

Dockeriza tu flujo de trabajo con WordPress

Miguel Useche



¡Hola soy Miguel!



- Desarrollador web de 🇨🇴 viviendo en 🇨🇴
- 15 años usando WordPress, fundador del primer Meetup de WP en Venezuela.
- Anteriormente: Profesor Universitario Y Mozilla Tech Speaker.
- Colaborador en varias comunidades:



Problemas comunes al desarrollar con WordPress

MySQL
phpMyAdmin

Servidor
Web

P D
H N
P S

Usuario y permisos de la BD

Descargar WordPress

Descargar plugins

Permisos de archivos

Descomprimir carpeta





Entorno de desarrollo

- PHP 7.4
- MySQL 8.0
- WordPress 5.7.2



Entorno de Producción

- PHP 7.2
- MySQL 5.7
- WordPress 5.7.2



Entorno de desarrollo

- PHP 7.4
- MySQL 8.0
- WordPress 5.5.1



Proyecto A

- PHP 7.4
- MySQL 5.7
- WordPress 4.9.x
- Apache



Proyecto B

- PHP 7.2
- MySQL 8.0
- WordPress 5.0
- LiteSpeed

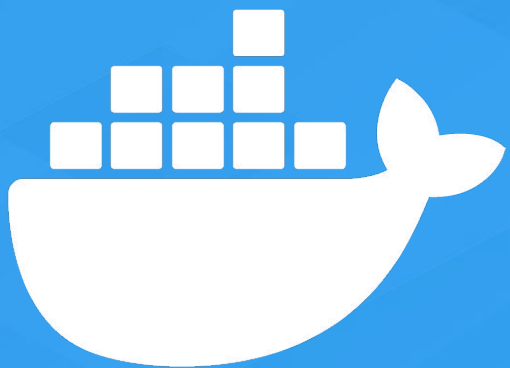


Proyecto C

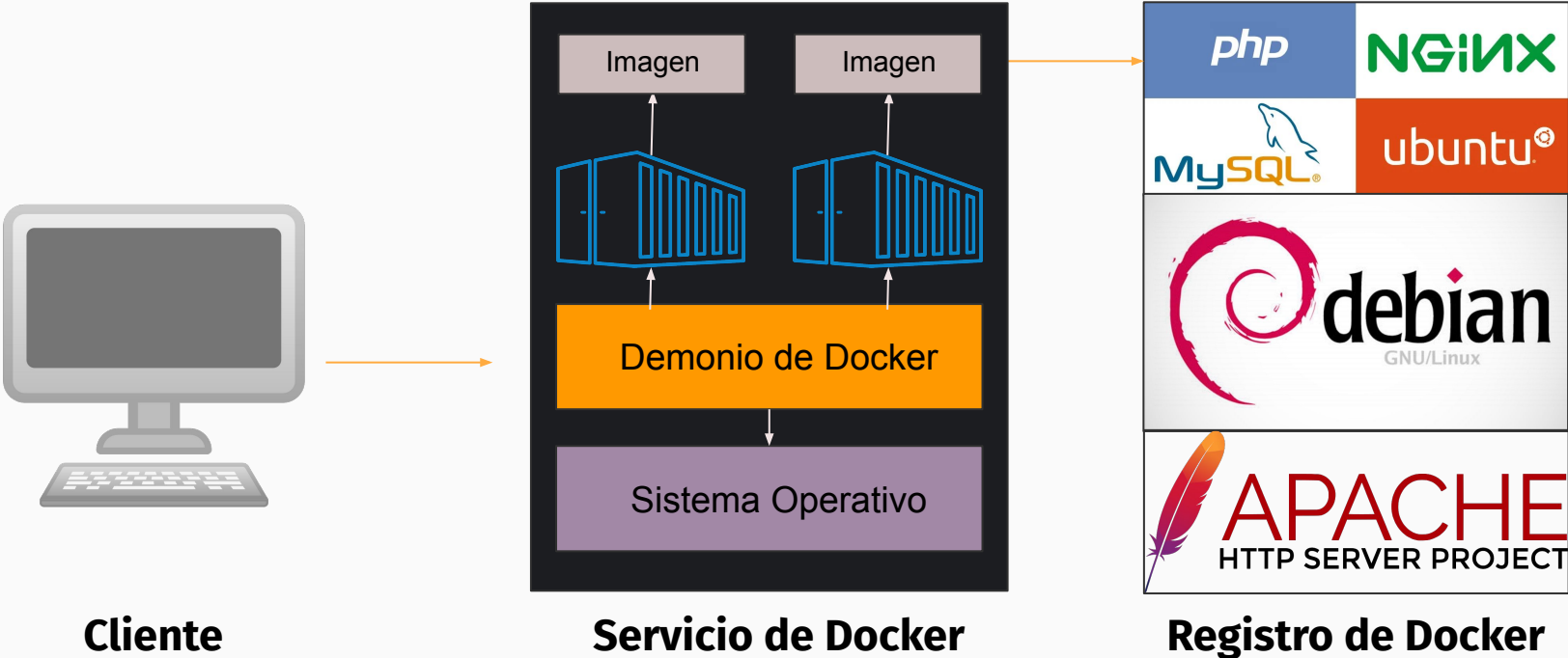
- PHP 7.3
- MySQL 5.5
- WordPress 5.5.1
- Nginx

