Django Python web framework

CONCEPTS

manage.py is django-admin pinned to your project's settings; inside a project use manage.py.

models.py edit models
↓ `makemigrations`
Migration file in migrations/
↓ `migrate`
Database schema updated
↓ `runserver`
Dev server localhost:8000

A project holds settings and URLs; it is made of reusable apps, each with its own models, views and migrations.

PROJECT

`django-admin startproject <name>`
Create a project

`python manage.py startapp <app>`
Create an app in the project

`python manage.py check`
Run system checks

`python manage.py check --deploy`
Check production settings

Generated layout

Path	Purpose
`manage.py`	project command runner
`<name>/settings.py`	configuration
`<name>/urls.py`	root URL routes
`<app>/models.py`	database models
`<app>/views.py`	request handlers
`<app>/migrations/`	schema history

RUN

`python manage.py runserver`
Start the dev server

`python manage.py runserver <port>`
Use another port

`python manage.py runserver 0.0.0.0:8000`
Listen on all interfaces

`python manage.py shell`
Python shell with Django set up

RUN (cont.)

`python manage.py dbshell`
Open the database client

`python manage.py createsuperuser`
Create an admin user

`python manage.py changepassword <user>`
Change a user's password

MIGRATIONS

`python manage.py makemigrations`
Create migrations from model changes

`python manage.py migrate`
Apply pending migrations

`python manage.py makemigrations <app>`
Limit to one app

`python manage.py showmigrations`
List migrations and applied state

`python manage.py makemigrations --dry-run`
Preview without writing files

`python manage.py makemigrations --check`
Fail if migrations are missing

`python manage.py migrate <app> <migration>`
Migrate to a given point

TESTS

`python manage.py test`
Run all tests

`python manage.py test <app>.tests.<Class>.<method>`
Run one test

STATIC & I18N

`python manage.py collectstatic`
Gather static files for deployment

SHELL QUERIES

`Model.objects.filter(<field>=<value>)`
Rows matching a condition

`Model.objects.get(pk=<id>)`
One row; raises if missing or many