
Email Client Documentation

Release 1.0

Hannon, Marcos Vinícius

January 27, 2016

1	push_email package	3
1.1	Subpackages	3
1.2	Submodules	4
1.3	push_email.admin module	4
1.4	push_email.apps module	4
1.5	push_email.calendarhandler module	4
1.6	push_email.forms module	4
1.7	push_email.models module	4
1.8	push_email.tests module	6
1.9	push_email.utils module	6
1.10	push_email.views module	7
1.11	Module contents	7
2	Models	9
3	Indices and tables	11
	Python Module Index	13
	Index	15

Contents:

push_email package

1.1 Subpackages

1.1.1 push_email.migrations package

Submodules

push_email.migrations.0001_initial module

```
class push_email.migrations.0001_initial.Migration (name, app_label)
    Bases: django.db.migrations.migration.Migration

    dependencies = []

    initial = True

    operations = [<CreateModel fields=[('id', <django.db.models.fields.AutoField>), ('copy_to', <django.db.models.fields.F
```

push_email.migrations.0002_auto_20160111_2042 module

```
class push_email.migrations.0002_auto_20160111_2042.Migration (name, app_label)
    Bases: django.db.migrations.migration.Migration

    dependencies = [('push_email', '0001_initial')]

    operations = [<RenameField new_name='email_address', old_name='copy_to', model_name='copy'>, <RenameField
```

push_email.migrations.0003_auto_20160111_2043 module

```
class push_email.migrations.0003_auto_20160111_2043.Migration (name, app_label)
    Bases: django.db.migrations.migration.Migration

    dependencies = [('push_email', '0002_auto_20160111_2042')]

    operations = [<AlterField name='copies', model_name='myemail', field=<django.db.models.fields.related.ManyToMa
```

push_email.migrations.0004_emailid module

```
class push_email.migrations.0004_emailid.Migration (name, app_label)
    Bases: django.db.migrations.migration.Migration
```

```
dependencies = [('push_email', '0003_auto_20160111_2043')]
operations = [<CreateModel fields=[('id', <django.db.models.fields.AutoField>), ('email_id', <django.db.models.fields
```

Module contents

1.2 Submodules

1.3 push_email.admin module

1.4 push_email.apps module

```
class push_email.apps.PushEmailConfig(app_name, app_module)
    Bases: django.apps.config.AppConfig
    name = 'push_email'
```

1.5 push_email.calendarhandler module

1.6 push_email.forms module

```
class push_email.forms.LoginForm(data=None, files=None, auto_id='id_%s', prefix=None, initial=None, error_class=<class 'django.forms.utils.ErrorList'>, label_suffix=None, empty_permitted=False, field_order=None)
    Bases: django.forms.forms.Form
    base_fields = OrderedDict([('email_address1', <django.forms.fields.CharField object at 0x2afe6e6fc518>), ('password', <django.forms.fields.CharField object at 0x2afe6e6fc518>)]
    declared_fields = OrderedDict([('email_address1', <django.forms.fields.CharField object at 0x2afe6e6fc518>), ('password', <django.forms.fields.CharField object at 0x2afe6e6fc518>)]
    media = <django.forms.media.Media object at 0x2afe6e6fc518>
```

1.7 push_email.models module

```
class push_email.models.Copy(*args, **kwargs)
    Bases: django.db.models.base.Model
    Creates a “copy” table
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    exception Copy.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned
    Copy.myemail_set
        Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.
        In the example:
```

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```


`pizza.toppings` and `topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

Copy. **objects** = <django.db.models.manager.Manager object>

```
class push_email.models.EmailId(*args, **kwargs)
```

Bases: `django.db.models.base.Model`

A unique email ID

exception DoesNotExist

Bases: `django.core.exceptions.ObjectDoesNotExist`

exception EmailId.MultipleObjectsReturned

Bases: `django.core.exceptions.MultipleObjectsReturned`

`EmailId.objects` = <django.db.models.manager.Manager object>

```
class push_email.models.MyEmail(*args, **kwargs)
```

Bases: `django.db.models.base.Model`

Email object to save all useful email information

exception DoesNotExist

Bases: `django.core.exceptions.ObjectDoesNotExist`

exception MyEmail.MultipleObjectsReturned

Bases: `django.core.exceptions.MultipleObjectsReturned`

`MyEmail.copies`

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`pizza.toppings` and `topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

`MyEmail.get_next_by_date(*moreargs, **morekwargs)`

`MyEmail.get_previous_by_date(*moreargs, **morekwargs)`

MyEmail.objects = <django.db.models.manager.Manager object>

MyEmail.recipients

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`pizza.toppings` and `topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

```
class push_email.models.Recipient(*args, **kwargs)
```

Bases: `django.db.models.base.Model`

Creates a “recipient” table

exception DoesNotExist

Bases: `django.core.exceptions.ObjectDoesNotExist`

exception Recipient.MultipleObjectsReturned

Bases: `django.core.exceptions.MultipleObjectsReturned`

`Recipient.myemail_set`

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza(Model):
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

`pizza.toppings` and `topping.pizzas` are `ManyToManyDescriptor` instances.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

`Recipient.objects = <django.db.models.manager.Manager object>`

1.8 push_email.tests module

class `push_email.tests.EmailTestCase` (*methodName='runTest'*)

Bases: `django.test.testcases.TestCase`

test_email_creation()

class `push_email.tests.ThreadTestCase` (*methodName='runTest'*)

Bases: `django.test.testcases.TestCase`

test_thread()

1.9 push_email.utils module

class `push_email.utils.EmailThread` (*username, password, imap_id*)

Bases: `threading.Thread`

check_for_new_emails()

Checks for new emails that match the pattern “[BBC423][xx.x.xxxx] Agenda dd/mm/aaaa hh:mm”

run()

