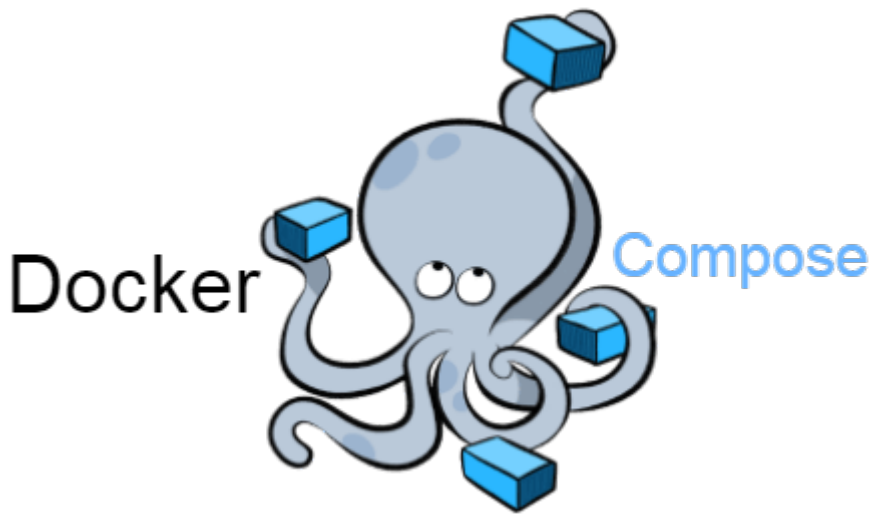


Docker Compose



Docker Compose es una utilidad de Docker que nos ayuda a la hora de crear múltiples contenedores Docker al mismo tiempo. A diferencia de Docker Cli este no hay que crearlo directamente en la terminal, sino que se crea un archivo .yml que es más fácil de ser modificado y configurado que en la propia terminal. Una vez creado solo se tiene que ejecutar un comando por terminal que comprobará el archivo e instalara y configurara lo que le hemos descrito en el mismo.

Creación de WordPress por Docker Cli

Primero se crea el Docker de la base de datos.

```
docker create \
  --name=mariadb \
  --net=lsio \
  -e PUID=1000 \
  -e PGID=1000 \
  -e MYSQL_ROOT_PASSWORD=mariadbpassword \
  -e TZ=Europe/London \
  -e MYSQL_DATABASE=WP_database \
  -e MYSQL_USER=WP_dbuser \
  -e MYSQL_PASSWORD=WP_dbpassword \
  -v /home/aptalca/appdata/mariadb:/config \
  --restart unless-stopped \
  lscr.io/linuxserver/mariadb
```

Y de la misma forma que la base de datos tendremos que crear el contenedor donde irá instalado la aplicación de WordPress.

```
docker create \
  --name=swag \
  --cap-add=NET_ADMIN \
  --net=lsio \
  -e PUID=1000 \
  -e PGID=1000 \
  -e TZ=Europe/London \
  -e URL=linuxserver-test.com \
  -e SUBDOMAINS=wildcard \
  -e VALIDATION=dns \
  -e DNSPLUGIN=cloudflare \
  -p 443:443 \
  -p 80:80 \
  -v /home/aptalca/appdata/swag:/config \
  --restart unless-stopped \
  lscr.io/linuxserver/swag
```

Una vez creadas los dos contenedores los tendremos que poner en la misma red de docker para que se puedan comunicar. Y por último dentro del segundo contenedor creado tendremos que instalar Wordpress:

```
wget https://wordpress.org/latest.tar.gz
tar xvf latest.tar.gz -C /home/aptalca/appdata/swag/www/
rm latest.tar.gz
```

Creado con Docker Compose

```
---
version: "2.1"
services:
  mariadb:
    image: lscr.io/linuxserver/mariadb
    container_name: mariadb
    environment:
      - PUID=1000
      - PGID=1000
```

