
mailjet Documentation

Release 1.3.1

Rick van Hattem

Apr 05, 2018

1	Introduction	3
2	Installation	5
3	Usage	7
4	FAQ	9
4.1	How do I give reserved python keywords as parameters?	9
4.2	How do I debug errors?	9
5	Django integration	11
5.1	Introduction	11
5.2	Installation	11
6	mailjet Package	13
6.1	mailjet Package	13
6.2	api Module	13
6.3	conf Module	13
6.4	connection Module	13
6.5	default_settings Module	13
6.6	settings Module	14
6.7	Subpackages	14
6.7.1	Contrib Package	14
	contrib Package	14
6.7.2	django_mailjet Package	14
	django_mailjet Package	14
	forms Module	15
	models Module	15
	tests Module	15
	urls Module	15
	views Module	16
7	Indices and tables	17
	Python Module Index	19

Contents:

READ THIS FIRST!!

This repository isn't compatible with the current Mailjet API (v3) and, as a consequence, is considered deprecated and won't undergo further development. As such, this repository isn't under active maintenance.
--

Introduction

Mailjet is a real-time Cloud Emailing platform and this is a python library to access the [Mailjet Web API](#).

Installation

- Clone this repository:

```
git clone https://github.com/WoLpH/mailjet
```

- cd into the cloned directory and execute:

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

The settings can be configured from a Django settings file through `MAILJET_API_KEY` and `MAILJET_SECRET_KEY`, or through environment variables with the same name.

i.e.

```
export MAILJET_API_KEY='YOUR_API_KEY'  
export MAILJET_SECRET_KEY='YOUR_SECRET_KEY'
```

Alternatively, you can just pass the API key and Secret key as parameters when initializing the mailjet API as follows:

```
import mailjet  
  
mailjet_api = mailjet.Api(api_key='YOUR_API_KEY', secret_key='YOUR_SECRET_KEY')
```

Usage

- To get your account and profile information:

```
import mailjet

mailjet_api = mailjet.Api(api_key='YOUR_API_KEY', secret_key='YOUR_SECRET_KEY')
account_info = mailjet_api.user.infos()
```

account_info would now be assigned the following python dict:

```
{
  'status': 'OK',
  'infos': {
    'username': 'user@domain.com',
    'firstname': 'firstname',
    'locale': 'en_US',
    'lastname': 'lastname',
    'company_name': 'company_name',
    'contact_phone': None,
  }
}
```

- Create a new list of contacts, following on from the previous example:

```
contact_list = mailjet_api.lists.create(
    label='test',
    name='testlist', # Only alphanumeric characters are allowed!
    method='POST'
)
```

contact_list will now contain a dictionary with the status and list id as below:

```
{
  'status': 'OK',
  'contact_id': 000000000
}
```

- You can now add contacts to your list using the contact_id:

```
mailjet_api.lists.addcontact(
    contact='example@example.com',
    id=contact_list['list_id'],
    method='POST'
)
```

How do I give reserved python keywords as parameters?

Methods such as creating a campaign require you to use reserved python keywords, such as `from` - hence, in order to overcome this, do the following:

```
params = dict()
params['method'] = 'POST'
params['subject'] = 'My first campaign'
params['list_id'] = contact_list['list_id']
params['lang'] = 'en'
params['from'] = 'noreply@example.com'
params['from_name'] = 'Your name'
params['footer'] = 'default'
campaign = mailjet_api.message.createcampaign(**params)
```

How do I debug errors?

The errors produced by the `mailjet` library (or actually, produced by the `urllib2` library) are still normal http responses. So if you wish to read the actual response, do something like this:

```
try:
    contact_list = mailjet_api.lists.create(
        label='test',
        name='Test list', # Incorrect because of the space in the name
        method='POST'
    )
except Exception, e:
    print 'Mailjet response: %r, %r' % (e, e.read())
```

Django integration

Introduction

`mailjet.contrib.django_mailjet` provides generic tools to plug basic mailjet support in your project.

