thecut-durationfield Documentation

Release 2.0.2

The Cut Creative

August 14, 2016

Contents

Contents:

CHAPTER 1

Welcome to thecut-durationfield

This app provides a custom Django model field, `RelativeDeltaField`, and related form fields and widgets. `RelativeDeltaField` stores time durations using ISO 8601 representations, and returns `dateutil.relativedelta` objects which may be used directly with `datetime.datetime` objects.

This project was inspired by packages such as django-durationfield. However, this project focuses on:

1. providing a database-agnostic, standards-compliant way of storing the durations in the database (using ISO 8601).
2. returning `dateutil.relativedelta` objects that can be used to perform calculations on `datetime.datetime` objects.

Note that django-durationfield provides the ability to filter querysets based on the relative size of the stored duration, which is not possible with this project. I.e., you can't use `__lt` and `__gt` etc., when filtering by fields provided by this project.

1.1 Documentation

The full documentation is at https://thecut-durationfield.readthedocs.org.

1.2 Quickstart

Install `thecut-durationfield` using the *Installation instructions*.

1.2.1 Model field

```
from django.db import models
from datetime import datetime
from thecut.durationfield.models import RelativeDeltaField


class MyModel(models.Model):
    duration = RelativeDeltaField(blank=True, null=True)


my_instance = MyModel(duration='P7D')
datetime(2014, 1, 1) + my_instance.duration  # datetime(2014, 1, 8, 0, 0)
```

1.2.2 Form field

Two form fields are provided: `RelativeDeltaChoiceField` and `RelativeDeltaTextInput`:

```
from django import forms
from thecut.durationfield.models import RelativeDeltaChoiceField

DURATIONS = [
    ('', 'Never'),
    ('P7D', 'One week'),
    ('P1M', 'One month'),
]

class MyForm(forms.ModelForm):

    duration = RelativeDeltaChoiceField(choices=DURATIONS)
```

or, if you'd prefer to type in the (ISO 8601 compliant) value manually:

```
from django import forms
from thecut.durationfield.forms import RelativeDeltaTextInput

class MyForm(forms.ModelForm):

    duration = RelativeDeltaTextInput()
```

1.3 Credits

See *Credits*.

CHAPTER 2

Installation instructions

1. Install via pip / pypi:

```
$ pip install thecut-durationfield
```

2. Add to your project's `INSTALLED_APPS` setting:

```
INSTALLED_APPS = [
    # ...
    'thecut.durationfield'
    # ...
]
```

3. Sync your project's migrations:

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

CHAPTER 3

Usage

CHAPTER 4

Testing

4.1 Running unit tests

4.1.1 Using your system's Python / Django

You can perform basic testing against your system's Python / Django.

1. Install the test suite requirements:

```
$ pip install -r requirements-test.txt
```

2. Ensure a version of Django is installed:

```
$ pip install Django
```

3. Run the test runner:

```
$ python runtests.py
```

4.1.2 Using a virtualenv

You can use `virtualenv` to test without polluting your system's Python environment.

1. Install `virtualenv`:

```
$ pip install virtualenv
```

2. Create and activate a `virtualenv`:

```
$ cd thecut-durationfield
$ virtualenv .
$ source bin/activate
(thecut-durationfield) $
```

3. Follow 'Using your system's Python / Django' above.

4.1.3 Using tox

You can use tox to automatically test the application on a number of different Python and Django versions.

1. Install `tox`:

```
$ pip install -r requirements-test.txt
```

2. Run `tox`:

```
(thecut-durationfield) $ tox --recreate
```

Tox assumes that a number of different Python versions are available on your system. If you do not have all required versions of Python installed on your system, running the tests will fail. See `tox.ini` for a list of Python versions that are used during testing.

4.1.4 Test coverage

The included `tox` configuration automatically detects test code coverage with `coverage`:

```
$ coverage report
```

4.2 Available tests

4.2.1 `TestISO8061DurationField`

4.2.2 `TestRelativeDeltaField`

CHAPTER 5

History

5.1 2.0.2 (2016-08-15)

- Documentation updates.

5.2 2.0.1 (2016-08-15)

- Documentation fixes.
- Testing fixes.

5.3 2.0 (2016-08-15)

- Added support for Django 1.10.
- Removed support for Django < 1.8.
- Restructured test suite.
- Restructured documentation.

5.4 1.0.8 (2015-08-26)

- Improved handling of seconds and milliseconds.

5.5 1.0.8 (2015-08-26)

- Improved handling of seconds and milliseconds.

5.6 1.0.7 (2015-03-17)

- Added Python 3 support.

5.7 1.0.6 (2014-07-28)

- Fix an issue which caused an empty `relativedelta` to be returned for a database NULL value.
- Get `tox` up and running.
- Update package for public release.

5.8 1.0.5 (2014-03-19)

- Remove `distribute` from `install_requires`.

5.9 1.0.4 (2013-12-17)

- Fixed an issue with Postgres's fixed-length 64 character field.

5.10 1.0.3 (2013-09-28)

- Minor code cleanup.

5.11 1.0.2 (20132-08-08)

- Add a Select widget for friendlier form input.

5.12 1.0.1 (2013-07-25)

- Fixes to south introspection rules.

5.13 1.0 (2013-07-25)

- First useful release with base model and form fields.

5.14 0.1 (2013-06-10)

- Initial release, mostly useless.

CHAPTER 6

Credits

- Elena Williams <elena.williams@thecut.net.au>
- Matt Austin <matt.austin@thecut.net.au>
- Josh Crompton <josh.crompton@thecut.net.au>
- Bertrand Svetchine https://github.com/bsvetchine
- Kye Russell <kye.russell@thecut.net.au>
- Guillaume Andreu Sabater https://github.com/AGASS007