
Django Webix Sender Documentation

Release 1.0.0

MPA Solutions

May 28, 2020

Contents

1	Contents	1
1.1	Django Webix Sender	1
1.2	Quick Start	2
1.3	Credits	6
1.4	Change Log	6
2	Indices and tables	9

1.1 Django Webix Sender

1.1.1 Documentation

The full documentation is at <https://django-webix-sender.readthedocs.io>.

1.1.2 Quickstart

Install Django Webix Sender:

```
$ pip install django-webix-sender
```

Add `django-webix-sender` to your `INSTALLED_APPS`

```
INSTALLED_APPS = [  
    # ...  
    'django_webix_sender',  
    # ...  
]
```

Add `django-webix-sender` `URLconf` to your project `urls.py` file

```
from django.conf.urls import url, include

urlpatterns = [
    # ...
    url(r'^django-webix-sender/', include('django_webix_sender.urls')),
    # ...
]
```

1.1.3 Running Tests

Does the code actually work?

```
$ source <YOURVIRTUALENV>/bin/activate
$ (myenv) $ pip install tox
$ (myenv) $ tox
```

1.2 Quick Start

1.2.1 Install

`django-webix-sender` is available on <https://pypi.python.org/pypi/django-webix/> install it simply with:

```
$ pip install django-webix-sender
```

1.2.2 Configure

Settings

Add `django_webix_sender` to your `INSTALLED_APPS`

```
INSTALLED_APPS = [
    # ...
    'django_webix_sender',
    # ...
]
```

Add `django-webix-sender` `URLconf` to your project `urls.py` file

```
from django.conf.urls import url, include

urlpatterns = [
    # ...
    url(r'^django-webix-sender/', include('django_webix_sender.urls')),
    # ...
]
```

Warning: This package requires a project with `django-webix` set up.

1.2.3 Usage

Settings

Create the models (e.g. `<app_name>/models.py`)

```
from django.utils.translation import ugettext_lazy as _

WEBIX_SENDER = {
    'send_methods': [
        {
            'method': 'sms',
            'verbose_name': _('Send sms'),
            'function': 'django_webix_sender.utils.send_sms'
        },
        {
            'method': 'email',
            'verbose_name': _('Send email'),
            'function': 'django_webix_sender.utils.send_email'
        }
    ],
    'attachments': {
        'model': 'django_webix_sender.MessageAttachment',
        'upload_folder': 'sender/',
        'save_function': 'django_webix_sender.models.save_attachments'
    },
    'typology_model': {
        'enabled': True,
        'required': False
    },
    'recipients': [
        {
            'model': 'django_webix_sender.Customer',
            'datatable_fields': ['user', 'name', 'sms', 'email']
        },
        {
            'model': 'django_webix_sender.ExternalSubject',
            'datatable_fields': ['user', 'name', 'sms', 'email']
        }
    ],
    'invoices_period': 'bimestrial'
}
```

WEBIX_SENDER['send_methods']

Defines the allowed send methods.

There are two allowed methods type:

- sms
- email

The methods already implemented in this package are:

- `django_webix_sender.send_methods.email.send_email`

The default Django email sender.

```
{
    'method': 'email',
    'verbose_name': _('Send email'),
    'function': 'django_webix_sender.send_methods.email.send_email'
}
```

- `django_webix_sender.send_methods.skebby.send_sms`

Skebby sms APIs.

```
{
    'method': 'sms',
    'verbose_name': _('Send sms with Skebby'),
    'function': 'django_webix_sender.send_methods.skebby.send_sms'
}
```

- `django_webix_sender.send_methods.skebby.send_sms_old`

Old Skebby sms APIs.

```
{
    'method': 'sms',
    'verbose_name': _('Send sms with Skebby'),
    'function': 'django_webix_sender.send_methods.skebby.send_sms_old'
},
```

WEBIX_SENDER['attachments']