Advanced users may use `mailjet.contrib.django_mailjet.forms.SubscriptionForm` directly without installing anything.

Installation

In `settings.py`:

- set `MAILJET_LIST_NAME` and `MAILJET_LIST_LABEL`, note that it may not contain non-alphanumeric characters, those are used as defaults for `mailjet.contrib.django_mailjet.forms.SubscriptionForm`,
- add to `INSTALLED_APPS`: `mailjet.contrib.django_mailjet` for templates to be loadable.

Include `mailjet.contrib.django_mailjet.urls` in `urls.py`, ie.:

```
url(r'^newsletter/', include('mailjet.contrib.django_mailjet.urls')),
```

Users may now subscribe to your mailing list on `/newsletter/`.

mailjet Package

mailjet Package

```
class mailjet.Api (connection=None, api_key=None, secret_key=None)  
    Bases: object
```

api Module

```
class mailjet.api.Api (connection=None, api_key=None, secret_key=None)  
    Bases: object
```

```
class mailjet.api.ApiMethod (api, method)  
    Bases: object
```

```
class mailjet.api.ApiMethodFunction (method, function)  
    Bases: object
```

conf Module

connection Module

```
class mailjet.connection.Connection (api_key=None, secret_key=None, timeout=None)  
    Bases: object
```

```
    classmethod get_connection (api_key, secret_key)
```

```
    get_opener (url)
```

```
    open (method, function, options=None, postdata=None)
```

default_settings Module

These are the default settings, DON'T MODIFY THIS FILE!

If (some) of these settings need changing, do that in *settings.py* instead

The global settings prefix can be changed by modifying the `SETTINGS_PREFIX` variable. These settings can be overwritten by modifying the Django settings (in that case the `SETTINGS_PREFIX` is used) or by modifying `settings.py`

settings Module

The global settings prefix can be changed by modifying the `SETTINGS_PREFIX` variable. These settings can be overwritten by modifying the Django settings (in that case the `SETTINGS_PREFIX` is used) or by modifying this file.

Subpackages

Contrib Package

`contrib` Package

This is where all the additional Mailjet packages reside.

These packages are build to work together with the `mailjet` library but are not needed for basic usage

`django_mailjet` Package

`django_mailjet` Package

Introduction

`mailjet.contrib.django_mailjet` provides generic tools to plug basic mailjet support in your project.

Advanced users may use `mailjet.contrib.django_mailjet.forms.SubscriptionForm` directly without installing anything.

Installation

In `settings.py`:

- set `MAILJET_LIST_NAME` and `MAILJET_LIST_LABEL`, note that it may not contain non-alphanumeric characters, those are used as defaults for `mailjet.contrib.django_mailjet.forms.SubscriptionForm`,
- add to `INSTALLED_APPS`: `mailjet.contrib.django_mailjet` for templates to be loadable.

Include `mailjet.contrib.django_mailjet.urls` in `urls.py`, ie.:

```
url(r'^newsletter/', include('mailjet.contrib.django_mailjet.urls')),
```

Users may now subscribe to your mailing list on `/newsletter/`.

forms Module

```
class mailjet.contrib.django_mailjet.forms.SubscriptionForm (list_name=None,
                                                           list_label=None, *args,
                                                           **kwargs)

Bases: django.forms.forms.Form

Simple subscription form

add_contact ()
    Create a contact with using the email on the list.

api
    Get or create an Api() instance using django settings.

base_fields = OrderedDict([('email', <django.forms.fields.EmailField object>)])

clean_email ()
    Raise ValidationError if the contact exists.

declared_fields = OrderedDict([('email', <django.forms.fields.EmailField object>)])

list_id
    Get or create the list id.

media

save ()
    Call add_contact()
```

models Module

tests Module

```
class mailjet.contrib.django_mailjet.tests.SubscriptionFormTest (methodName='runTest')
Bases: django.test.testcases.TestCase

list_name_and_label ()

setUp ()

tearDown ()

test_add_contact ()

test_clean_email ()

test_save ()

test_settings_override ()
```

urls Module

Defines basic urls.

