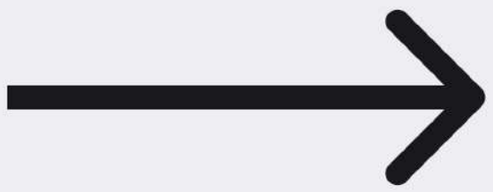
Important

Model Utilities & Mixins In Django



django-model-utils package provides some important model utilities & mixins that is very helpful while working with the django models

1. Install the package

We need to install **django-model-utils** package by just hitting this command in terminal

```
pip install django-model-utils
```

Important Fields

comes with django-model-utils

1. StatusField

A simple convenience for giving a model a set of “states.” StatusField is a CharField subclass that expects to find a class attribute called STATUS on its model

models.py

```python
from model_utils.fields import StatusField
from model_utils import Choices

class Article(models.Model):

    STATUS = Choices('draft', 'published')

    status = StatusField()
```

2. MonitorField

A DateTimeField subclass that monitors another field on the model, and updates itself to the current date-time whenever the monitored field changes:

models.py

```python
from model_utils.fields import MonitorField, StatusField

class Article(models.Model):
    STATUS = Choices('draft', 'published')

    status = StatusField()
    status_changed = MonitorField(monitor='status')
```

If a list is passed to the when parameter, the field will only update when it matches one of the specified values:

models.py

```python
from model_utils.fields import MonitorField, StatusField

class Article(models.Model):
    STATUS = Choices('draft', 'published')

    status = StatusField()
    published_at = MonitorField(monitor='status', when=['published'])
```

3. SplitField

A TextField subclass that automatically pulls an excerpt out of its content (based on a “split here” marker or a default number of initial paragraphs) and stores both its content and excerpt values in the database.

models.py

```python
from django.db import models
from model_utils.fields import SplitField

class Article(models.Model):
    title = models.CharField(max_length=100)
    body = SplitField()
```

Accessing a SplitField on a model

```
>>> a = Article.objects.all()[0]
>>> a.body.content
u'some text\n\n<!-- split -->\n\nmore text'
>>> a.body.excerpt
u'some text\n'
>>> unicode(a.body)
u'some text\n\n<!-- split -->\n\nmore text'
```

4. UUIDField

A UUIDField subclass that provides an UUID field. You can add this field to any model definition.

models.py

```python
from django.db import models
from model_utils.fields import UUIDField

class MyAppModel(models.Model):
    uuid = UUIDField(primary_key=True, version=4, editable=False)
```

With the param primary_key you can set if this field is the primary key for the model, default is True.

Param version is an integer that set default UUID version. Versions 1,3,4 and 5 are supported, default is 4.

If editable is set to false the field will not be displayed in the admin or any other ModelForm, default is False.

5. UrlsafeTokenField

A CharField subclass that provides random token generating using python's secrets.token_urlsafe as default value.

models.py

```python
from django.db import models
from model_utils.fields import UrlsafeTokenField


class MyAppModel(models.Model):
    uuid = UrlsafeTokenField(editable=False, max_length=128)
```

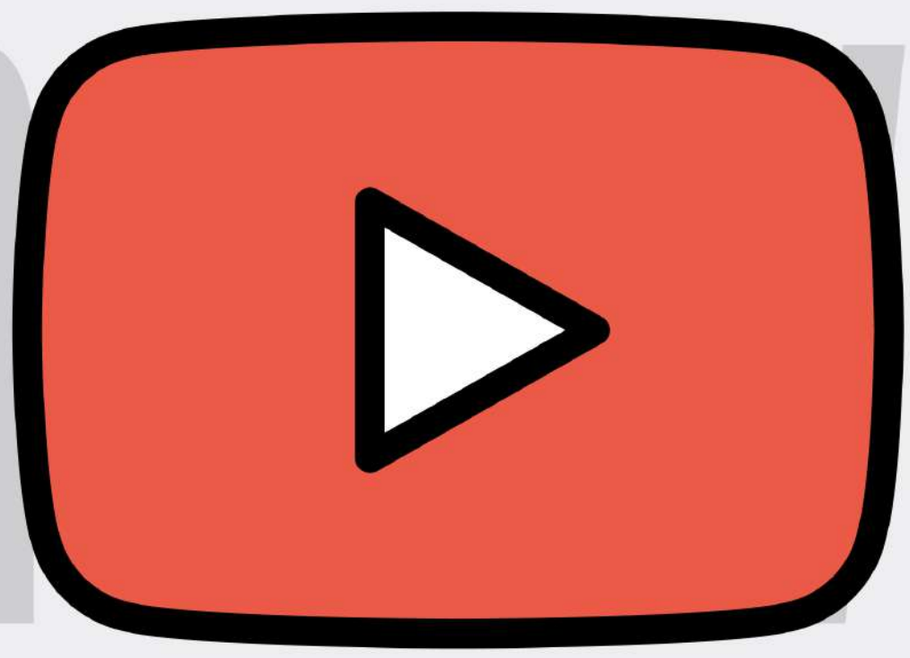
You can provide your custom token generator using the factory argument

models.py

```python
import uuid

from django.db import models
from model_utils.fields import UrlsafeTokenField


def _token_factory(max_length):
    return uuid.uuid4().hex


class MyAppModel(models.Model):
    uuid = UrlsafeTokenField(max_length=32, factory=_token_factory)
```

For more django related content

Subscribe To My Youtube Channel



Link in bio

Did You Find This Helpful?

Follow for more