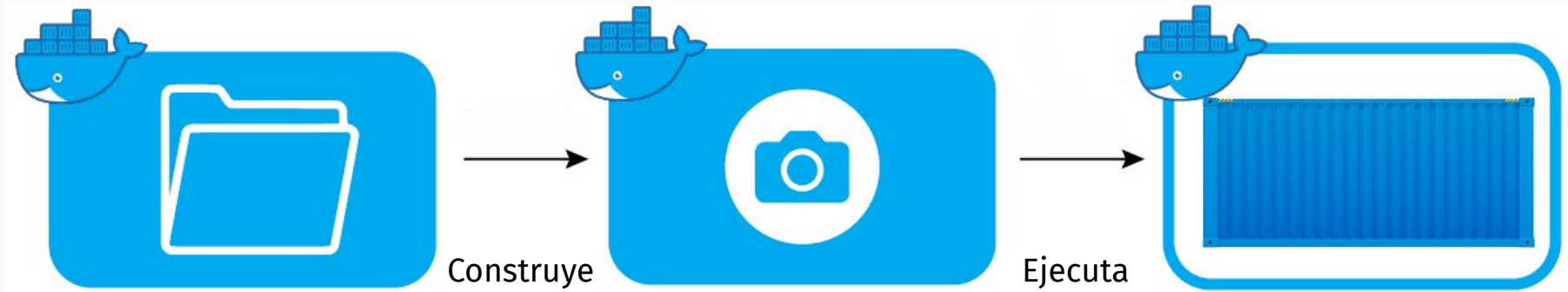
¿No debería existir una forma de estandarizar esto?





