

---

# **django-atom Documentation**

***Release 0.15.2***

**Adam Dobrawy**

**Aug 04, 2017**



---

## Contents

---

<b>1</b>	<b>django-atom</b>	<b>3</b>
1.1	Documentation . . . . .	3
1.2	Quickstart . . . . .	3
1.3	Extensions . . . . .	3
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Usage</b>	<b>7</b>
<b>4</b>	<b>Django-guardian</b>	<b>9</b>
4.1	Forms . . . . .	9
4.2	Tests . . . . .	9
<b>5</b>	<b>Django-crispy-forms</b>	<b>13</b>
5.1	Filters . . . . .	13
5.2	Forms . . . . .	13
5.3	Views . . . . .	14
<b>6</b>	<b>Contributing</b>	<b>15</b>
6.1	Types of Contributions . . . . .	15
6.2	Get Started! . . . . .	16
6.3	Pull Request Guidelines . . . . .	17
6.4	Tips . . . . .	17
<b>7</b>	<b>Credits</b>	<b>19</b>
7.1	Development Lead . . . . .	19
7.2	Contributors . . . . .	19
<b>8</b>	<b>History</b>	<b>21</b>
8.1	0.1.0 (2015-08-01) . . . . .	21
8.2	0.2.0 (2015-08-04) . . . . .	21
8.3	0.3.0 (2015-08-04) . . . . .	21
8.4	0.6.0 (2015-10-25) . . . . .	21
8.5	0.10.0 (2015-11-21) . . . . .	21
8.6	0.11.0 (2015-12-10) . . . . .	22
8.7	0.12.0 (2016-12-4) . . . . .	22
8.8	0.12.8 (2016-12-16) . . . . .	22

8.9	0.14.0 (2017-06-1)	22
8.10	0.14.1 (2017-06-1)	22
8.11	0.14.2 (2017-06-1)	22
8.12	0.14.3 (2017-06-1)	22
8.13	0.15.0 (2017-07-21)	22
8.14	0.16.0 (2017-08-4)	22
<b>9</b>	<b>Docstrings</b>	<b>23</b>
9.1	Filters	23
9.2	Forms	23
9.3	Models	23
9.4	Views	24
9.5	Mixins	25
	<b>Python Module Index</b>	<b>27</b>

Contents:



A different stuff for Django to faster make a world a better place.

## Documentation

The full documentation is at <https://django-atom.readthedocs.org>.

## Quickstart

Install django-atom:

```
pip install django-atom
```

Then use it in a project:

```
import atom
```

## Extensions

### slugify

Example usage in `settings.py` add

```
AUTOSLUG_SLUGIFY_FUNCTION = 'atom.ext.slugify.slugifier.ascii_slugify'
```

Required ``unicode-slugify``, ``django-autoslug``.





## CHAPTER 2

---

### Installation

---

At the command line:

```
$ easy_install django-atom
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv django-atom  
$ pip install django-atom
```



## CHAPTER 3

---

### Usage

---

To use django-atom in a project:

```
import atom
```



There is some extensions for [django-guardian](#) package.

## Forms

```
class atom.ext.guardian.forms.PermissionsTranslationMixin(*args, **kwargs)
class atom.ext.guardian.forms.TranslatedUserObjectPermissionsForm(*args,
                                                                    **kwargs)

    base_fields = OrderedDict()
    declared_fields = OrderedDict()
    media
```

## Tests

### Example usage

```
from django.core.urlresolvers import reverse
from django.test import TestCase

from atom.ext.guardian.tests import PermissionStatusMixin
from feder.monitorings.factories import MonitoringFactory
from feder.users.factories import UserFactory

class ObjectMixin(object):
    def setUp(self):
        self.user = UserFactory(username='john')
        self.monitoring = self.permission_object = MonitoringFactory()
        self.case = CaseFactory(monitoring=self.monitoring)
```

```
        self.from_user = OutgoingLetterFactory(title='Wniosek',
                                                case__monitoring=self.monitoring)
        self.from_institution = IncomingLetterFactory(title='Odpowiedz',
                                                       case__monitoring=self.
↪monitoring)

class LetterRssFeedTestCase(ObjectMixin, PermissionStatusMixin, TestCase):
    status_anonymous = 200
    status_no_permission = 200
    permission = []

    def get_url(self):
        return reverse('letters:rss')
```