django_mailjet_subscription_form Url to the basic form view: SubscriptionView.

django_mailjet_subscription_success Url to the basic success template.

views Module

class mailjet.contrib.django_mailjet.views.**SubscriptionView** (**kwargs)
Bases: django.views.generic.edit.ModelForm

Basic subscription views

form_class

alias of SubscriptionForm

form_valid (form)

Call *form.save()* and super itself.

success_url = <django.utils.functional.__proxy__ object>

template_name = 'django_mailjet/subscription_form.html'

Indices and tables

- `genindex`
- `modindex`
- `search`

m

- mailjet, 13
- mailjet.api, 13
- mailjet.conf, 13
- mailjet.connection, 13
- mailjet.contrib, 14
- mailjet.contrib.django_mailjet, 14
- mailjet.contrib.django_mailjet.forms, 15
- mailjet.contrib.django_mailjet.models, 15
- mailjet.contrib.django_mailjet.tests, 15
- mailjet.contrib.django_mailjet.urls, 15
- mailjet.contrib.django_mailjet.views, 16
- mailjet.default_settings, 13
- mailjet.settings, 14

A

add_contact() (mailjet.contrib.django_mailjet.forms.SubscriptionForm method), 15
 Api (class in mailjet), 13
 Api (class in mailjet.api), 13
 api (mailjet.contrib.django_mailjet.forms.SubscriptionForm attribute), 15
 ApiMethod (class in mailjet.api), 13
 ApiMethodFunction (class in mailjet.api), 13

B

base_fields (mailjet.contrib.django_mailjet.forms.SubscriptionForm attribute), 15

C

clean_email() (mailjet.contrib.django_mailjet.forms.SubscriptionForm method), 15
 Connection (class in mailjet.connection), 13

D

declared_fields (mailjet.contrib.django_mailjet.forms.SubscriptionForm attribute), 15

F

form_class (mailjet.contrib.django_mailjet.views.SubscriptionView attribute), 16
 form_valid() (mailjet.contrib.django_mailjet.views.SubscriptionView method), 16

G

get_connection() (mailjet.connection.Connection class method), 13
 get_opener() (mailjet.connection.Connection method), 13

L

list_id (mailjet.contrib.django_mailjet.forms.SubscriptionForm attribute), 15
 list_name_and_label() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15

M

mailjet (module), 13
 mailjet.api (module), 13
 mailjet.conf (module), 13
 mailjet.connection (module), 13
 mailjet.contrib (module), 14
 mailjet.contrib.django_mailjet (module), 14
 mailjet.contrib.django_mailjet.forms (module), 15
 mailjet.contrib.django_mailjet.models (module), 15
 mailjet.contrib.django_mailjet.tests (module), 15
 mailjet.contrib.django_mailjet.urls (module), 15
 mailjet.contrib.django_mailjet.views (module), 16
 mailjet.default_settings (module), 13
 mailjet.settings (module), 14
 media (mailjet.contrib.django_mailjet.forms.SubscriptionForm attribute), 15

O

open() (mailjet.connection.Connection method), 13

S

save() (mailjet.contrib.django_mailjet.forms.SubscriptionForm method), 15
 setUp() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15
 SubscriptionForm (class in mailjet.contrib.django_mailjet.forms), 15
 SubscriptionFormTest (class in mailjet.contrib.django_mailjet.tests), 15
 SubscriptionView (class in mailjet.contrib.django_mailjet.views), 16
 success_url (mailjet.contrib.django_mailjet.views.SubscriptionView attribute), 16

T

tearDown() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15
 template_name (mailjet.contrib.django_mailjet.views.SubscriptionView attribute), 16

test_add_contact() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15

test_clean_email() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15

test_save() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15

test_settings_override() (mailjet.contrib.django_mailjet.tests.SubscriptionFormTest method), 15