



MOST POPULAR FIELD OPTIONS

- Null - if True, Django allows to store for the existing records values of null. To understand this, let's imagine we are adding a field to a existing table (in fact we are adding a column). The existing records in this column will be equal to null and we won't need to provide a default value.
- Blank - if is set to True, this field won't be required.
- Unique = if set to True, values of this field have to be unique and no duplicates are allowed.
- Primary_key = if set to True, this field will become the primary key of the model.
- Choices - allow storing values that are defined by us as defined choices. The first element in every tuple is the actual value stored in the database, the second element is the human-readable name
- Default - set a default value of the objects coming from a particular model
- Editable - if set to False, this field won't be visible in the admin and you can't edit it.
- verbose_name - change the human readable field name
- Validators - provide some extra validation for the particular field

MOST POPULAR FIELD TYPES

- BooleanField - checkbox, True or False
- CharField - for strings, requires max_length. Max size is 255 chars. For larger strings use TextField
- DateField - Field of date, if we import timezone, we can set default value for today's date by writing default=timezone.now
- DecimalField - for numbers as decimals. Specify max_digits and decimal_places:

Ensure that there are no more than 5 digits in total.

Decimal:

- EmailField - a CharField extended by a special validator - EmailValidator
- FileField - upload files, requires upload_to
- ImageField - According to the documentation : "Inherits all attributes and methods from FileField, but also validates that the uploaded object is a valid image". Requires pillow library (pip install pillow)
- IntegerField - for values from -2147483648 to 2147483647
- PositiveIntegerField - for values from 0 to 2147483647
- PositiveSmallIntegerField - for values from 0 to 32767
- SlugField - for slugs
- TextField - for strings above 255 chars
- URLField - it's a CharField for urls, but comes with URLValidator.
- UUIDField - field for storing universally unique identifiers

RELATIONSHIP FIELD TYPES

- ForeignKey - requires on_delete and Model
- ManyToManyField - requires Model
- OneToOneField - requires on_delete

EXTERNAL FIELD:

- PhoneNumberField

PhoneNumberField install:

pip install django-phonenumber-field

pip install phonenumbers

To installed apps in settings.py add:

'phonenumber_field',

In models.py do a import before start using:

from phonenumber_field.modelfields import PhoneNumberField

More info on phone number field : <https://github.com/stefanfoulis/django-phonenumbers-field>

Let's do this in practice!

In models.py we made some test settings:

```
from django.db import models
import uuid
from django.template.defaultfilters import slugify
from phonenumbers_field.modelfields import PhoneNumberField
from django.utils import timezone
from django.contrib.auth.models import User
# Create your models here.

my_choices = (
    ('one', 'number one'),
    ('two', 'number two')
)

class TestModel(models.Model):
    boolean = models.BooleanField(
        default=True, verbose_name="this is a boolean field")
    char = models.CharField(verbose_name="New name",
                           max_length=220, unique=True, help_text="added help text")
    date = models.DateField(
        default=timezone.now)
    decimal = models.DecimalField(max_digits=5, decimal_places=2)
    email = models.EmailField(max_length=200)
    file = models.FileField(upload_to='uploads', blank=True)
    image = models.ImageField(
        upload_to='uploads', blank=True)
    integer = models.IntegerField()
    positive_small_int = models.PositiveIntegerField()
    slug = models.SlugField(blank=True)
    text = models.TextField()
    url = models.URLField(max_length=200)
    uuid1 = models.UUIDField(default=uuid.uuid4)
    uuid2 = models.UUIDField(
        default=uuid.uuid4, primary_key=True, editable=False)
    updated = models.DateTimeField(auto_now=True)
    created = models.DateTimeField(auto_now_add=True)
    date_and_time = models.DateTimeField()
    choice = models.CharField(max_length=10, choices=my_choices)

    phone_number = PhoneNumberField()

    new_field = models.ForeignKey(User, on_delete=models.CASCADE, null=True)

    def save(self, *args, **kwargs):
        self.slug = slugify(self.text[:30])
        super(TestModel, self).save(*args, **kwargs)
```

On the next page, we'll see how this will show up in the admin

In the Admin:

☒ This is a boolean field


New name:

charfield

added help text

Date:

2019-11-23

Today | 

Note: You are 1 hour ahead of server time.

Decimal:

10,00

Email:

info@pyplane.com

File:

Currently: uploads/3_copy.png ☐ Clear

Change: No file chosen

Image:

Currently: uploads/3_copy_RhZ7wFP.png ☐ Clear

Change: No file chosen

Integer:

10

Positive small int:

10

Slug:

discovering-field-types

Text:

discovering field types

Url:


Currently: <https://www.pyplane.com>


Change:

Uuid1:

0b9b6581-daf3-4bee-99d8-3cd1607abe4d


Date and time:

Date: 2019-11-23 Today | 

Time: 12:48:39 Now | 

Note: You are 1 hour ahead of server time.


Choice:

number one 

Phone number:

+48123456789

New field:

----- 



Some remarks:

- DateTimeField is not visible in the admin for updated and created. When setting **auto_now=True** or **auto_now_add=True** we automatically are setting **editable=False**
- After creating some objects we added to **new_field** we added **null=True**, this way we allowed Django storing values as Null in this column. If we wouldn't set **null=True**, we would have to provide a default value
- Now the **new_field** is empty for the existing records, but required for the new ones. If we set it to **blank=True** it will become optional.
- With the **verbose_name** option we can change the name of the particular field and we did so for the two first fields
- Fields **file** and **image** are set as **blank=True** so this way they became optional
- Field **uuid1** is visible, while **uuid2** isn't. Again it's because of **editable=False**. Also when setting **primary_key=True**, this field will become the primary key for the model.
- To the first field we added help text which appears below the input.