docker®





Dockerfile

Imagen

Contenedor

Crear una imagen:

```
docker build --tag nombre_de_mi_imagen:1.0 .
```


Ejecutar la imagen:

```
docker run --publish 80:8080 --detach --name bb nombre_de_mi_imagen:1.0
```



```
1 FROM php:7.0-apache
2
3 # install the PHP extensions we need
4 RUN set -ex; \
5     \
6     apt-get update; \
7     apt-get install -y \
8         libjpeg-dev \
9         libpng-dev \
10        ; \
11    rm -rf /var/lib/apt/lists/*; \
12    \
13    docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg-dir=/usr; \
14    docker-php-ext-install gd mysqli opcache
15 # TODO consider removing the *-dev deps and only keeping the necessary lib* packages
16
17 # set recommended PHP.ini settings
18 # see https://secure.php.net/manual/en/opcache.installation.php
19 RUN { \
20     echo 'opcache.memory_consumption=128'; \
21     echo 'opcache.interned_strings_buffer=8'; \
22     echo 'opcache.max_accelerated_files=4000'; \
23     echo 'opcache.revalidate_freq=2'; \
24     echo 'opcache.fast_shutdown=1'; \
25     echo 'opcache.enable_cli=1'; \
26 } > /usr/local/etc/php/conf.d/opcache-recommended.ini
27
28 RUN a2enmod rewrite expires
29
30 VOLUME /var/www/html
31
32 ENV WORDPRESS_VERSION 4.9.1
33 ENV WORDPRESS_SHA1 892d2c23b9d458ec3d44de59b753adb41012e903
34
35 RUN set -ex; \
36     curl -o wordpress.tar.gz -fSL "https://wordpress.org/wordpress-${WORDPRESS_VERSION}.tar.gz"; \
37     echo "WORDPRESS_SHA1 *wordpress.tar.gz" | shasum -c -; \
38 # upstream tarballs include ./wordpress/ so this gives us /usr/src/wordpress
39     tar -xzf wordpress.tar.gz -C /usr/src/; \
40     rm wordpress.tar.gz; \
41     chown -R www-data:www-data /usr/src/wordpress
42
43 COPY docker-entrypoint.sh /usr/local/bin/
44
45 ENTRYPOINT ["docker-entrypoint.sh"]
46 CMD ["apache2-foreground"]
```

Ejemplo de un Dockerfile para WordPress

A cartoon illustration of Homer Simpson from 'The Simpsons' running on a treadmill in a warehouse. He is wearing a blue tracksuit and red sneakers. He has a thoughtful expression, with his hand on his chin. A speech bubble next to him contains the text '¿Tengo que hacer todo eso para instalar WordPress?'. The background shows cardboard boxes and industrial equipment.

¿Tengo que hacer todo eso para instalar WordPress?

Global
HD

Descarga imágenes ya
preconfiguradas en:



¿Cómo se configura?

version: '3.3'

services:

db:

image: mysql:5.7

ports:

- 3306:3306

environment:

MYSQL_ROOT_PASSWORD: wordpress

MYSQL_DATABASE: wordpress

MYSQL_USER: wordpress

MYSQL_PASSWORD: wordpress

volumes:

- mysql_startup:/var/lib/mysql:delegated

wp:

image: wordpress-ssl:5.5.1

depends_on:

- db

volumes:

- ./wp-content:/var/www/html/wp-content:delegated

ports:

- 8000:80

- 4444:443

environment:

WORDPRESS_DEBUG: 1

WORDPRESS_DB_HOST: db:3306

WORDPRESS_DB_USER: wordpress

WORDPRESS_DB_PASSWORD: wordpress

volumes:

mysql_startup:



docker-machine

Aplicación para gestionar contenedores:

- Define los servicios que proveen los contenedores en formato YAML en **docker-compose.yml**
- Ahí puedes:
 - Configurar variables del entorno.
 - Define las rutas de los archivos.
 - Crear redes entre contenedores.


```
services:
  db:
    image: mysql:5.7
    ports:
      - 3306:3306
    environment:
      MYSQL_ROOT_PASSWORD: wordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress
    volumes:
      - mysql_base_datos:/var/lib/mysql:delegated
```



Definimos un contenedor con MySQL 5.7 donde:

- El puerto 3306 está abierto.
- El usuario root, el de WordPress y la contraseña es la palabra: **wordpress**.
- La base datos se monta en el volumen: **mysql_base_datos**



WORDPRESS

Definimos un contenedor con WordPress y nginx donde:

- WP es 5.5.1 y PHP 7.4
- Se monta la carpeta local de **wp-content** en el contenedor.
- Los puertos 8000 (HTTP) y 4444 (HTTPS) están abiertos.
- Se define la configuración de la base de datos y el modo de depuración.