## Docstrings

**class** `atom.ext.guardian.tests.PermissionStatusMixin`

Mixin to verify object permission status codes for different users

Require user with username='john' and password='pass'

**Attributes:** `permission` (list): Description `status_anonymous` (int): Status code for anonymous user `status_has_permission` (int): Status code for user with permission `status_no_permission` (403): Status code for user without permission `url` (string): url to test

**get\_permission()**

Returns the permission to assign for granted permission user

**Returns:** list: A list of permission in format ``codename.permission_name``

**Raises:** `ImproperlyConfigured`: Missing a permission to assign

**get\_permission\_object()**

Returns object of permission-carrying object for grant permission

**get\_url()**

Get url to tests

**Returns:** str: url to test

**Raises:** `ImproperlyConfigured`: Missing a url to test

**grant\_permission()**

Grant permission to user in `self.user`

**Returns:** TYPE: Description

**login\_permitted\_user()**

Login client to user with granted permissions

**permission = None**

**setUp()**

**status\_anonymous = 302**

**status\_has\_permission = 200**

**status\_no\_permission = 403**

**test\_status\_code\_for\_anonymous\_user()**

A test status code of response for anonymous user

**test\_status\_code\_for\_privileged\_user()**

A test for status code of response for privileged user

Grant permission to permission-carrying object and login before test

**test\_status\_code\_for\_signed\_user()**

A test for status code of response for signed (logged-in) user

Only login before test.

**url = None**





There is some extensions for [django-crispy-forms](#) package.

## Filters

```
class atom.ext.crispy_forms.filters.CrispyFilterMixin
```

```
    form
    form_class = 'form-inline'
```

## Forms

```
class atom.ext.crispy_forms.forms.BaseTableFormSet (*args, **kwargs)
class atom.ext.crispy_forms.forms.FormHorizontalMixin (*args, **kwargs)
class atom.ext.crispy_forms.forms.FormsetHelper (form=None)

    form_method = 'post'
    form_tag = False
class atom.ext.crispy_forms.forms.HelperMixin (*args, **kwargs)

    form_helper_cls
        alias of FormHelper
class atom.ext.crispy_forms.forms.InlineTableFormSet (*args, **kwargs)
```

**class** `atom.ext.crispy_forms.forms.SingleButtonMixin` (*\*args, \*\*kwargs*)  
 Dynamically add crispy button to form layout

Usage of mixins is obvious:

```
from django import forms
from atom.ext.crispy_forms.forms import SingleButtonMixin

class PersonModelForm(SingleButtonMixin, forms.ModelForm):

    class Meta:
        model = Person
```

HelperMixin

**action\_text**

A text of added action submit button.

In standards it detects when forms save or update objects

string – A text used in button

**class** `atom.ext.crispy_forms.forms.TableFormSet` (*\*args, \*\*kwargs*)

**class** `atom.ext.crispy_forms.forms.TableFormSetHelper` (*form=None*)

**template** = 'bootstrap/table\_inline\_formset.html'

**class** `atom.ext.crispy_forms.forms.TableFormSetMixin` (*\*args, \*\*kwargs*)

## Views

**class** `atom.ext.crispy_forms.views.FormSetMixin`

**form\_valid** (*form*)

**formset** = None

**formset\_cls**

alias of BaseTableFormSet

**formset\_invalid** (*form, formset*)

**formset\_valid** (*form, formset*)

**get\_context\_data** (*\*\*kwargs*)

**get\_form** (*\*args, \*\*kwargs*)

**get\_formset** ()

**get\_formset\_kwargs** ()

**get\_formset\_valid\_message** ()

**get\_instance** ()

**get\_success\_url** ()

**inline\_form\_cls** = None

**inline\_model** = None

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/ad-m/django-atom/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

django-atom could always use more documentation, whether as part of the official django-atom 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/ad-m/django-atom/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 *django-atom* for local development.

