djangocms-redirect Documentation

Release 0.7.2

Nephila

Sep 26, 2023

Contents

CHAPTER 1

djangocms-redirect

A django CMS enabled application to handle redirects

This is heavily borrowed from `django.contrib.redirects` with three major changes:

- Selection of django CMS pages
- Selection of redirect status code
- Middleware can processed in the request or response phase

1.1 Why using process_request?

Doing database queries in the middleware `process_request` is heavily discouraged as it's a performance hit, especially when doing redirects which are just a tiny part of the processed requests. Except that sometimes it's just what you need (for example to "hide" content without deleting / unpublishing it) By caching both existing and non existing redirects for a given URL the performance hit is minimized for the use cases that requires `process_request`.

1.2 Documentation

The full documentation is at https://djangocms-redirect.readthedocs.io.

1.3 Installation

See https://djangocms-redirect.readthedocs.io/en/latest/installation.html

1.4 Features

- Set old and new path, by selection existing django CMS pages or writing down the complete address
- Select the redirect status code (301, 302)
- Support for status code 410

1.5 Credits

Tools used in rendering this package:

- Cookiecutter
- cookiecutter-djangopackage-helper

Contents:

1.5.1 Installation

- Install djangocms-redirect:

```
pip install djangocms-redirect
```

- Add to INSTALLED_APPS:

```
INSTALLED_APPS = [
    ...
    djangocms_redirect
    ...
]
```

- Add to MIDDLEWARE_CLASSES:

  If you intend to use `process_request` the add it near the top (after `django.middleware.cache.UpdateCacheMiddleware`); if using `process_response` add at the bottom (before `django.middleware.cache.FetchFromCacheMiddleware`):

```
MIDDLEWARE_CLASSES = [
    ...
    djangocms_redirect.middleware.RedirectMiddleware
    ...
]
```

- Choose if you want to process the redirect during the request (default) or response by setting:
  - `DJANGOCMS_REDIRECT_USE_REQUEST = True`: during request
  - `DJANGOCMS_REDIRECT_USE_REQUEST = False`: during response
- Migrate:

```
python manage.py migrate
```

The go to `http://mysite.com/admin/djangocms_redirect/` and create redirect instances.

Settings

- `DJANGOCMS_REDIRECT_USE_REQUEST`: If `True` the redirect check will be done in the request phase, to allow preempting any other logic. **Beware**: this will result in extra queries on **each** request because the redirects will be checked before the view logic triggers. If `False` the redirect will be checked in the response phase.
- `DJANGOCMS_REDIRECT_CACHE_TIMEOUT`: You can provide a custom cache timeout (Default: 3600 sec)
- `DJANGOCMS_REDIRECT_404_ONLY`: If `True` (the default) and `DJANGOCMS_REDIRECT_USE_REQUEST=False` the redirect will be checked only for responses that return 404 (the default `django.contrib.redirect` behavior). This is the lowest impact option in terms of performance and the advised configuration.

1.5.2 Usage

To manage redirect rules navigate to the `Redirect` section of the django admin (usually `http://www.yoursite.com/admin/djangocms_redirect/redirect`).

For each redirect you must provide:

- **Site**: The site for which you want to add a redirect.
- **Redirect from**: The path that need to be redirected: you can type any URL or select an existing django CMS page.
- **Redirect to**: The path to which the request will be redirected: you can type any URL or select an existing django CMS page.
- **Response code**: You can select 3 types of status_code header: 301 (permanent redirect), 302 (temporary redirect) or 410 (permanent unavailable resource).

Each **redirect from** URL must be unique and start with a slash. If you leave out the leading slash when creating a redirect, it is added automatically.