services:

image: **wordpress-5.5.1-php7.4-fpm**

depends_on:

- **db**

volumes:

- **./wp-content:/var/www/html/wp-content**

ports:

- **8000:80**

- **4444:443**

environment:

WORDPRESS_DEBUG: **1**

WORDPRESS_DB_HOST: **db:3306**

WORDPRESS_DB_USER: **wordpress**

WORDPRESS_DB_PASSWORD: **wordpress**

La primera vez, o cuando cambiamos imágenes. Descargamos y construimos las imágenes con:

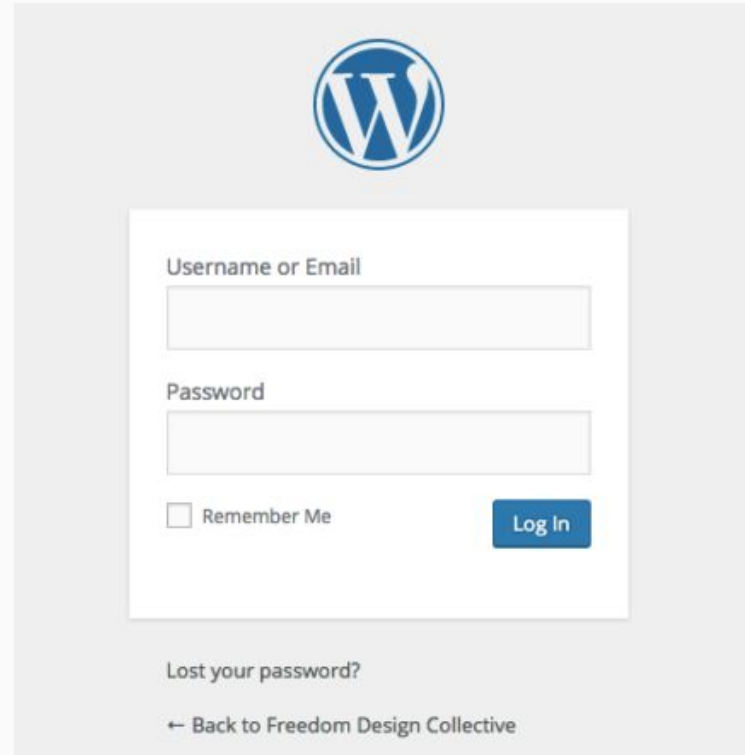
```
docker-compose build -f
```

Creamos los contenedores y levantamos los servicios con:

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

¡Listo!

Al entrar a **localhost:8080**, el puerto definido en docker-compose. Veremos el asistente de WP o nuestra página si la BD ya existe.

A screenshot of the WordPress login interface. At the top center is the WordPress logo (a blue 'W' in a circle). Below it is a white login box with a light gray border. Inside the box, there are two input fields: "Username or Email" and "Password". Below the "Password" field is a checkbox labeled "Remember Me". To the right of the checkbox is a blue "Log In" button. Below the login box, there is a link "Lost your password?" and a link "← Back to Freedom Design Collective".