- MYSQL_ROOT_PASSWORD=mariadbpassword
- TZ=Europe/London
- MYSQL_DATABASE=WP_database
- MYSQL_USER=WP_dbuser
- MYSQL_PASSWORD=WP_dbpassword

volumes:

- /home/aptalca/appdata/mariadb: /config

restart: unless-stopped

swag:

image: lscr.io/linuxserver/swag

container_name: swag

cap_add:

- NET_ADMIN

environment:

- PUID=1000
- PGID=1000
- TZ=Europe/London
- URL=linuxserver-test.com
- SUBDOMAINS=wildcard
- VALIDATION=dns
- DNSPLUGIN=cloudflare

volumes:

- /home/aptalca/appdata/swag: /config

ports:

- 443: 443
- 80: 80

depends_on:

- mariadb

restart: unless-stopped

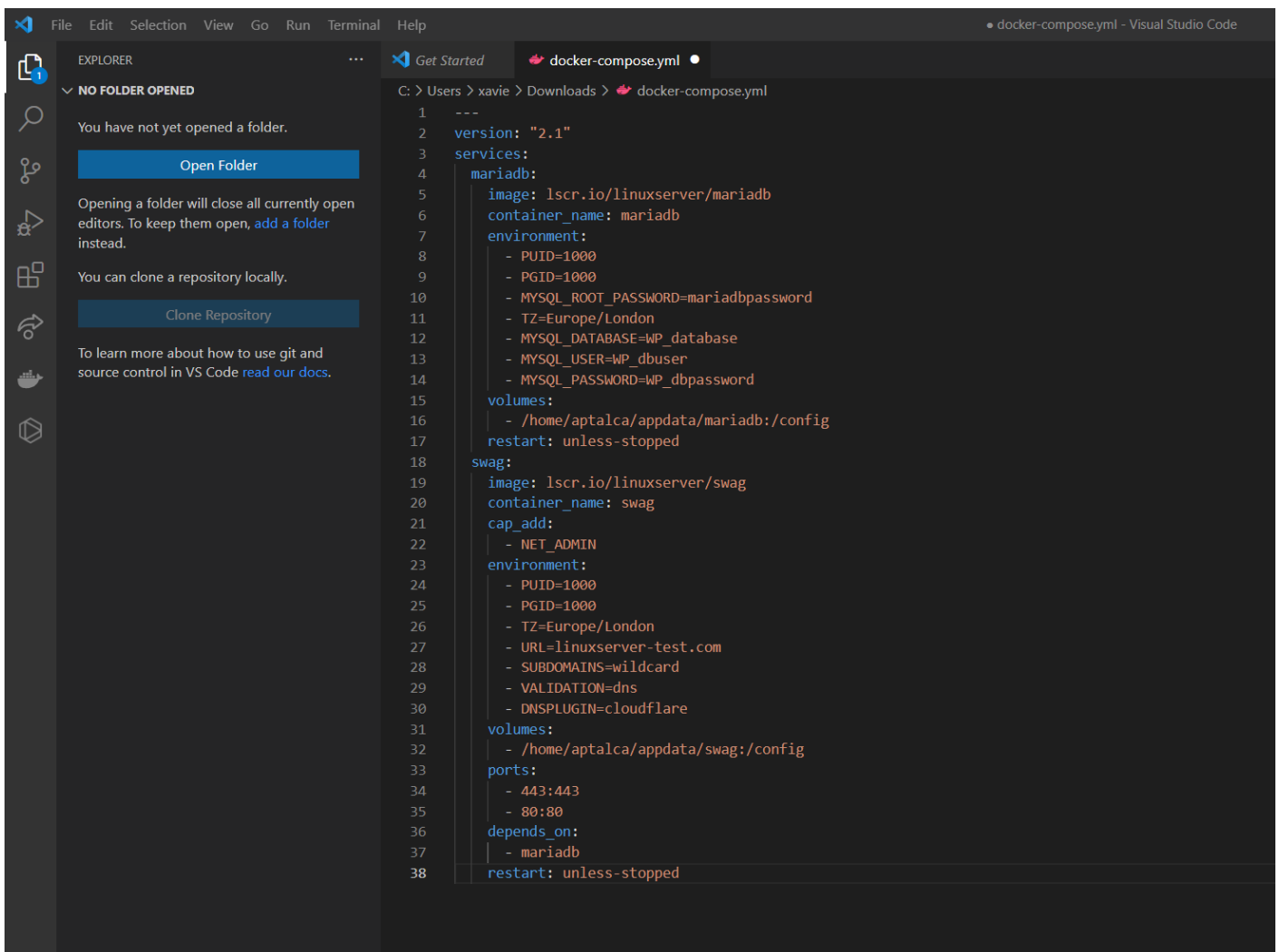
A diferencia de Docker Cli, en Docker Compose se puede hacer todo en el mismo archivo y después ejecutarlo en terminal

```
docker-compose up -d
```