Redirect examples

From: `/something/old-path` (no trailing slash) To: `/something-else/new-path/` `APPEND_SLASH` setting is `True`

- When on Django >= 4.2:
  - visiting `/something/old-path` will correctly redirect to `something-else/new-path/`
  - visiting `/something/old-path/` will return a 404
- When on Django < 4.2:
  - visiting `/something/old-path` will redirect to `something/old-path/`
  - visiting `/something/old-path/` will return a 404

From: `/something/old-path/` (trailing slash) To: `/something-else/new-path/` `APPEND_SLASH` setting is `True`

- When on Django >= 4.2:
  - visiting `/something/old-path` will correctly redirect to `something-else/new-path/`
  - visiting `/something/old-path/` will correctly redirect to `something-else/new-path/`
- When on Django < 4.2:

  - visiting `/something/old-path` will redirect to `something/old-path/` and then will correctly redirect to `something-else/new-path/`
  - visiting `/something/old-path/` will correctly redirect to `something-else/new-path/`

From: `/something/old-path` (no trailing slash) To: `/something-else/new-path/` `APPEND_SLASH` setting is `False`

- Every version of Django:
  - visiting `/something/old-path` will correctly redirect to `something-else/new-path/`
  - visiting `/something/old-path/` will return a 404

From: `/something/old-path/` (trailing slash) To: `/something-else/new-path/` `APPEND_SLASH` setting is `False`

- Every version of Django:
  - visiting `/something/old-path` will correctly redirect to `something-else/new-path/`
  - visiting `/something/old-path/` will correctly redirect to `something-else/new-path/`

Subpath matching

Each redirect can match the exact incoming request path (the default behavior) or a subpath.

Subpath matching comes in two behaviour:

Plain subpath matching

The registered redirects will be checked against the incoming URL and the longest string matching the beginning of the request path will be selected.

The request will be then redirected by replacing the matching **Redirect from** with the **Redirect to** in the original URL.

Example

- Incoming request: `/en/some/path/`
- Redirect from: `/en/some`
- Redirect to: `/en/other`
- Resulting redirect: `/en/other/path/`

Catchall redirect

As in **plain subpath matching** the registered redirects will be checked against the incoming URL and the longest string matching the beginning of the request path will be selected.

The request will be then redirected to the **Redirect to** without further changes.

Example

- Incoming request: `/en/some/path/`
- Redirect from: `/en/some`
- Redirect to: `/en/other`
- Resulting redirect: `/en/other`

1.5.3 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

Please read the instructions here to start contributing to *djangocms-redirect*.

1.5.4 Credits

Development Lead

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

Previous maintainers

- Paolo Romolini <p.romolini@nephila.it>

Contributors

- Adam Chainz
- Gaetano D'Onghia <g.donghia@frankhood.it>
- Giovanni Bottalico <g.bottalico@frankhood.it>
- Nicola Ciccarone <n.ciccarone@frankhood.it>
- schroedingersZombie
- Vladimir Kuvandjiev
- Xiphoseer

1.5.5 History

0.7.2 (2023-09-26)

Features

- Migrate to bump-my-version (#56)

0.7.1 (2023-08-11)

Features

- Add missing subpath_match and catchall_redirect fields in admin (#44)

Bugfixes

- Fix django-multisite compatibility issue (#47)

0.7.0 (2023-08-09)

Features

- Add django 4.2 compatibility, drop python<3.9, djangocms<3.9 and django<3.2 (#42)

0.6.0 (2020-11-15)

Features

- Drop Python 2, Django < 2.2 - Update toolchain (#39)
- Fix Handling of trailing slashes in redirects (#31)

Unreleased

- Nothing yet

0.5.0 (2019-12-27)

- Add compatibility with Django 2.2
- Drop compatibility with Django < 1.11
- Drop compatibility with django CMS < 3.6
- Move to django-app-helper
- Add support to match unquoted strings as redirect old path

0.4.0 (2019-08-22)

- Add subpath matching

0.3.1 (2019-07-13)

- Ignore querystring when matching redirect objects

0.3.0 (2019-03-11)

- Added compatibility to Django 2.0, 2.1

0.2.3 (unreleased)

- Add support to match unquoted strings as redirect old path

0.2.2 (2019-06-02)

- Ignore querystring when matching redirect objects

0.2.1 (2019-04-22)

- Fixed compatibility issue with Django 1.8

0.2.0 (2018-11-03)

- Updated for Django 1.11
- Added configurable cache timeout
- Added configuration option to check redirect on 404 only

0.1.1 (2017-11-19)

- Added missing migration.
- Fixed compatibility issue with Django 1.8

0.1.0 (2016-02-01)

- First release on PyPI.