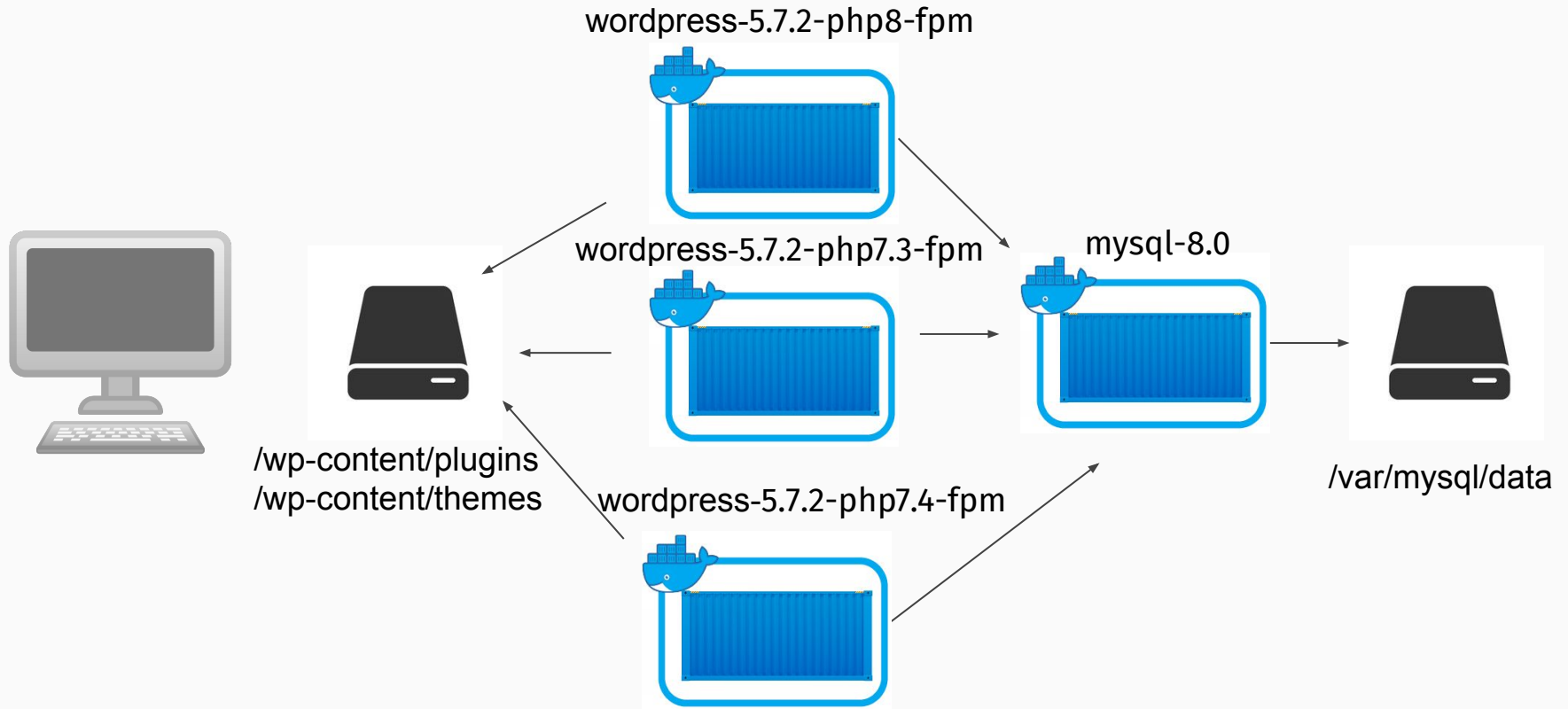
¿Por qué se ahorra tiempo?

- Solo necesitas el archivo ***docker-composer.yml*** y con un comando tienes todo levantando.
- Define la arquitectura una vez y se re-usa en otros proyectos.
- No lidias con configuraciones de servidor.
- Tienes el mismo entorno en desarrollo y producción.
- La ejecución es segura.
- No lidias con la repetición de archivos en tu equipo.