1. Fork the *django-atom* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-atom.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 django-atom
$ cd django-atom/
$ 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 atom 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 2.6, 2.7, and 3.3, and for PyPy. Check [https://travis-ci.org/ad-m/django-atom/pull\\_requests](https://travis-ci.org/ad-m/django-atom/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_atom
```



## CHAPTER 7

---

### Credits

---

### Development Lead

- Adam Dobrawy <naczelnik@jawnosc.tk>

### Contributors

None yet. Why not be the first?





#### **0.1.0 (2015-08-01)**

- First release on PyPI.

#### **0.2.0 (2015-08-04)**

- Added AutocompleteChoiceFilter
- Add CSS-class settings for CrispyFormFilter
- Drop strict requirements for django-tinycontent

#### **0.3.0 (2015-08-04)**

- Split class to independent package to resolve dependencies issues

#### **0.6.0 (2015-10-25)**

- A lot

#### **0.10.0 (2015-11-21)**

- Add asci\_slugifier
- Add absolute\_import to crispy\_forms

## 0.11.0 (2015-12-10)

- Fix support of dj19 by autocomplete\_light API changes

## 0.12.0 (2016-12-4)

- Add `atom.ext.guardian.tests.PermissionStatusMixin`

## 0.12.8 (2016-12-16)

- Mark code as Python 3 compatible

## 0.14.0 (2017-06-1)

- Add `atom.mixins.AdminTestCaseMixin`

## 0.14.1 (2017-06-1)

- Add base class validation in `atom.mixins.AdminTestCaseMixin`

## 0.14.2 (2017-06-1)

- Fix username and message in `atom.mixins.AdminTestCaseMixin`

## 0.14.3 (2017-06-1)

- Fix django 1.8-1.9 compatibility of `atom.mixins`

## 0.15.0 (2017-07-21)

- Add management command `generate_factory`, `generate_routers`, `generate_serializers`, `generate_viewsets`, `generate_viewsets_tests`
- Add serializer `atom.ext.contenttypes.ContentTypeSerializer` , `atom.ext.sites.serializer.SiteSerializer`
- Add mixin `atom.ext.rest_framework.ViewSetTestCaseMixin`

## 0.16.0 (2017-08-4)

- Fix management commands import

## Filters

## Forms

```
class atom.forms.AuthorMixin
```

```
    save (*args, **kwargs)
```

```
class atom.forms.PartialMixin
```

```
    classmethod partial (*args, **kwargs)
```

## Models

```
class atom.models.AttachmentBase (*args, **kwargs)
```

```
    class Meta
```

```
        abstract = False
```

```
        verbose_name = u'Attachment'
```

```
        verbose_name_plural = u'Attachments'
```

```
AttachmentBase.attachment
```

The descriptor for the file attribute on the model instance. Returns a FieldFile when accessed so you can do stuff like:

```
>>> from myapp.models import MyModel
>>> instance = MyModel.objects.get(pk=1)
>>> instance.file.size
```

Assigns a file object on assignment so you can do:

```
>>> with open('/path/to/hello.world', 'r') as f:
...     instance.file = File(f)
```

AttachmentBase.filename

AttachmentBase.get\_absolute\_url()

## Views

class atom.views.ActionMessageMixin

post(request, \*args, \*\*kwargs)

class atom.views.ActionMixin

action()

get\_success\_url()

post(request, \*args, \*\*kwargs)

success\_url = None

class atom.views.ActionView(\*\*kwargs)

template\_name\_suffix = 'action'

class atom.views.BaseActionView(\*\*kwargs)

Base view for action on an object. Using this base class requires subclassing to provide a response mixin.

class atom.views.CreateMessageMixin

get\_form\_valid\_message()

class atom.views.DeleteMessageMixin

delete(request, \*args, \*\*kwargs)

get\_success\_message()

hide\_field = None

class atom.views.FormInitialMixin

get\_initial(\*args, \*\*kwargs)

class atom.views.MessageMixin

get\_success\_message()

