djangomailup Documentation

Release 0.2.0

Simone Basso

Apr 22, 2017

Contents

Django app to integrate with MailUp

CHAPTER 1

Documentation

The full documentation is at https://djangomailup.readthedocs.org.

CHAPTER 2

Requirements

- OAuth2 tokens for the MailUp REST API
- MailUp account
- Django >= 1.8
- python 2.7+, 3.5+

CHAPTER 3

Quickstart

Install djangomailup:

```
pip install djangomailup
```

Add configuration in settings.py:

```
INSTALLED_APPS = [
    'djangomailup',
]

MAILUP = {
    "default": {
        "client_id": "client_id",
        "client_secret": "client_secret",
        "username": "m1234",
        "password": "password",
    },
}
```

Then use it in a project:

```
from djangomailup import MailUpClient

client = MailUpClient()
```

CHAPTER 4

Features

- TODO

CHAPTER 5

Running Tests

Does the code actually work?

```
$ pip install -r requirements_test.txt
$ python runtests.py
```

CHAPTER 6

Credits

Tools used this package:

- Cookiecutter
- cookiecutter-djangopackage
- requests

Djangomailup

Django app to integrate with MailUp

Documentation

The full documentation is at https://djangomailup.readthedocs.org.

Requirements

- OAuth2 tokens for the MailUp REST API
- MailUp account
- Django >= 1.8
- python 2.7+, 3.5+

Quickstart

Install djangomailup:

```
pip install djangomailup
```

Add configuration in settings.py:

```
INSTALLED_APPS = [
    'djangomailup',
]

MAILUP = {
    "default": {
        "client_id": "client_id",
        "client_secret": "client_secret",
        "username": "m1234",
        "password": "password",
    },
}
```

Then use it in a project:

```
from djangomailup import MailUpClient

client = MailUpClient()
```

Features

- TODO

Running Tests

Does the code actually work?

```
$ pip install -r requirements_test.txt
$ python runtests.py
```

Credits

Tools used this package:

- Cookiecutter
- cookiecutter-djangopackage
- requests

Installation

Run this simple command in your terminal:

```
pip install djangomailup
```

Add `'djangomailup'` to your `INSTALLED_APPS` setting:

```
INSTALLED_APPS = [
    ...  # other apps
    'djangomailup',
]
```

Then add `MAILUP` configurations in your settings like:

```
MAILUP = {
    "default": {
        "client_id": "client_id",
        "client_secret": "client_secret",
        "username": "m1234",
        "password": "password",
    },
}
```

where:

1. `default`: is the name of configuration
2. `client_id` and `client_secret` are tokens for MailUp API REST
3. `username` and `password` are credentials for MailUp account

Usage

To use djangomailup in a project:

```
from djangomailup import MailUpClient

client = MailUpClient()

# return account info
info = client.get_info()
```

Session

`AuthenticateSession` is a low level objects to send http request to MailUp, it's perfect to check your configuration or try an unsupported endpoint.

This is probably the easiest way to connect with MailUp.

class `djangomailup.`**`AuthenticateSession`**(*using='default'*)

A requests authenticated `Session`.

Parameters **`using`** (*`string`*) – name of MAILUP configuration (default: 'default')

Same of `Session`, with oAuth2 logic.

Usage:

```
>>> form djangomailup import AuthenticateSession
>>> s = AuthenticateSession()
>>> url = 'https://services.mailup.com/'
>>> s.get(url)
<Response [200]>
```

To use a different configuration use `using` argument:

```
...
>>> s = AuthenticateSession(using='myotherconfiguration')
...
```

Client

`MailUpClient` is an *`AuthenticateSession`* with every built-in request to work with MailUp.

class `djangomailup.`**`MailUpClient`**(*using='default'*)

MailUp client for django.

A requests authenticated `Session`.

Parameters **`using`** (*`string`*) – name of MAILUP configuration (default: 'default')

Same of `Session`, with oAuth2 logic.

Usage:

```
>>> form djangomailup import MailUpClient()
>>> s = MailUpClient()
>>> s.get_info()
<Response [200]>
```

To use a different configuration use `using` argument:

```
...
>>> s = MailUpClient(using='myotherconfiguration')
...
```

`create_lists`(*name*, *default=1*, *scope='newsletters'*, *extra=None*)

Create a new list.

Parameters

- **`name`** (*`str`*) – Name of new list
- **`default`** (*`int`*) – list as a template
- **`scope`** (*`newsletters or Direct_Advertising or Transactional`*) – Type of list
- **`extra`** (*`dict or None`*) – override default params

Take a look at MailUp's documentation if you want know more about Create list

Reference: Create Lists

`get_info`()

Return MailUp Account Info.

Take a look at MailUp's documentation if you want know more about Account Info

Reference: Account Info

`read_lists`()

Return the lists that are visible for authenticated user.

Take a look at MailUp's documentation if you want know more about Read Lists

Reference: Read Lists

update_lists(*list_id*, *extra=None*)
Update an existing list.

Parameters

- **list_id** (*str*) – id of the list
- **extra** (*dict or None*) – override default params

Take a look at MailUp's documentation if you want know more about Update list

Reference: Update Lists

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/simobasso/djangomailup/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" 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.

Write Documentation

djangomailup could always use more documentation, whether as part of the official djangomailup 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/simobasso/djangomailup/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 *djangomailup* for local development.

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

```
$ git clone git@github.com:your_name_here/djangomailup.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 djangomailup
$ cd djangomailup/
$ 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 djangomailup tests
$ python setup.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 2.7, 3.5, and 3.6 and for PyPy. Check https://travis-ci.org/simobasso/djangomailup/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_djangomailup
```

Credits

Development Lead

- Simone Basso <simone.basso@maieuticallabs.it>

Contributors

None yet. Why not be the first?

History

0.2.0 (2016-07-15)

Behavioural Changes

- remove rauth from dependencies

Improvements

- add codacy badge and coverage

Documentation

- add documentation for MailUpClient
- add documentation for AuthenticateSession

Update

- update sphinx from 1.4.4 to 1.4.5

0.1.1 (2016-07-11)

Documentation

- add readme in index.rst
- fix python works versions
- add Usage page

- add installation page
- add rtd_theme
- add requirements section
- add sphinx for building documentation

Fix

- remove setup.py file from bumpersion config

0.1.0 (2016-07-04)

- First release on PyPI.

Index