Runs this thread

`push_email.utils.add_to_calendar` (*message*)

Formats an even to add it to google calendar :param message: an email message :return:

`push_email.utils.load_already_seen()`

Loads messages that have already been seen in previous runs of this code so that they don’t appear as duplicate in the view for the user :return:

`push_email.utils.login` (*username, password, imap_id*)

Tries to log in and return the imbox object :param username: :param password: :param imap_id: :return: imbox

`push_email.utils.save_email` (*message*)

Saves a given email in the database :param message: email message :return:

1.10 push_email.views module

```
class push_email.views.EmailLoginView(**kwargs)
    Bases: django.views.generic.edit.FormView

    Renders the login view

    form_class
        alias of LoginForm

    success_url = '/home/'

    template_name = 'index.html'

class push_email.views.EmailView(**kwargs)
    Bases: django.views.generic.base.View

    Renders the email view page

    post (request)
        Gets a POST request with the login data and sends it to the backend :param request: POST request from
        the Login view :return: None

    template_name = 'emails.html'

class push_email.views.FullEmail(email_subject=None, email_from=None, email_body=None,
                                email_date=None, recipients=None, copies=None)
    Bases: object

    Creates a new Email object with all fields that will be used in the html templates
```

1.11 Module contents

Models

Indices and tables

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

p

- `push_email`, 7
- `push_email.admin`, 4
- `push_email.apps`, 4
- `push_email.forms`, 4
- `push_email.migrations`, 4
- `push_email.migrations.0001_initial`, 3
- `push_email.migrations.0002_auto_20160111_2042`, 3
- `push_email.migrations.0003_auto_20160111_2043`, 3
- `push_email.migrations.0004_emailid`, 3
- `push_email.models`, 4
- `push_email.tests`, 6
- `push_email.utils`, 6
- `push_email.views`, 7

A

add_to_calendar() (in module push_email.utils), 6

B

base_fields (push_email.forms.LoginForm attribute), 4

C

check_for_new_emails() (push_email.utils.EmailThread method), 6

copies (push_email.models.MyEmail attribute), 5

Copy (class in push_email.models), 4

Copy.DoesNotExist, 4

Copy.MultipleObjectsReturned, 4

D

declared_fields (push_email.forms.LoginForm attribute), 4

dependencies (push_email.migrations.0001_initial.Migration attribute), 3

dependencies (push_email.migrations.0002_auto_20160111_1942.Migration attribute), 3

dependencies (push_email.migrations.0003_auto_20160111_1943.Migration attribute), 3

dependencies (push_email.migrations.0004_emailid.Migration attribute), 3

E

EmailId (class in push_email.models), 5

EmailId.DoesNotExist, 5

EmailId.MultipleObjectsReturned, 5

EmailLoginView (class in push_email.views), 7

EmailTestCase (class in push_email.tests), 6

EmailThread (class in push_email.utils), 6

EmailView (class in push_email.views), 7

F

form_class (push_email.views.EmailLoginView attribute), 7

FullEmail (class in push_email.views), 7

G

get_next_by_date() (push_email.models.MyEmail method), 5

get_previous_by_date() (push_email.models.MyEmail method), 5

I

initial (push_email.migrations.0001_initial.Migration attribute), 3

L

load_already_seen() (in module push_email.utils), 6

login() (in module push_email.utils), 6

LoginForm (class in push_email.forms), 4

M

media (push_email.forms.LoginForm attribute), 4

Migration (class in push_email.migrations.0001_initial), 3

Migration (class in push_email.migrations.0002_auto_20160111_2042), 3

Migration (class in push_email.migrations.0003_auto_20160111_2043), 3

Migration (class in push_email.migrations.0004_emailid), 3

MyEmail (class in push_email.models), 5

MyEmail.DoesNotExist, 5

MyEmail.MultipleObjectsReturned, 5

myemail_set (push_email.models.Copy attribute), 4

myemail_set (push_email.models.Recipient attribute), 6

N

name (push_email.apps.PushEmailConfig attribute), 4

O

objects (push_email.models.Copy attribute), 5

objects (push_email.models.EmailId attribute), 5

objects (push_email.models.MyEmail attribute), 5

objects (push_email.models.Recipient attribute), 6

operations (push_email.migrations.0001_initial.Migration attribute), 3
operations (push_email.migrations.0002_auto_20160111_2042.Migration attribute), 3
operations (push_email.migrations.0003_auto_20160111_2043.Migration attribute), 3
operations (push_email.migrations.0004_emailid.Migration attribute), 4

P

post() (push_email.views.EmailView method), 7
push_email (module), 7
push_email.admin (module), 4
push_email.apps (module), 4
push_email.forms (module), 4
push_email.migrations (module), 4
push_email.migrations.0001_initial (module), 3
push_email.migrations.0002_auto_20160111_2042 (module), 3
push_email.migrations.0003_auto_20160111_2043 (module), 3
push_email.migrations.0004_emailid (module), 3
push_email.models (module), 4
push_email.tests (module), 6
push_email.utils (module), 6
push_email.views (module), 7
PushEmailConfig (class in push_email.apps), 4

R

Recipient (class in push_email.models), 5
Recipient.DoesNotExist, 6
Recipient.MultipleObjectsReturned, 6
recipients (push_email.models.MyEmail attribute), 5
run() (push_email.utils.EmailThread method), 6

S

save_email() (in module push_email.utils), 6
success_url (push_email.views.EmailLoginView attribute), 7

T

template_name (push_email.views.EmailLoginView attribute), 7
template_name (push_email.views.EmailView attribute), 7
test_email_creation() (push_email.tests.EmailTestCase method), 6
test_thread() (push_email.tests.ThreadTestCase method), 6
ThreadTestCase (class in push_email.tests), 6