```
        success_message = None
class atom.views.UpdateMessageMixin

    get_form_valid_message()
```

## Mixins

```
class atom.mixins.AdminTestCaseMixin

    QUERY_LIMIT = 30
    change_viewname = None
    changelist_viewname = None
    delete_viewname = None
    factory_cls = None
    get_change_viewname()
    get_changelist_viewname()
    get_delete_viewname()
    get_factory_cls()
    get_history_viewname()
    history_viewname = None
    model = None
    setUp()
    test_change_view_queries_limit()
    test_changelist_queries_limit()
    test_status_change_view()
    test_status_changelist()
    test_status_delete_view()
    test_status_history_view()
    user_factory_cls = None
```



### a

- `atom.ext.crispy_forms.filters`, [13](#)
- `atom.ext.crispy_forms.forms`, [13](#)
- `atom.ext.crispy_forms.views`, [14](#)
- `atom.ext.guardian.forms`, [9](#)
- `atom.ext.guardian.tests`, [10](#)
- `atom.filters`, [23](#)
- `atom.forms`, [23](#)
- `atom.mixins`, [25](#)
- `atom.models`, [23](#)
- `atom.views`, [24](#)





## A

abstract (atom.models.AttachmentBase.Meta attribute), 23

action() (atom.views.ActionMixin method), 24

action\_text (atom.ext.crispy\_forms.forms.SingleButtonMixin attribute), 14

ActionMessageMixin (class in atom.views), 24

ActionMixin (class in atom.views), 24

ActionView (class in atom.views), 24

AdminTestCaseMixin (class in atom.mixins), 25

atom.ext.crispy\_forms.filters (module), 13

atom.ext.crispy\_forms.forms (module), 13

atom.ext.crispy\_forms.views (module), 14

atom.ext.guardian.forms (module), 9

atom.ext.guardian.tests (module), 10

atom.filters (module), 23

atom.forms (module), 23

atom.mixins (module), 25

atom.models (module), 23

atom.views (module), 24

attachment (atom.models.AttachmentBase attribute), 23

AttachmentBase (class in atom.models), 23

AttachmentBase.Meta (class in atom.models), 23

AuthorMixin (class in atom.forms), 23

## B

base\_fields (atom.ext.guardian.forms.TranslatedUserObjectPermissionsForm attribute), 9

BaseActionView (class in atom.views), 24

BaseTableFormSet (class in atom.ext.crispy\_forms.forms), 13

## C

change\_viewname (atom.mixins.AdminTestCaseMixin attribute), 25

changelist\_viewname (atom.mixins.AdminTestCaseMixin attribute), 25

CreateMessageMixin (class in atom.views), 24

CrispyFilterMixin (class in atom.ext.crispy\_forms.filters), 13

## D

declared\_fields (atom.ext.guardian.forms.TranslatedUserObjectPermissionsForm attribute), 9

delete() (atom.views.DeleteMessageMixin method), 24

delete\_viewname (atom.mixins.AdminTestCaseMixin attribute), 25

DeleteMessageMixin (class in atom.views), 24

## F

factory\_cls (atom.mixins.AdminTestCaseMixin attribute), 25

filename (atom.models.AttachmentBase attribute), 24

form (atom.ext.crispy\_forms.filters.CrispyFilterMixin attribute), 13

form\_class (atom.ext.crispy\_forms.filters.CrispyFilterMixin attribute), 13

form\_helper\_cls (atom.ext.crispy\_forms.forms.HelperMixin attribute), 13

form\_method (atom.ext.crispy\_forms.forms.FormsetHelper attribute), 13

form\_tag (atom.ext.crispy\_forms.forms.FormsetHelper attribute), 13

form\_valid() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

FormHorizontalMixin (class in atom.ext.crispy\_forms.forms), 13

FormInitialMixin (class in atom.views), 24

formset (atom.ext.crispy\_forms.views.FormSetMixin attribute), 14

formset\_cls (atom.ext.crispy\_forms.views.FormSetMixin attribute), 14

formset\_invalid() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

formset\_valid() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

FormsetHelper (class in atom.ext.crispy\_forms.forms), 13

FormSetMixin (class in atom.ext.crispy\_forms.views), 14

## G

get\_absolute\_url() (atom.models.AttachmentBase method), 24

get\_change\_viewname() (atom.mixins.AdminTestCaseMixin method), 25

get\_changelist\_viewname() (atom.mixins.AdminTestCaseMixin method), 25

get\_context\_data() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_delete\_viewname() (atom.mixins.AdminTestCaseMixin method), 25

get\_factory\_cls() (atom.mixins.AdminTestCaseMixin method), 25

get\_form() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_form\_valid\_message() (atom.views.CreateMessageMixin method), 24