Una vez el docker está creado, si se modifica el archivo se tendrá que crear los docker de nuevo y para eso está el siguiente comando

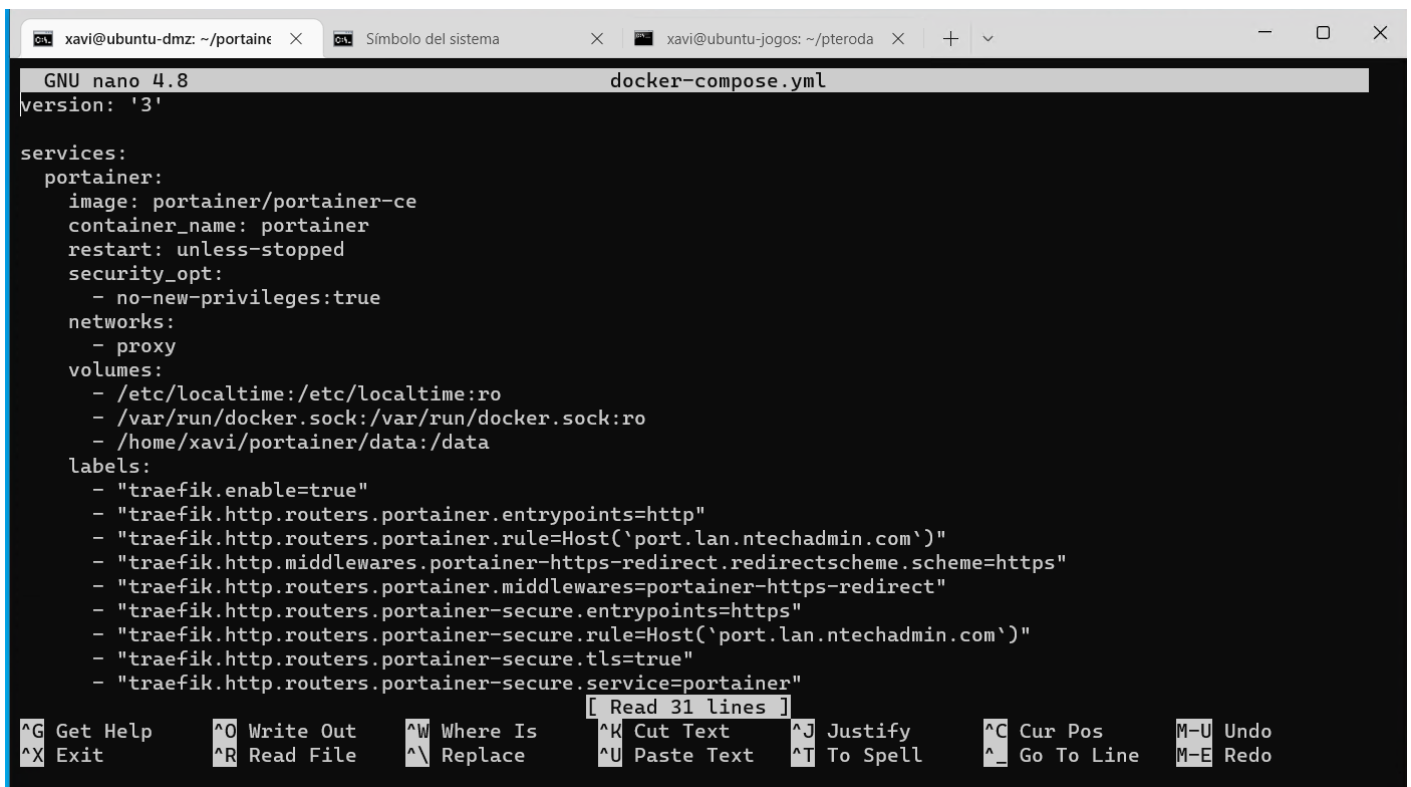
```
docker-compose up -d --force-recreate
```

