
django-durationfield Documentation

Release 0.5.5

John Paulett

Aug 31, 2017

Contents

1	Django Versions	3
2	Usage	5
3	Example	7
4	Years and Months	9
5	Development	11
5.1	Testing	11
6	Changelog	13
7	Authors	15

Warning: Django 1.8 introduced a native `DurationField`. It is highly recommended to Django's `DurationField` instead of this reusable app, as ongoing maintenance of this app is likely to end.

Currently there is no automatic migration path from `django-durationfield` to Django's `DurationField`, but pull requests are welcome.

A reusable application for a `DurationField` in Django.

This reusable app was conceived as a temporary solution for an old request to add native support for an interval or duration field to Django core, [#2443](#), “Add `IntervalField` to database models.” This app started from the 2010-01-25 patch by Adys (Jerome Leclanche), [DurationField.patch](#) and has evolved considerably as people have used it in their own applications.

There have been discussions as to the merit of including into a `DurationField` in Django core. As of the moment, it appears that most developers favor keeping `DurationField` separate from Django (both to prevent bloat and to allow rapid evolution of the `DurationField`'s implementation).

That being said, we have developed the `DurationField` so that if it ever does get merged into Django core, it should be simple for users to switch.

This is beta software, please test thoroughly before putting into production and report back any issues.

CHAPTER 1

Django Versions

django-durationfield supports Django 1.4.21, and Django 1.6.11 through Django 1.10+, with the goal to target the currently support versions of Django releases in the future. So as the Django Project drops support for older versions, django-durationfield will do the same.

Django 1.4.2 is a minimum version as it introduced [compatibility features](#) for supporting both Python 2 and Python 3. django-durationfield has support for Python 3.4 & 3.5.

Please report any bugs or patches in [improve version support](#).

CHAPTER 2

Usage

In models.py:

```
from durationfield.db.models.fields.duration import DurationField

class Time(models.Model):
    ...
    duration = DurationField()
    ...
```

In your forms:

```
from durationfield.forms import DurationField as FDurationField

class MyForm(forms.ModelForm):
    duration = FDurationField()
```

Note that database queries still need to treat the values as integers. If you are using things like aggregates, you will need to explicitly convert them to timedeltas yourself:

```
timedelta(microseconds=list.aggregate(sum=Sum('duration'))['sum'])
```


CHAPTER 3

Example

Enter the time into the textbox in the following format:

```
3 days 2:20:10
```

or:

```
3d 2:20:10
```

This is interpreted as:

```
3 days 2 hours 20 minutes 10 seconds
```

In your application it will be represented as a python `datetime.timedelta`:

```
3 days, 2:20:10
```

This will be stored into the database as a 'bigint'.

You can be rather more elaborate by using microseconds and weeks (or even months, and years with a configuration setting):

```
1 year, 2 months, 3 weeks, 4 days, 5:06:07.000008
```


CHAPTER 4

Years and Months

You will need to set a setting in your `settings.py` to allow your users to enter values for years or months. This causes a loss of precision because the number of days in a month is not exact. This has not been extensively tested.

To allow users to enter `X years` or `X months` add:

```
``DURATIONFIELD_ALLOW_YEARS = True``
```

or:

```
``DURATIONFIELD_ALLOW_MONTHS = True``
```

Currently, those fields will be translated into days using either 365 or 30 respectively.

To override this setting, add an entry into your settings named `DURATIONFIELD_YEARS_TO_DAYS` or `DURATIONFIELD_MONTHS_TO_DAYS` setting a new translation value.

CHAPTER 5

Development

Please fork and submit issues at <https://github.com/johnpaulett/django-durationfield>

Testing

If you are interested in developing django-duration field, you can use tox to test django-durationfield across all current versions of Django:

```
tox
```


CHAPTER 6

Changelog

0.5.5

- Add Django 1.11 support thanks to [Anton Linevych](#). However, recommend future projects use Django's native `DurationField`

0.5.4

- Handle cases when database returns a `:py:cls:'decimal.Decimal'` under Django 1.10. Cast the Decimal to a float (may be a lossy conversion)

0.5.3

- Add support for Django 1.10.

0.5.2

- Remove deprecation warning in Django 1.8

0.5.1

- Correctly parse microseconds. Previously, “0.01” would be incorrectly interpreted to mean 1 microsecond, instead of 10000 microseconds. Thanks to [‘Troy Grosfield >https://github.com/troygrosfield>’](https://github.com/troygrosfield) for bug report and patch.

0.5.0

- Raise a `ValidationError` when given an invalid time string. Thanks to [blueyed](#) for the bug report.

0.4.0

- Python 3 support. Drop support for Django < 1.4

CHAPTER 7

Authors

Thanks to the authors of the original DurationField patches, Marty Alchin, Jerome Leclanche, and Yuri Baburov.

Thanks to the contributors to django-durationfield:

- John Paulet (<https://github.com/johnpaulett>)
- Paul Oswald (<https://github.com/poswald>)
- Wes Winham (<https://github.com/winhamwr>)
- Guillaume Libersat (<https://github.com/glibersat>)
- Jason Mayfield (<https://github.com/jwmayfield>)
- silent1mezzo (<https://github.com/silent1mezzo>)
- Adam Coddington (<https://github.com/latestrevision>)
- Troy Grosfield (<https://github.com/troygrosfield>)
- Anton Linevych (<https://github.com/linevich>)