
djangoCMS-conditional Documentation

Release 0.2.1


Iacopo Spalletti

Feb 22, 2018

Contents

1	djangoCMS-conditional	3
1.1	Documentation	3
1.2	Quickstart	3
1.3	Features	3
1.4	Caveats	4
2	Contributing	5
2.1	Types of Contributions	5
2.2	Get Started!	6
2.3	Pull Request Guidelines	7
2.4	Tips	7
3	Credits	9
3.1	Development Lead	9
3.2	Contributors	9
3.3	Acknowledgements	9
4	History	11
4.1	0.1.2 (2018-01-01)	11

Contents:

 Django CMS plugin that shows content if a user is logged in and a member of a specific Django group.

1.1 Documentation

The full documentation is at <https://djangoCMS-conditional.readthedocs.org>

1.2 Quickstart

1. Install djangoCMS-conditional:: `pip install djangoCMS-conditional`
2. Add “djangoCMS_conditional” to your `INSTALLED_APPS` setting like this:

```
INSTALLED_APPS = [  
    ...  
    'djangoCMS_conditional',  
]
```

3. Run `python manage.py migrate` to create the djangoCMS_conditional models.

1.3 Features

Shows and hides child plugins according to group membership, as configured in the plugin instance.

1.4 Caveats

This plugin only prevents rendering of plugins, just like `djangocms-timer`, and is subject to the same limitations:

In its current form, plugin won't save you from the queries to retrieve child plugins due to the way plugin rendering works in django CMS. Still, the `render` method of child plugins is not executed (and grandchildren plugins are not retrieved) with mitigating effect on performance hit.

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

2.1 Types of Contributions

2.1.1 Report Bugs

Report bugs at <https://github.com/rhooper/djangocms-conditional/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.

2.1.2 Fix Bugs

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

2.1.3 Implement Features

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

2.1.4 Write Documentation

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

2.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/nephila/djangocms-conditional/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 :)

2.2 Get Started!

Ready to contribute? Here's how to set up *djangocms-conditional* for local development.

1. Fork the *djangocms-conditional* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/djangocms-conditional.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-conditional
$ cd djangocms-conditional/
$ 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 djangocms_conditional 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.

2.3 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 2.7, and 3.4, 3.5, and 3.6. Run tox to verify. Pyenv will help you install multiple pythons. Check Travis CI: https://travis-ci.org/rhooper/djangocms-conditional/pull_requests

2.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_djangocms_conditional tests.test_djangocms_  
↳conditional
```


3.1 Development Lead

- Roy Hooper <rhooper@toybox.ca>

3.2 Contributors

None yet. Why not be the first?

3.3 Acknowledgements

This module is adapted from django-timer by Iacopo Spalletti <i.spalletti@nephila.it> at <https://github.com/nephila/djangocms-timer>

4.1 0.1.2 (2018-01-01)

- First release on PyPi.