#022 docker-compose manage app

Introduction

this is part 22 from the journey it's a long journey(360 day) so go please check previous parts , and if you need to walk in the journey with me please make sure to follow because I may post more than once in 1 Day but surely I will post daily at least one 😍.

And I will cover lot of tools as we move on.

---

useful commands

this is a contd... for last part , I will talk about some useful commands that you may use on daily basis with docker-compose.

first let's stop all containers with

```
docker container stop $(docker container ls -a -q)
```

```
(base) in ~
> docker ps -a
CONTAINER ID        IMAGE                        COMMAND                  CREATED             STATUS                      PORTS                    NAMES
d44f8b556385        app_021_local_django         "/entrypoint /start"     24 hours ago        Exited (255) 9 hours ago    0.0.0.0:8000->8000/tcp   django
a1b07ee62af8        app_021_production_postgres  "docker-entrypoint.s…"   24 hours ago        Exited (255) 9 hours ago    5432/tcp                 postgres
(base) in ~
>
```

now let's say we need to run only PostgreSQL without Django image , we can do this by

```
docker-compose -f local.yml up postgres
```

postgres here is the name of the service.

```
> docker-compose -f local.yml up postgres
postgres is up-to-date
Attaching to postgres
postgres    | The files belonging to this database system will be owned by user "postgres".
postgres    | This user must also own the server process.
postgres    |
postgres    | The database cluster will be initialized with locale "en_US.utf8".
postgres    | The default database encoding has accordingly been set to "UTF8".
postgres    | The default text search configuration will be set to "english".
postgres    |
postgres    | Data page checksums are disabled.
postgres    |
postgres    | fixing permissions on existing directory /var/lib/postgresql/data ... ok
postgres    | creating subdirectories ... ok
postgres    | selecting default max_connections ... 100
postgres    | selecting default shared_buffers ... 128MB
postgres    | selecting dynamic shared memory implementation ... posix
postgres    | creating configuration files ... ok
postgres    | running bootstrap script ... ok
postgres    | performing post-bootstrap initialization ... ok
postgres    | syncing data to disk ... ok
postgres    |
postgres    | Success. You can now start the database server using:
postgres    |
```

We can see that PostgreSQL run without Django because in our local.yml it doesn't depend on Django.

In other hand in local.yml we have Django depends on postgres.

```yaml
local.yml
version: '3'

volumes:
  local_postgres_data: {}
  local_postgres_data_backups: {}

services:
  django:
    build:
      context: .
      dockerfile: ./compose/local/django/Dockerfile
    image: app_021_local_django
    container_name: django
    depends_on:
      - postgres
    volumes:
      - .:/app
    env_file:
      - ./.envs/.local/.django
      - ./.envs/.local/.postgres
    ports:
      - "8000:8000"
    command: /start

  postgres:
    build:
      context: .
      dockerfile: ./compose/production/postgres/Dockerfile
    image: app_021_production_postgres
    container_name: postgres
    volumes:
      - local_postgres_data:/var/lib/postgresql/data
- 759 bytes local.yml  yaml
```

so when I try to run Django postgres will automatically start also.

```
docker-compose -f local.yml up django
```

```
(base) (master)in ~/Documents/DevOpsJourney/app_021
> docker-compose -f local.yml up django
Starting postgres ... done
Starting django   ... done
Attaching to django
django       | PostgreSQL is available
```

So in case in our development Database got stuck or whatever we can restart database only using

```
docker-compose -f local.yml restart postgres
```

and this will make it restart.

Exec

In case you need to interact with docker container you can use

```
docker-compose exec sh
```

```
postgres    | 2020-06-23 23:28:34.383 UTC [1] LOG:  background worker "l
ogical replication launcher" (PID 31) exited with exit code 1
postgres    | 2020-06-23 23:28:34.383 UTC [26] LOG:  shutting down
postgres    | 2020-06-23 23:28:34.583 UTC [1] LOG:  database system is s
hut down
postgres    | 2020-06-23 23:28:42.043 UTC [1] LOG:  listening on IPv4 ad
dress "0.0.0.0", port 5432
postgres    | 2020-06-23 23:28:42.043 UTC [1] LOG:  listening on IPv6 ad
dress "::", port 5432
postgres    | 2020-06-23 23:28:42.152 UTC [1] LOG:  listening on Unix so
cket "/var/run/postgresql/.s.PGSQL.5432"
postgres    | 2020-06-23 23:28:42.334 UTC [24] LOG:  database system was
 shut down at 2020-06-23 23:28:34 UTC
postgres    | 2020-06-23 23:28:42.366 UTC [1] LOG:  database system is r
eady to accept connections
django      | PostgreSQL is available
django      | Operations to perform:
django      |   Apply all migrations: account, admin, auth, contenttypes
, sessions, sites, socialaccount, users
django      | Running migrations:
django      |   No migrations to apply.
django      |  * Running on http://0.0.0.0:8000/ (Press CTRL+C to quit)
django      |  * Restarting with stat
django      | Performing system checks...
django      |
django      | System check identified no issues (0 silenced).
django      |
django      | Django version 3.0.5, using settings 'config.settings.loca
l'
django      | Development server is running at http://0.0.0.0:8000/
django      | Using the Werkzeug debugger (http://werkzeug.pocoo.org/)
django      | Quit the server with CONTROL-C.
django      |  * Debugger is active!
django      |  * Debugger PIN: 271-812-314
```

```
(base) (master)in ~/Documents/DevOpsJourney/app_021
> docker-compose -f local.yml exec django bash
root@d44f8b556385:/app# ls
CONTRIBUTORS.txt  config     merge_production_dotenvs_in_dotenv.py
LICENSE           docs       production.yml
README.rst        local.yml  pytest.ini
app_021           locale     requirements
compose           manage.py  setup.cfg
root@d44f8b556385:/app#
```

In my case in Django sometimes I need to used python shell to create Objects and feed them to Database.
I can access python shell in same way

```
docker-compose exec python
```

```
(base) (master)in ~/Documents/DevOpsJourney/app_021
> docker-compose -f local.yml exec django python
Python 3.8.3 (default, Jun  9 2020, 17:49:41)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

actually you can do what ever you want with exec like run a script inside the folder.
Let's take example you have a script that load fake data to database.