By **Vitor Freitas**

I'm a passionate software developer and researcher. I write about Python, Django and Web Development on a weekly basis. Read more.

TUTORIAL

How to Reset Migrations

Jul 26, 2016 4 minutes read comments views

(Picture: https://www.pexels.com/photo/sky-flying-animals-birds-1209/)

The Django migration system was developed and optmized to work with large number of migrations. Generally you shouldn't mind to keep a big amount of models

migrations in your code base. Even though sometimes it causes some undesired effects, like consuming much time while running the tests. But in scenarios like this you can easily disable the migrations (although there is no built-in option for that at the moment).

Anyway, if you want to perform a clean-up, I will present a few options in this tutorial.

Scenario 1:

The project is still in the development environment and you want to perform a full clean up. You don't mind throwing the whole database away.

1. Remove the all migrations files within your project

Go through each of your projects apps migration folder and remove everything inside, except the `__init__.py` file.

Or if you are using a unix-like OS you can run the following script (inside your project dir):

```
find . -path "*/migrations/*.py" -not -name "__init__.py" -delete
find . -path "*/migrations/*.pyc"  -delete
```

2. Drop the current database, or delete the `db.sqlite3` if it is your case.

3. Create the initial migrations and generate the database schema:

```
python manage.py makemigrations
python manage.py migrate
```

And you are good to go.

Scenario 2:

You want to clear all the migration history but you want to keep the existing database.

1. Make sure your models fits the current database schema

The easiest way to do it is trying to create new migrations:

```
python manage.py makemigrations
```

If there are any pending migration, apply them first.

If you see the message:

```
No changes detected
```

You are good to go.

2. Clear the migration history for each app

Now you will need to clear the migration history app by app.

First run the `showmigrations` command so we can keep track of what is going on:

```
$ python manage.py showmigrations
```

Result:

```
admin
 [X] 0001_initial
 [X] 0002_logentry_remove_auto_add
auth
 [X] 0001_initial
 [X] 0002_alter_permission_name_max_length
 [X] 0003_alter_user_email_max_length
 [X] 0004_alter_user_username_opts
 [X] 0005_alter_user_last_login_null
 [X] 0006_require_contenttypes_0002
 [X] 0007_alter_validators_add_error_messages
contenttypes
 [X] 0001_initial
 [X] 0002_remove_content_type_name
core
 [X] 0001_initial
 [X] 0002_remove_mymodel_i
 [X] 0003_mymodel_bio
sessions
 [X] 0001_initial
```

Clear the migration history (please note that **core** is the name of my app):

```
$ python manage.py migrate --fake core zero
```

The result will be something like this:

```
Operations to perform:
  Unapply all migrations: core
Running migrations:
  Rendering model states... DONE
  Unapplying core.0003_mymodel_bio... FAKED
  Unapplying core.0002_remove_mymodel_i... FAKED
  Unapplying core.0001_initial... FAKED
```

Now run the command `showmigrations` again:

```
$ python manage.py showmigrations
```

Result:

```
admin
 [X] 0001_initial
 [X] 0002_logentry_remove_auto_add
auth
 [X] 0001_initial
 [X] 0002_alter_permission_name_max_length
 [X] 0003_alter_user_email_max_length
 [X] 0004_alter_user_username_opts
 [X] 0005_alter_user_last_login_null
 [X] 0006_require_contenttypes_0002
 [X] 0007_alter_validators_add_error_messages
contenttypes
 [X] 0001_initial
 [X] 0002_remove_content_type_name
core
 [ ] 0001_initial
 [ ] 0002_remove_mymodel_i
 [ ] 0003_mymodel_bio
sessions
 [X] 0001_initial
```

You must do that for all the apps you want to reset the migration history.

3. Remove the actual migration files.

Go through each of your projects apps migration folder and remove everything inside, except for the `__init__.py` file.

Or if you are using a unix-like OS you can run the following script (inside your project dir):

```
find . -path "*/migrations/*.py" -not -name "__init__.py" -delete
```

```
find . -path "*/migrations/*.pyc"  -delete
```

PS: The example above will remove all the migrations file inside your project.

Run the `showmigrations` again:

```
$ python manage.py showmigrations
```

Result:

```
admin
 [X] 0001_initial
 [X] 0002_logentry_remove_auto_add
auth
 [X] 0001_initial
 [X] 0002_alter_permission_name_max_length
 [X] 0003_alter_user_email_max_length
 [X] 0004_alter_user_username_opts
 [X] 0005_alter_user_last_login_null
 [X] 0006_require_contenttypes_0002
 [X] 0007_alter_validators_add_error_messages
contenttypes
 [X] 0001_initial
 [X] 0002_remove_content_type_name
core
 (no migrations)
sessions
 [X] 0001_initial
```

4. Create the initial migrations

```
$ python manage.py makemigrations
```

Result:

```
Migrations for 'core':
  0001_initial.py:
```

```
    - Create model MyModel
```

5. Fake the initial migration

In this case you won't be able to apply the initial migration because the database table already exists. What we want to do is to fake this migration instead:

```
$ python manage.py migrate --fake-initial
```

Result:

```
Operations to perform:
  Apply all migrations: admin, core, contenttypes, auth, sessions
Running migrations:
  Rendering model states... DONE
  Applying core.0001_initial... FAKED
```

Run `showmigrations` again:

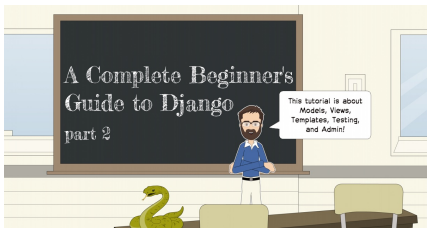
```
admin
 [X] 0001_initial
 [X] 0002_logentry_remove_auto_add
auth
 [X] 0001_initial
 [X] 0002_alter_permission_name_max_length
 [X] 0003_alter_user_email_max_length
 [X] 0004_alter_user_username_opts
 [X] 0005_alter_user_last_login_null
 [X] 0006_require_contenttypes_0002
 [X] 0007_alter_validators_add_error_messages
contenttypes
 [X] 0001_initial
 [X] 0002_remove_content_type_name
core
 [X] 0001_initial
```

```
sessions
 [X] 0001_initial
```

And we are all set up :-)

Related Posts

[Django Tips #22 Designing Better Models]



[A Complete Beginner's Guide to Django - Part 2]

[Django Tips #17 Using QuerySet Latest & Earliest Methods]

django models migration

Share this post

Subscribe to our Mailing List

Receive updates from the Blog!

Your email address

SUBSCRIBE

Popular Posts

How to Extend Django User Model

How to Setup a SSL Certificate on Nginx for a Django Application

How to Deploy a Django Application to Digital Ocean