get\_form\_valid\_message() (atom.views.UpdateMessageMixin method), 25

get\_formset() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_formset\_kwargs() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_formset\_valid\_message() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_history\_viewname() (atom.mixins.AdminTestCaseMixin method), 25

get\_initial() (atom.views.FormInitialMixin method), 24

get\_instance() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_permission() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

get\_permission\_object() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

get\_success\_message() (atom.views.DeleteMessageMixin method), 24

get\_success\_message() (atom.views.MessageMixin method), 24

get\_success\_url() (atom.ext.crispy\_forms.views.FormSetMixin method), 14

get\_success\_url() (atom.views.ActionMixin method), 24

get\_url() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

grant\_permission() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

## H

HelperMixin (class in atom.ext.crispy\_forms.forms), 13

hide\_field (atom.views.DeleteMessageMixin attribute), 24

history\_viewname (atom.mixins.AdminTestCaseMixin attribute), 25

inline\_form\_cls (atom.ext.crispy\_forms.views.FormSetMixin attribute), 14

inline\_model (atom.ext.crispy\_forms.views.FormSetMixin attribute), 14

InlineTableFormSet (class in atom.ext.crispy\_forms.forms), 13

## L

login\_permitted\_user() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

## M

media (atom.ext.guardian.forms.TranslatedUserObjectPermissionsForm attribute), 9

MessageMixin (class in atom.views), 24

model (atom.mixins.AdminTestCaseMixin attribute), 25

## P

partial() (atom.forms.PartialMixin class method), 23

PartialMixin (class in atom.forms), 23

permission (atom.ext.guardian.tests.PermissionStatusMixin attribute), 10

PermissionStatusMixin (class in atom.ext.guardian.tests), 10

PermissionsTranslationMixin (class in atom.ext.guardian.forms), 9

post() (atom.views.ActionMessageMixin method), 24

post() (atom.views.ActionMixin method), 24

## Q

QUERY\_LIMIT (atom.mixins.AdminTestCaseMixin attribute), 25

## S

save() (atom.forms.AuthorMixin method), 23

setUp() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

setUp() (atom.mixins.AdminTestCaseMixin method), 25

SingleButtonMixin (class in atom.ext.crispy\_forms.forms), 13

status\_anonymous (atom.ext.guardian.tests.PermissionStatusMixin attribute), 10

status\_has\_permission (atom.ext.guardian.tests.PermissionStatusMixin attribute), 10

status\_no\_permission (atom.ext.guardian.tests.PermissionStatusMixin attribute), 10

success\_message (atom.views.MessageMixin attribute), 24

success\_url (atom.views.ActionMixin attribute), 24

## T

TableFormSet (class in atom.ext.crispy\_forms.forms), 14

TableFormSetHelper (class in atom.ext.crispy\_forms.forms), 14

TableFormSetMixin (class in atom.ext.crispy\_forms.forms), 14

template (atom.ext.crispy\_forms.forms.TableFormSetHelper attribute), 14

template\_name\_suffix (atom.views.ActionView attribute), 24

test\_change\_view\_queries\_limit() (atom.mixins.AdminTestCaseMixin method), 25

test\_changelist\_queries\_limit() (atom.mixins.AdminTestCaseMixin method), 25

test\_status\_change\_view() (atom.mixins.AdminTestCaseMixin method), 25

test\_status\_changelist() (atom.mixins.AdminTestCaseMixin method), 25

test\_status\_code\_for\_anonymous\_user() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

test\_status\_code\_for\_privileged\_user() (atom.ext.guardian.tests.PermissionStatusMixin method), 10

test\_status\_code\_for\_signed\_user() (atom.ext.guardian.tests.PermissionStatusMixin method), 11

test\_status\_delete\_view() (atom.mixins.AdminTestCaseMixin method), 25

test\_status\_history\_view() (atom.mixins.AdminTestCaseMixin method), 25

TranslatedUserObjectPermissionsForm (class in atom.ext.guardian.forms), 9

## U

UpdateMessageMixin (class in atom.views), 25

url (atom.ext.guardian.tests.PermissionStatusMixin attribute), 11

user\_factory\_cls (atom.mixins.AdminTestCaseMixin attribute), 25

## V

verbose\_name (atom.models.AttachmentBase.Meta attribute), 23

verbose\_name\_plural (atom.models.AttachmentBase.Meta attribute), 23