuple

5.1.4 pystrava.pystravaexceptions module

Custom exception code for pystrava

5.1.5 Module contents

pystrava package

Import all parts from pystrava here

Credits

6.1 Development Lead

- Oriol Fabregas <fabregas.oriol@gmail.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

History

0.2.2 (2019-02-17)

- Fixed bug that gets *redirects_uri*
- Fixed bug about relative imports

CHAPTER 9

0.2.1 (2019-02-12)

- Retrieve new access token if expired transparently

- Logging

- Docs

CHAPTER 10

0.2.0 (2019-02-08)

- V3 API with refresh token

0.1.2 (2018-08-22)

- Parsing code query string with built-in methods

0.1.1 (2018-08-22)

- First release

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index

A

access_token (pystrava.constants.Token attribute),

C

client_id (pystrava.constants.User attribute),
client_secret (pystrava.constants.User attribute),

E

email (pystrava.constants.User attribute),
expires_at (pystrava.constants.Token attribute),
expires_in (pystrava.constants.Token attribute),

P

password (pystrava.constants.User attribute),
pystrava (module),
pystrava.constants (module),
pystrava.pystrava (module),
pystrava.pystravaexceptions (module),

R

refresh_token (pystrava.constants.Token attribute),

S

Strava (class in pystrava.pystrava),
StravaAuthenticator (class in pystrava.pystrava),

T

Token (class in pystrava.constants),
token (pystrava.pystrava.StravaAuthenticator attribute),
token_type (pystrava.constants.Token attribute),

U

User (class in pystrava.constants),