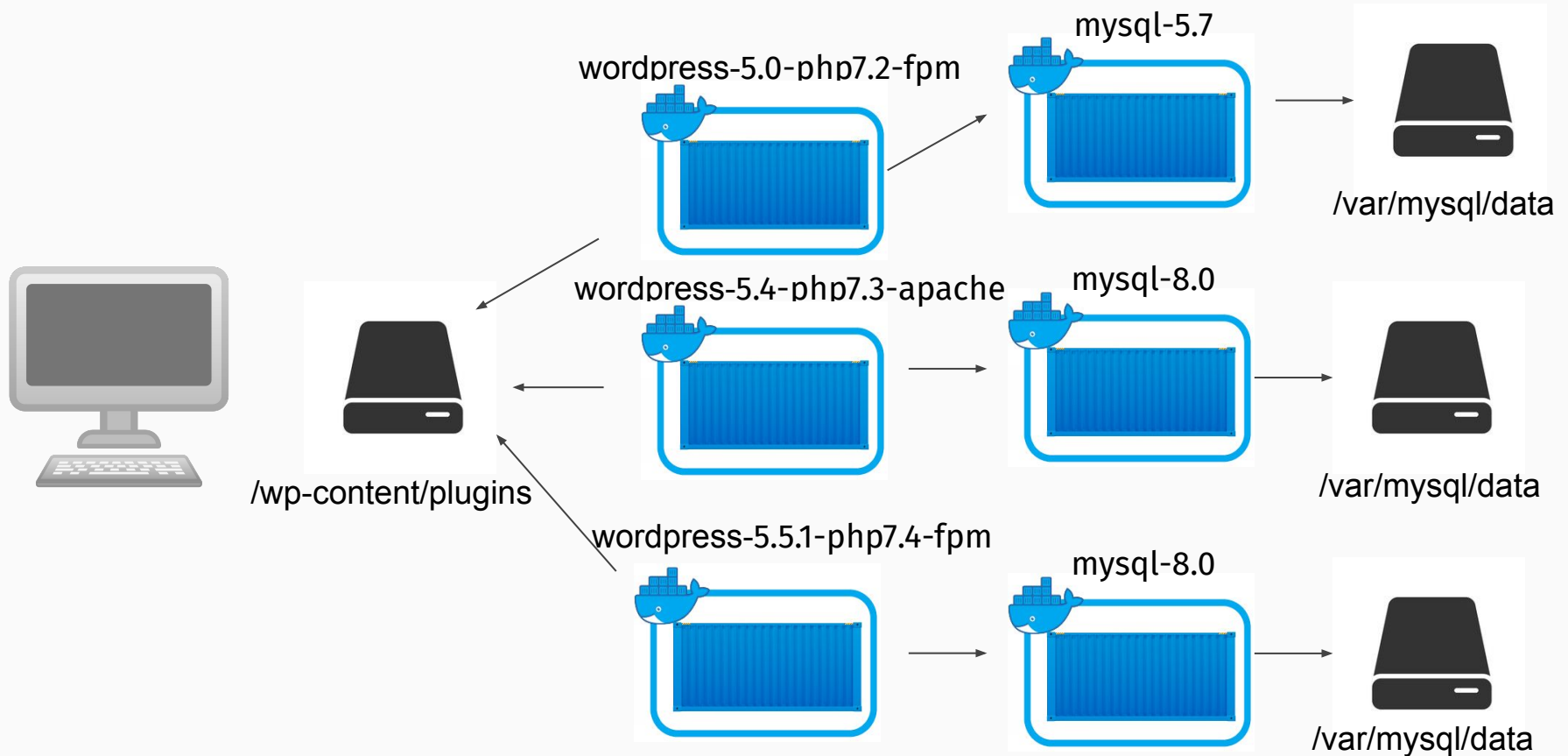
¿Cómo se hace el despliegue del sitio?

- Instalar docker en un VPS y hacer los mismos pasos que en desarrollo (puedes definir un **docker-compose.yml** para producción)
- Amazon, Heroku y servicios de la nube permiten crear instancias en la nube con docker.
- Con automatización de CI. Evitas estar usando FTP/SSH y scripts para enviar la información al servidor.

¿Qué cosas puedo hacer con Docker?



Probar en distintas versiones de PHP





¡Experimenta!



Gracias!!!

 @skatox

 migueluseche@skatox.com

 <https://skatox.com>