Defines the attachments model and the method to store files.

```
{
    'model': 'django_webix_sender.MessageAttachment',
    'upload_folder': 'sender/',
    'save_function': 'django_webix_sender.models.save_attachments'
}
```

WEBIX_SENDER['typology_model']

Defines if the message typology are enabled.

```
{
    'enabled': True,
    'required': False
}
```

WEBIX_SENDER['recipients']

Defines the models to show as a list of recipients.

```
{
    'model': 'django_webix_sender.Customer',
    'datatable_fields': ['user', 'name', 'sms', 'email']
}
```

Base Template

Create a base html template (e.g. `<app_name>/templates/base.html`)


```
{% load i18n %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    {% include "django_webix/static_meta.html" %}
</head>
<body>
</body>

<script type="text/javascript" charset="utf-8">
    webix.ready(function () {
        webix.ui({
            id: 'content_right',
            rows: []
        });

        webix.extend($$('content_right'), webix.OverlayBox);

        load_js('{% url 'django_webix_sender.list' %}');
    });
</script>
</html>
```

1.2.4 Customization

Recipient class

Create a subclass of DjangoWebixSender and define `get_sms`, `get_email`, `get_sms_related` and `get_email_related` properties.

```
class Recipients(DjangoWebixSender):
    name = models.CharField(max_length=255, verbose_name=_('Name'))
    sms = models.CharField(max_length=32, blank=True, null=True, verbose_name=_('Sms
    ↪'))
    email = models.EmailField(max_length=255, blank=True, null=True, verbose_name=_(
    ↪'Email'))
    parent = models.ForeignKey('self', blank=True, null=True, verbose_name=_('Parent
    ↪'))

    @property
    def get_sms(self):
        return self.sms

    @property
    def get_email(self):
        return self.email

    @property
    def get_sms_related(self):
        return self.parent_set.all()
```

(continues on next page)

(continued from previous page)

```
@property
def get_email_related(self):
    return self.parent_set.all()
```

Send method

```
def send_sms(recipients, body, message_sent):

    # ...
    # API gateway sms send
    # ...

    for recipient, recipient_address in recipients['valids']:
        message_recipient = MessageRecipient(
            message_sent=message_sent,
            recipient=recipient,
            sent_number=1,
            status='success',
            recipient_address=recipient_address
        )
        message_recipient.save()
    for recipient in recipients['invalids']:
        pass
    for recipient, recipient_address in recipients['duplicates']:
        pass
    return message_sent
```

1.3 Credits

This package was developed by MPA Solutions

1.3.1 Development Lead

- Alessio Bazzanella <bazzanella@mpasol.it>

1.3.2 Contributors

None yet. Why not be the first?

1.4 Change Log

All notable changes to this project will be documented in this file.

The format is based on [KeepAChangelog](#) and this project adheres to [SemanticVersioning](#).

1.4.1 [1.0.0] - 2020-05-28

Added

- Added maintainability badge
- Added translations
- Added email attachment link

Changed

- *django-webix* min version v1.2.0
- Better invoice area
- Optimized count with exists

Removed

- Removed old skebby gateway

Fixed

- Fixed gateway utils without set sender as django app
- Fixed filters
- Fixed python3 compatibility
- Fixed multiselect split
- Fixed sender window typology autocomplete

1.4.2 [0.3.6] - 2019-04-19

- Added and/or in list filters

1.4.3 [0.3.5] - 2019-04-05

- Fixed parametric send method

1.4.4 [0.3.4] - 2019-03-29

- Fixed kwargs in *send_mixin*
- Added parametric sms functionality

1.4.5 [0.3.3] - 2019-03-28

- Split send function

1.4.6 [0.3.2] - 2019-03-28

- Fix migration 0003

1.4.7 [0.3.1] - 2019-02-27

- Fix Django 2.0 templatetags
- Added support to model *select_related*, *prefetch_related* and *filter* of sender querysets

1.4.8 [0.1.0] - 2018-08-XX

- First release on PyPI.

CHAPTER 2

Indices and tables

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