How to set the default of a JSONField to empty list in Django and django-jsonfield?

Asked 6 years, 7 months ago Modified 3 years ago Viewed 38k times

57

Question

What is the best way to set a `JSONField` to have the default value of a new list in `django`?

Context

There is a model where one of the fields is a list of items. In the case where there are no items set, the model should have an empty list.

Current solution

```
from django.models import Model

class MyModel(Model):
    the_list_field = JSONField(default=[])
```

Is this the best way to do it? Should it be switched to use `list` instead?

Thanks!

python django django-models reference

Share Improve this question Follow

asked Feb 15, 2017 at 2:47

Juan Carlos Coto
11.9k 22 63 102

Is there any reason why you're not using Django's built-in jsonfield? – Rod Xavier Feb 15, 2017 at 2:55

@RodXavier MySQL < 5.7 and other legacy code uses them, so for consistency. – Juan Carlos Coto Feb 15, 2017 at 3:47

2 Answers

Sorted by: Highest score (default)

74

According to the Django documentation for JSONField you should indeed use `default=list` because using `default=[]` would create a mutable object that is shared between all instances of your field and could lead to some objects not having an empty list as a default.

Please note that this does not only apply for `django.contrib.postgres.fields.JSONField` but for all other kinds of objects and functions in Python in general.

Quote from the docs:

> If you give the field a default, ensure it's a callable such as list (for an empty default) or a callable that returns a list (such as a function). Incorrectly using default=[] creates a mutable default that is shared between all instances of

Share Improve this answer Follow

edited Sep 1, 2017 at 10:37

answered Mar 15, 2017 at 9:00

jnns
5,168 4 47 74

51

`list` and `dict` are callable, while `[]` and `{}` are not (you can't do `[]()`). So:

- Use **`JSONField(default=list)`** over `JSONField(default=[])`
- Use **`JSONField(default=dict)`** over `JSONField(default={})`

If you want to instantiate with some data you can do the following:

```
def jsonfield_default_value():  # This is a callable
    return [0, 0]  # Any serializable Python obj, e.g. `["A", "B"]` or
`{"price": 0}`

class MyModel(Model):
    the_list_field = JSONField(default=jsonfield_default_value)
```

Share Improve this answer Follow

edited Sep 8, 2020 at 15:00

answered Nov 15, 2019 at 14:11

Nitin Nain
5,133 1 37 51