

Wordpress en docker para desarrollo.

Cómo usar docker para desarrollo o gestión de incidentes con Wordpress

A veces es necesario tener un entorno de desarrollo para realizar determinadas acciones, como puede ser una actualización conflictiva, un cambio de versión u otras que no se van a necesitar de un entorno que no sea el de producción.

[Docker+](#), es la elección perfecta, al menos para mí, acostumbrado a trabajar para todo con él, acompañandome todo mi desarrollo en mi portatil, como buen nómada que soy.

En el caso descrito habitual relativo a [WordPress](#), muy, muy, muy antiguos, que usan determinadas versiones de PHP, y que al intentar actualizarlas en producción, puede ser un completo desastre, y aunque pese a tener un sistema basado en **rsync** y **mysqldump** en la consola de comandos, la web que tenía que actualizar no tenía esa posibilidad de poder estar perdiendo 45 minutos en producción así que la dockerize.

“ Este tip es valido, para cualquier sitio web que queramos actualizar con un minimo de profesionalidad y seguridad

Dockerización de Wordpress

Habitualmente uso [Laravel Sail](#), pero sin embargo, he preferido hacer el tema de WordPress con una instalación independiente y usando Docker composer.

Los datos son específicos a mi entorno, pero si no eres de copiar y pegar y tratas de comprender cada paso, podrás trabajar donde quieras y como quieras que se adapte a tus necesidades. Esto supone diertas configuraciones de docker que pueden hacer que en otros escenarios o configuraciones no funcione lo que digo, sobre todo si la confioguración específica afecta al modo de red. Ver

[Docker Tips :: Algos sobre redes](#)

“ Se recomienda leer los [tips](#)

Creación de del entorno de trabajo

Directorio de trabajo

```
mkdir ~/Sites/myweb  
cd $_
```

Creación del docker-compose.yml

La creación la haremos en un principio ajustándose al escenario de funcionamiento, para luego progresivamente ir subiendo a la versión que queremos, para lidiar con todos los problemas de plugins, templates desactualizados.

Por tanto lo suyo no es copiar, por ejemplo las imágenes de docker que se indican, sino las que necesitamos obteniendo la información de las tags, en el hub de docker.

La imagen de [Wordpress](#) deberá ser la adecuada al escenario de nuestro sitio con problema en producción.

Versión de PHP, apache o nginx, fpm,...

Más adelante iremos subiendo a la que queremos.

“ Uso docker desktop por muchas razones, y lo recomiendo.

Editamos el fichero `editor ~/Sites/myweb/docker-compose.yml`

services:

db:

We use a mariadb image which supports both amd64 & arm64 architecture

image: mariadb

If you really want to use MySQL, uncomment the following line

image: mysql:8

command: '--default-authentication-plugin=mysql_native_password'

volumes:

- db_data:/var/lib/mysql

restart: always

environment:

- MYSQL_ROOT_PASSWORD=somewordpress

- MYSQL_DATABASE=la_misma_que_en_producción

- MYSQL_USER=el_mismo_que_en_producción

- MYSQL_PASSWORD=PasSW0rD_que_en_producción

ports:

- "3306:3306"

wordpress:

image: wordpress:php7.4-fpm-alpine

volumes:

- ./wp_data:/var/www/html

ports:

- 80:80

restart: always

environment:

- WORDPRESS_DB_HOST=db

- WORDPRESS_DB_USER=el_mismo_que_en_producción

- WORDPRESS_DB_PASSWORD=PasSW0rD_que_en_producción

- WORDPRESS_DB_NAME=la_misma_que_en_producción

volumes:

db_data:

wp_data:

Explicación

- **MySQL** lo montamos con un volumen, para hacer permanente sus datos.
- La instancia de **WordPress** lo mismo, poniendo el directorio de montaje en el directorio en el que vamos a clonar el sitio, apuntando al directorio `/var/www/html` del volumen.

Copia del contenido de producción

La mejor opción es `rsync` una herramienta **imprescindible** no sólo para los administradores de sistemas, sino para los webmaster o mantenedores de sitios web.

```
$ rsync -avzzz --progress -e "ssh" user_remoto@dominio_remoto.tld:/path/al/sitio/ ~/Sites/myweb/wp_data/
receiving file list ...
33791 files to consider
./
license.txt
    19915 100% 18.99MB/s 0:00:00 (xfer#1, to-check=33778/33791)
readme.html
    7402 100% 7.06MB/s 0:00:00 (xfer#2, to-check=33774/33791)
wp-config-sample.php
    3013 100% 2.87MB/s 0:00:00 (xfer#3, to-check=33766/33791)
wp-config.php
    3601 100% 3.43MB/s 0:00:00 (xfer#4, to-check=33765/33791)
...
...
wp-includes/widgets/class-wp-widget-text.php
    21342 100% 20.99kB/s 0:00:00 (xfer#2301, to-check=0/33791)
sent 479944 bytes received 860743 bytes 536274.80 bytes/sec
total size is 1363857930 speedup is 1017.28
```

Copia o dump de la base de datos de producción

Podemos usar nuestra herramienta preferida, como TablePlus, Navicat, phpMyAdmin... o el propio shell.

El caso es hacer una copia completa.

Me gusta usar el shell.