Puede ser creado y configurado en un editor de texto y despues trasladarlo al archivo .yaml de nuestro server.



The screenshot shows the Visual Studio Code interface with a file named `docker-compose.yml` open. The Explorer sidebar on the left shows 'NO FOLDER OPENED' with buttons for 'Open Folder' and 'Clone Repository'. The main editor area displays the following YAML configuration:

```
1 ---
2 version: "2.1"
3 services:
4   mariadb:
5     image: lscr.io/linuxserver/mariadb
6     container_name: mariadb
7     environment:
8       - PUID=1000
9       - PGID=1000
10      - MYSQL_ROOT_PASSWORD=mariadbpassword
11      - TZ=Europe/London
12      - MYSQL_DATABASE=WP_database
13      - MYSQL_USER=WP_dbuser
14      - MYSQL_PASSWORD=WP_dbpassword
15     volumes:
16       - /home/aptalca/appdata/mariadb:/config
17     restart: unless-stopped
18   swag:
19     image: lscr.io/linuxserver/swag
20     container_name: swag
21     cap_add:
22       - NET_ADMIN
23     environment:
24       - PUID=1000
25       - PGID=1000
26       - TZ=Europe/London
27       - URL=linuxserver-test.com
28       - SUBDOMAINS=wildcard
29       - VALIDATION=dns
30       - DNSPLUGIN=cloudflare
31     volumes:
32       - /home/aptalca/appdata/swag:/config
33     ports:
34       - 443:443
35       - 80:80
36     depends_on:
37       - mariadb
38     restart: unless-stopped
```



The screenshot shows a terminal window with the nano text editor open, editing a file named `docker-compose.yml`. The terminal title bar shows the user is `xavi@ubuntu-dmz` in the directory `~/portainer`. The editor displays the following configuration:

```
GNU nano 4.8 docker-compose.yml
version: '3'

services:
  portainer:
    image: portainer/portainer-ce
    container_name: portainer
    restart: unless-stopped
    security_opt:
      - no-new-privileges:true
    networks:
      - proxy
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - /home/xavi/portainer/data:/data
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.portainer.entrypoints=http"
      - "traefik.http.routers.portainer.rule=Host(`port.lan.ntechadmin.com`)"
      - "traefik.http.middlewares.portainer-https-redirect.redirectscheme.scheme=https"
      - "traefik.http.routers.portainer.middlewares=portainer-https-redirect"
      - "traefik.http.routers.portainer-secure.entrypoints=https"
      - "traefik.http.routers.portainer-secure.rule=Host(`port.lan.ntechadmin.com`)"
      - "traefik.http.routers.portainer-secure.tls=true"
      - "traefik.http.routers.portainer-secure.service=portainer"
```

At the bottom of the terminal, there is a status bar with various keyboard shortcuts for nano, such as `^G Get Help`, `^O Write Out`, `^W Where Is`, `^K Cut Text`, `^J Justify`, `^C Cur Pos`, `M-U Undo`, `^X Exit`, `^R Read File`, `^_ Replace`, `^U Paste Text`, `^T To Spell`, `^_ Go To Line`, and `M-E Redo`.

Revision #2

Created 2 October 2022 11:49:02 by Xavi

Updated 3 October 2022 19:55:57 by Xavi