
djangoCMS-page-tags Documentation

Release 1.2.0.dev1

Iacopo Spalletti

Feb 13, 2024

CONTENTS

1	djangocms-page-tags	3
1.1	Quickstart	3
1.2	Usage	3
1.3	Templatetags	4
1.4	Documentation	4
2	Installation	5
2.1	Dependencies	5
3	Usage	7
3.1	Assigning tags to pages / titles	7
3.2	Retrieving tags in the template	7
4	djangocms_page_tags.utils	9
5	Contributing	11
5.1	Types of Contributions	11
6	Credits	15
6.1	Development Lead	15
6.2	Contributors	15
7	History	17
7.1	1.1.0 (2023-05-08)	17
7.2	1.0.0 (2022-08-27)	17
7.3	0.9.0 (2020-12-05)	17
7.4	0.8.0 (2019-12-27)	17
7.5	0.7.0 (2019-08-22)	18
7.6	0.6.2 (2019-03-09)	18
7.7	0.6.1 (2018-03-29)	18
7.8	0.6.0 (2017-12-30)	18
7.9	0.5.0 (2016-12-02)	18
7.10	0.4.2 (2016-01-08)	18
7.11	0.4.1 (2015-10-03)	19
7.12	0.4.0 (2015-06-29)	19
7.13	0.3.0 (2014-04-20)	19
7.14	0.2.1 (2014-03-29)	19
7.15	0.2.0 (2014-02-02)	19
7.16	0.1.4 (2014-01-26)	19
7.17	0.1.3 (2014-01-22)	19
7.18	0.1.2 (2014-01-16)	20

7.19	0.1.1 (2014-01-16)	20
7.20	0.1.0 (2014-01-15)	20
8	Indices and tables	21
	Python Module Index	23
	Index	25

Contents:

DJANGOCMS-PAGE-TAGS

Tagged pages for django CMS 3

Python: 3.7, 3.8, 3.9, 3.10

Django: 2.2, 3.2

django CMS: 3.7 - 3.10

1.1 Quickstart

Install.djangocms-page-tags:

```
pip install.djangocms-page-tags
```

Then add it to INSTALLED_APPS along with its dependencies:

```
"taggit",  
"taggit_autosuggest",  
"djangocms_page_tags",
```

Add taggit_autosuggest to urlconf:

```
path("taggit_autosuggest", include("taggit_autosuggest.urls")),
```

Execute migration:

```
$ python manage.py migrate
```

1.2 Usage

You will find two new items in the toolbar Page menu:

- Title tags (per language)
- Page tags (global)

These items allows to add tags to Title and Page instances, respectively

1.3 Templatetags

djangocms-page-tags allows showing tags using four templatetags

- `include_page_tags`
- `include_title_tags`
- `page_tags`
- `title_tags`

1.4 Documentation

For further documentation see <https://djangocms-page-tags.readthedocs.io/>

INSTALLATION

1. Install `django-cms-page-tags`:

```
$ pip install django-cms-page-tags
```

or from the repository:

```
pip install -e https://github.com/nephila/django-cms-page-tags#egg=django-cms-page-  
↪ tags
```

2. Then add it to `INSTALLED_APPS` along with its dependencies:

```
"taggit",  
"taggit_autosuggest",  
"django_cms_page_tags",
```

3. Add `taggit_autosuggest` to `urlpatterns`:

```
path("taggit_autosuggest", include("taggit_autosuggest.urls")),
```

4. Synchronize the database with `migrate`:

```
$ python manage.py migrate
```

5. That's all!

2.1 Dependencies

- `django-taggit` `>= 0.11`
- `django-taggit-autosuggest` `>= 0.2.1`

you may want to install `django-taggit-templatetags` for better tag management in the template.

3.1 Assigning tags to pages / titles

Tags can be assigned from the admin interface or the toolbar.

In the toolbar you will find a Tags submenu in the Page menu, with two or more sub-items:

- Common: it allows to edit page-wide (language independent) tags;
- One entry per active language to edit language dependent tags.

3.2 Retrieving tags in the template

`djangoCMS-page-tags` provides both a tag and an inclusion tag for each object.

3.2.1 `include_page_tags` and `include_title_tags`

These templatetags retrieve the tags for the given object and render them according to the relative template (`djangoCMS-page-tags/template/page_tags.html` and `djangoCMS-page-tags/template/title_tags.html`) respectively.

You can override the template using the standard django mechanism. Tags are available in the context variable `tags_list`.

