django-star-ratings Documentation

Release 0.9.2

Wildfish

Dec 10, 2021

Contents

Python 3 compatible ratings for Django.

Add ratings to any Django model with a template tag.

See full documentation.

Built by Wildfish. https://wildfish.com

CHAPTER 1

Requirements

- Python 3.6+.
- Django 2.2+

CHAPTER 2

Installation

Install from PyPI:

```
pip install django-star-ratings
```

add `star_ratings` to `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    ...
    'star_ratings'
)
```

sync your database:

```
python manage.py migrate
```

add the following to your urls.py:

```
path('ratings/', include('star_ratings.urls', namespace='ratings')),
```

Make sure `'django.core.context_processors.request',` is in `TEMPLATE_CONTEXT_PROCESSORS`.

CHAPTER 3

Usage

Add the following javascript and stylesheet to your template

```
{% load static %}
<html>
...
<link rel="stylesheet" href="{% static 'star-ratings/css/star-ratings.css' %}">
<script type="text/javascript" src="{% static 'star-ratings/js/dist/star-ratings.min.
↪js' %}"></script>
...
</html>
```

To enable ratings for a model add the following tag in your template

```
{% load ratings %}
<html>
...
{% ratings object %}
...
</html>
```

CHAPTER 4

Template tags

The template tag takes four arguments:

- `icon_height`: defaults to `STAR_RATINGS_STAR_HEIGHT`
- `icon_width`: defaults to `STAR_RATINGS_STAR_WIDTH`
- `read_only`: overrides the `editable` behaviour to make the widget read only
- `template_name`: overrides the tempalte to use for the widget

CHAPTER 5

Settings

To prohibit users from altering their ratings set `STAR_RATINGS_RERATE = False` in settings.py

To allow users to delete a rating by selecting the same score again, set `STAR_RATINGS_RERATE_SAME_DELETE = True` in settings.py, note that `STAR_RATINGS_RERATE` must be True if this is set.

To allow uses to delete a rating via a clear button, set ``STAR_RATINGS_CLEARABLE = True` `` in settings.py. This can be used with or without STAR_RATINGS_RERATE.

To change the number of rating stars, set `STAR_RATINGS_RANGE` (defaults to 5)

To enable anonymous rating set `STAR_RATINGS_ANONYMOUS = True`.

Please note that `STAR_RATINGS_RERATE`, `STAR_RATINGS_RERATE_SAME_DELETE` and `STAR_RATINGS_CLEARABLE` will have no affect when anonymous rating is enabled.

CHAPTER 6

Anonymous Rating

If anonymous rating is enabled only the ip address for the rater will be stored (even if the user is logged in). When a user rates an object a preexisting object will not be searched for, instead a new rating object will be created

If this value is changed your lookups will return different results!

To control the default size of stars in pixels set the values of `STAR_RATINGS_STAR_HEIGHT` and `STAR_RATINGS_STAR_WIDTH`. By default `STAR_RATINGS_STAR_WIDTH` is the same as `STAR_RATINGS_STAR_HEIGHT` and `STAR_RATINGS_STAR_HEIGHT` defaults to 32.

CHAPTER 7

Changing the star graphics

To change the star graphic, add a sprite sheet to `/static/star-ratings/images/stars.png` with the states aligned horizontally. The stars should be laid out in three states: full, empty and active.

You can also set `STAR_RATINGS_STAR_SPRITE` to the location of your sprite sheet.

CHAPTER 8

Customize widget template

You can customize ratings widget by creating `star_ratings/widget.html`. For example :

```
{% extends "star_ratings/widget_base.html" %}
{% block rating_detail %}
Whatever you want
{% endblock %}
```

See `star_ratings/widget_base.html` for other blocks to be extended.

CHAPTER 9

Ordering by ratings

The easiest way to order by ratings is to add a `GenericRelation` to the `Rating` model from your model:

```
from django.contrib.contenttypes.fields import GenericRelation
from star_ratings.models import Rating

class Foo(models.Model):
    bar = models.CharField(max_length=100)
    ratings = GenericRelation(Rating, related_query_name='foos')

Foo.objects.filter(ratings__isnull=False).order_by('ratings__average')
```

CHAPTER 10

Custom Rating Model

In some cases you may need to create your own rating model. This is possible by setting `STAR_RATING_RATING_MODEL` in your settings file. This can be useful to add additional fields or methods to the model. This is very similar to the how django handles swapping the user model (see [https://docs.djangoproject.com/en/1.10/topics/auth/customizing/#substituting-a-custom-user-model]).

For ease `AbstractBaseRating` is supplied. For example if you wanted to add the field `foo` to the rating model you would need to crate your rating model extending `AbstractBaseRating`:

```
./myapp/models.py

class MyRating(AbstractBaseRating):
    foo = models.TextField()
```

And add the setting to the setting file:

```
./settings.py

...
STAR_RATINGS_RATING_MODEL = 'myapp.MyRating'
...
```

NOTE: If you are using a custom rating model there is an issue with how django migration handles dependency orders. In order to create your initial migration you will need to comment out the `STAR_RATINGS_RATING_MODEL` setting and run `makemigrations`. After this initial migration you will be able to add the setting back in and run `migrate` and `makemigrations` without issue.

CHAPTER 11

Changing the pk type (Requires django >= 1.10)

One use case for changing the rating model would be to change the pk type of the related object. By default we assume the pk of the rated object will be a positive integer field which is fine for most uses, if this isn't though you will need to override the object_id field on the rating model as well as set STAR_RATINGS_OBJECT_ID_PATTERN to a reasonable value for your new pk field. As of django 1.10 you can now hide fields form parent abstract models, so to change the object_id``to a ``CharField you can do something like:

```
class MyRating(AbstractBaseRating):
    object_id = models.CharField(max_length=10)
```

And add the setting to the setting file:

```
./settings.py

...
STAR_RATINGS_OBJECT_ID_PATTERN = '[a-z0-9]{32}'
...
```

CHAPTER 12

Events

Some events are dispatched from the javascript when an object is raised. Each event that ias dispatched has a `details` property that contains information about the object and the rating.

12.1 `rate-success`

Dispatched after the user has rated an object and the display has been updated.

The event details contains

```
{
    sender: ... // The star DOM object that was clicked
    rating: {
        average: ... // Float giving the updated average of the rating
        count: ... // Integer giving the total number of ratings
        percentage: ... // Float giving the percentage rating
        total: ... // Integer giving the sum of all ratings
        user_rating: ... // Integer giving the rating by the user
}
```

12.2 `rate-failed`

Dispatched after the user has rated an object but the server responds with an error.

The event details contains

```
{
    sender: ... // The star DOM object that was clicked
    error: ... // String giving the error message from the server
}
```

12.3 Running tests

To run the test use:

```
$> ./runtests.py
```

12.4 Releasing

Travis is setup to push releases to pypi automatically on tags, to do a release:

1. Up version number.
2. Update release notes.
3. Push dev.
4. Merge develop into master.
5. Tag with new version number.
6. Push tags.

12.5 Bugs and feature requests

> https://github.com/wildfish/django-star-ratings/issues

Created by Wildfish and contributors.