Como no tengo acceso mysql remoto, y no me apetece montar un túnel, hago el backup en remoto y luego lo bajo con `rsync`

Servidor remoto

```
mysqldump --no-tablespaces --opt -u <wp_user> -p <database> > ~/<database>.sql
Enter password:
```

Servidor local

Ahora traemos el fichero de dump

```
rsync -avzz --progress -e "ssh -p 2244" user_remoto@remoto.tld:/home/myuser/<database>.sql .
receiving file list ...
1 file to consider
<database>.sql
 130317359 100%  7.80MB/s   0:00:15 (xfer#1, to-check=0/1)
sent 25352 bytes  received 35045636 bytes  2004056.46 bytes/sec
total size is 130317359  speedup is 3.72
```

⚠ Atención al `.` final del comando que quiere decir `cópiame aquí`

Restaurar la copia

Obtener información del container

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
NAMES					
9b780079bb	wordpress:latest	"docker-entrypoint.s..."	40 minutes ago	Up 40 minutes	0.0.0.0:80->80/tcp
myweb-wordpress-1					
c591fdcae17a	mysql:8	"docker-entrypoint.s..."	40 minutes ago	Up 40 minutes	0.0.0.0:3306->3306/tcp, 33060/tcp
myweb-db-1					

Volcar el backup

Usaremos el **id** del contenedor de mysql `c591fdcae17a` para ejecutar un comando en él.

```
cat <database>.sql | docker exec -i c591fdcae17a /usr/bin/mysql -u root --password=somewordpress
<database>
mysql: [Warning] Using a password on the command line interface can be insecure.
```

En mi caso por vaquería y comodidad, uso TablePlus para muchas cosas, y tiene un importador. Pero siempre hay una vez para fallar, y esta vez, fue al importar una bd con 45Mb, (he importado en table plus bd de 6 GB sin problemas). Se terminaba siempre en una de las tablas. Fue hacerlo en el el shell, y funcionó sin problemas.

Levantamos la imagen

```
docker compose up -d
```

`docker compose` type unknown

Para probar que esta todo correcto, solo tendríamos que ir a nuestro navegador y solicitar localhost

Devolver a su sitio.

Tras actualziar y ajustar todo a las últimas versiones tanto de wordpress, como de PHP, Mysql o MariaDb, deberemos subirlo a producción.

Para devolver a su sitio la actualización completa es sencillo. Usamos la misma técnica de `rsync` + `mysqldump` pero al revés.

Mysqldump

“ Esta vez usaremos el **name** del container en lugar del **ID**

```
docker exec -i myweb-db-1 /usr/bin/mysqldump --opt -u root --password=somewordpress --databases  
<database> --skip-comments > dump-updated.sql
```

Subir los cambios a producción

```
~/Sites/myweb $ rsync -avzz --progress --delete -e "ssh -p 2244" ~/Sites/myweb/wp_data/  
user_remoto@dominio_remoto.tld:/path/al/sitio/  
~/Sites/myweb $ rsync -avzz --progress --delete -e "ssh -p 2244" ~/Sites/myweb/dump-updated.sql  
user_remoto@dominio_remoto.tld:/path/al/user/
```

Restaurar la base de datos en el servidor de producción

```
Mysql -u <user_de_mysql> -p <bd_produccion> < ~/dump-updated.sql
```

Tips.

- Editar `/etc/hosts` para que apunte a local (127.0.0.1) y evitar líos de redirecciones. En el caso de MacOSx, sería en `/private/etc/hosts`

```
127.0.0.1 myweb.com
```

- Usar un navegador permisivo o configurado para no usar https (safari, opera...). Si tenemos algun plugin que fuerce la redirección `http` a `https` deberías desactivarlo.
- Usar una página `info.php` para tener las cosas claras. Simplemente debe tener `<?php phpinfo();` y ponerla en el `public` de nuestra web.
- Restaurar el backup (mejor desde shell como siempre)
- Los datos de creación de mysql, user y password deben ser los mismos en el par, producción-desarrollo.
- El parámetro del fichero `wp-config.php` `define('DB_HOST', 'localhost');` debe estar definido a `'localhost'`.
- **editor** es el alias de mi editor preferido.
- `$` indica que estamos en una cuenta de usuario y su path y `#` es root
- `<variable>` indica que debe ser sustituido por los valores apropiados incluyendo en el reemplazo los `<>`
- Atención a la barra `/` al final de los parámetros de `rsync` porque no es lo mismo con ella que sin ella.

A trabajar con el tema.

Tips globales

- Una vez que ya hemos conseguido la actualización menor necesaria para poder pasar a una versión más alta de php sin problemas, simplemente debemos editar el fichero `docker-compose.yml` y solicitar la nueva versión de php, y reconstruir el volumen de Wordpress. (Borrar container y volumen wordpress).
- Siempre debemos trabajar inicialmente con las mismas versiones y software que como está en producción. Hay que recordar que existe, versiones de Mysql, MariaDb (no son iguales no... ojo a ese datos que más de un quebradero de cabeza os podéis llevar un

día), versiones de PHP, con FPM, mod, con Nginx, con Apache, etc.

- Si no te sientes cómodo con el shell, al menos elige un [hosting](#) que tenga una herramienta de backup como Jetbackup que te permite crear snapshots antes de liarte con cambios.

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de [Castris](#) Si necesitas soporte profesional puedes contratar con Castris [soporte profesional](#).

Revision #3

Created 5 April 2023 06:59:59 by Abkrim

Updated 3 March 2024 08:15:24 by Abkrim