Arguments:

- `page_lookup` (see [page_lookup](#) for more information)
- `language` (optional)
- `site` (optional)

For performance reason is advisable to always use a Page object as `page_lookup` parameter.

3.2.2 `page_tags` and `title_tags`

These templatetags pulls the tags for the given object and save them in the `varname` context variable.

Arguments:

- `page_lookup` (see [page_lookup](#) for more information)
- `language` (optional)
- `site` (optional)
- `varname` (required)

For performance reason is advisable to always use a Page object as `page_lookup` parameter.

DJANGOCMS_PAGE_TAGS.UTILS

Warning: While functions in this module can be used in code to work with the tags, they are by no means to be considered as a stable API, so they may change in future release.

`djangoCMS_page_tags.utils.get_cache_key(request, page, lang, site_id, title)`

Create the cache key for the current page and tag type

`djangoCMS_page_tags.utils.get_page_tags(page)`

Retrieves all the tags for a Page instance.

Parameters

page – a Page instance

Returns

list or queryset of attached tags

Type

List

`djangoCMS_page_tags.utils.get_page_tags_from_request(request, page_lookup, lang, site, title=False)`

Get the list of tags attached to a Page or a Title from a request from usual *page_lookup* parameters.

Parameters

- **request** – request object
- **page_lookup** – a valid *page_lookup* argument
- **lang** – a language code
- **site** – a site id
- **title** – a boolean to extract the Page (if False) or Title instance

Returns

list of tags

Type

List

`djangoCMS_page_tags.utils.get_title_tags(page, lang)`

Retrieves all the tags for a Title (given as page and language). This function does not use fallbacks to retrieve title object.

Parameters

- **page** – a Page instance

- **lang** – a language code

Returns

list or queryset of attached tags

Type

List

`djangocms_page_tags.utils.get_title_tags_from_request(request, page_lookup, lang, site)`

Get the list of tags attached to a Title from a request from usual *page_lookup* parameters.

Parameters

- **request** – request object
- **page_lookup** – a valid *page_lookup* argument
- **lang** – a language code
- **site** – a site id

Returns

list of tags attached to the given Title

Type

List

`djangocms_page_tags.utils.page_has_tag(page, tag)`

Check if a Page object is associated with the given tag.

Parameters

- **page** – a Page instance
- **tag** – a Tag instance or a slug string.

Returns

whether the Page instance has the given tag attached (False if no Page or no attached PageTags exists)

Type

Boolean

`djangocms_page_tags.utils.title_has_tag(page, lang, tag)`

Check if a Title object is associated with the given tag. This function does not use fallbacks to retrieve title object.

Parameters

- **page** – a Page instance
- **lang** – a language code
- **tag** – a Tag instance or a slug string.

Returns

whether the Title instance has the given tag attached (False if no Title or no attached TitleTags exists)

Type

Boolean

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:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/nephila/djangocms-page-tags/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.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

5.1.4 Write Documentation

djangocms-page-tags could always use more documentation, whether as part of the official djangocms-page-tags docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/nephila/djangoCMS-page-tags/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 djangoCMS-page-tags for local development.

1. Fork the djangoCMS-page-tags repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/djangoCMS-page-tags.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 djangoCMS-page-tags
$ cd djangoCMS-page-tags/
$ pip install -r requirements-test.txt
$ pip install -e .
```

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:

```
$ tox
```

To get tox, pip install it 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.

Development tips

This project allows you to use [pre-commit](#) to ensure an easy compliance to the project code styles.

If you want to use it, install it globally (for example with `pip3 install --user precommit`, but check [installation instruction](#). When first cloning the project ensure you install the git hooks by running `pre-commit install`.

From now on every commit will be checked against our code style.

Check also the available tox environments with `tox -l`: the ones not marked with a python version number are tools to help you work on the project buy checking / formatting code style, running docs etc.

Testing tips

You can test your project using any specific combination of python, django and django cms.

For example `tox -epy37-django30-cms37` runs the tests on python 3.7, Django 3.0 and django CMS 3.7.

5.1.6 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. Pull request must be named with the following naming scheme:

`<type>/(<optional-task-type>-)<number>-description`

See below for available types.

2. The pull request should include tests.
3. If the pull request adds functionality, the docs should be updated. Documentation must be added in docs directory, and must include usage information for the end user. In case of public API method, add extended docstrings with full parameters description and usage example.
4. Add a changes file in changes directory describing the contribution in one line. It will be added automatically to the history file upon release. File must be named as `<issue-number>.<type>` with type being:
 - `.feature`: For new features.
 - `.bugfix`: For bug fixes.
 - `.doc`: For documentation improvement.
 - `.removal`: For deprecation or removal of public API.
 - `.misc`: For general issues.

Check [towncrier](#) documentation for more details.

5. The pull request should work for all python / django / django CMS versions declared in tox.ini. Check the CI and make sure that the tests pass for all supported versions.

5.1.7 Release a version

1. Update authors file
2. Merge develop on master branch
3. Bump release via task: `inv tag-release (major|minor|patch)`
4. Update changelog via towncrier: `towncrier --yes`
5. Commit changelog with `git commit --amend` to merge with bumpversion commit
6. Create tag `git tag <version>`
7. Push tag to github
8. Publish the release from the tags page
9. If pipeline succeeds, push master
10. Merge master back on develop
11. Bump development version via task: `inv tag-dev -l (major|minor|patch)`
12. Push develop

CREDITS

6.1 Development Lead

- Iacopo Spalletti <i.spalletti@nephila.digital>

6.2 Contributors

- Adam Chainz
- Mark Walker
- Serge Fedoruk

HISTORY

7.1 1.1.0 (2023-05-08)

7.1.1 Features

- Update GH actions / linting configuration (#40)

7.2 1.0.0 (2022-08-27)

7.2.1 Bugfixes

- Add support for Django 3.2 / django CMS 3.10 (#33)
- Skip creating toolbar item in page types (#36)

7.3 0.9.0 (2020-12-05)

7.3.1 Features

- Add support for Python 3.9 (#29)
- migrate from travis, drop Python 2/Django Cms 3.6 support (#10210)

7.4 0.8.0 (2019-12-27)

- Add Django 2.2
- Add django CMS 3.7
- Add Python 3.7
- Drop compatibility with django CMS < 3.6

7.5 0.7.0 (2019-08-22)

- Drop compatibility with Django < 1.11
- Drop compatibility with Python 3 < 3.5

7.6 0.6.2 (2019-03-09)

- Add Django 2.0, 2.1 support
- Add django CMS 3.6 support

7.7 0.6.1 (2018-03-29)

- Properly set migration dependencies
- Improve compatibility with django CMS 3.5
- Fix issues with tox on travis

7.8 0.6.0 (2017-12-30)

- Add Django 1.11 support
- Drop django CMS 3.2, 3.3

7.9 0.5.0 (2016-12-02)

- Add Django 1.10 support
- Add support for django CMS 3.3, 3.4
- Drop Django 1.7 and below
- Drop django CMS 3.1 and below

7.10 0.4.2 (2016-01-08)

- Fix handling of pages in non existing languages

7.11 0.4.1 (2015-10-03)

- Add support for django CMS 3.2
- Add support for Django 1.8
- Update translation
- PEP-8 / isort code cleanup

7.12 0.4.0 (2015-06-29)

- Add support for django CMS 3.1
- Add support for Django 1.7
- Use djangoCMS-helper for testing

7.13 0.3.0 (2014-04-20)

- Support for django CMS 3.0 final

7.14 0.2.1 (2014-03-29)

- Support for django CMS 3.0 RC1
- Add verbose description

7.15 0.2.0 (2014-02-02)

- Fix for django CMS 3 develop update

7.16 0.1.4 (2014-01-26)

- Metadata cleanup

7.17 0.1.3 (2014-01-22)

- Put toolbar menu items in a submenu

7.18 0.1.2 (2014-01-16)

- Documentation update

7.19 0.1.1 (2014-01-16)

- Add transifex support

7.20 0.1.0 (2014-01-15)

- Experimental release.

INDICES AND TABLES

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

PYTHON MODULE INDEX

d

`djangoCMS_page_tags.utils`, [9](#)

INDEX

D

`djangoCMS_page_tags.utils`
module, 9

G

`get_cache_key()` (in module *djangoCMS_page_tags.utils*), 9

`get_page_tags()` (in module *djangoCMS_page_tags.utils*), 9

`get_page_tags_from_request()` (in module *djangoCMS_page_tags.utils*), 9

`get_title_tags()` (in module *djangoCMS_page_tags.utils*), 9

`get_title_tags_from_request()` (in module *djangoCMS_page_tags.utils*), 10

M

module
`djangoCMS_page_tags.utils`, 9

P

`page_has_tag()` (in module *djangoCMS_page_tags.utils*), 10

T

`title_has_tag()` (in module *djangoCMS_page_tags.utils*), 10