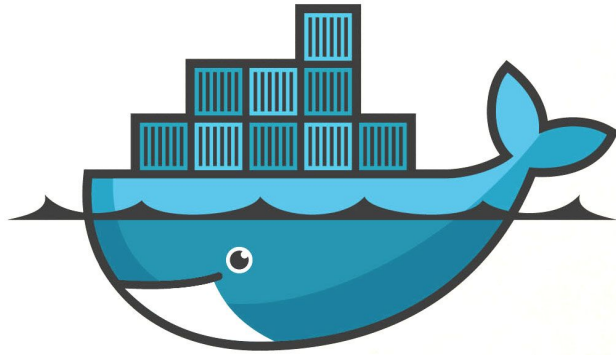


Imágenes



docker

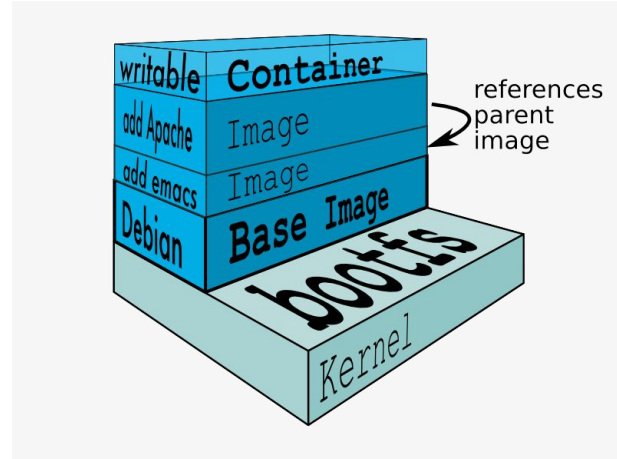
Repaso - *Imágenes*

Las imágenes son **plantillas** para crear contenedores.

A partir de una imagen se pueden crear **múltiples** contenedores.

Pueden tener **parámetros** a concretar al crear el contenedor.

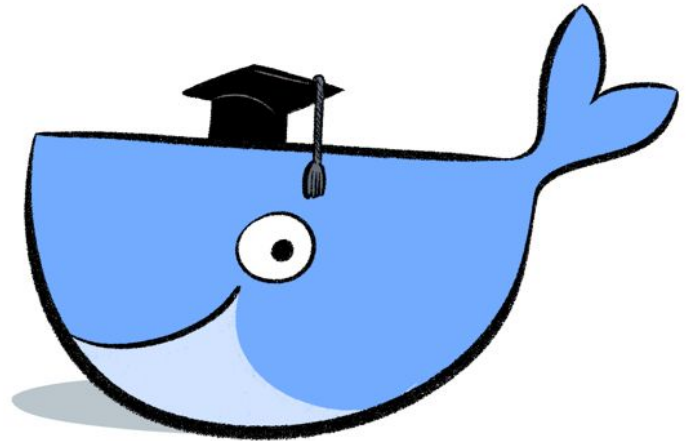
Se crean a partir de otras, añadiendo modificaciones en forma de **capas**.



Contenido

1. Comandos básicos de gestión de imágenes
2. Creación de imágenes
3. Contenedores e imágenes
4. Publicación de imágenes

Comandos básicos de gestión de imágenes



Comandos básicos de gestión de imágenes

1. Listado local
2. Búsqueda en el repositorio
3. Descarga
4. Ver historial / capas
5. Eliminar imagen local
6. Eliminar imágenes y contenedores parados



Listado local

docker images

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
wordpress	fpm-alpine	54b80d601342	5 days ago	290MB
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	fpm-alpine	7625aeffad8a	5 days ago	126MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	5 days ago	521MB
postgres	14-alpine	114818c12d10	6 days ago	211MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

Listado local - Filtro básico

docker images [REPOSITORIO[:TAG]]

→ docker images wordpress

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
wordpress	fpm-alpine	54b80d601342	5 days ago	290MB
wordpress	latest	ef25498ff6a9	5 days ago	605MB

→ docker images "*:latest"

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB



Búsqueda en el repositorio

docker search *TERM*

→ docker search wordpress

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
wordpress	The WordPress rich content management system...	4665	[OK]	
bitnami/wordpress	Bitnami Docker Image for WordPress	189		[OK]
bitnami/wordpress-nginx	Bitnami Docker Image for WordPress with NGINX	57		[OK]
tutum/wordpress	Out-of-the-box Wordpress docker image	41		
aveltens/wordpress-backup	Easily backup and restore your WordPress blo...	26		[OK]
arm32v7/wordpress	The WordPress rich content management system...	15		
centurylink/wordpress	Wordpress image with MySQL removed.	14		[OK]
appsvcoreg/wordpress-alpine-php	This is a WordPress Docker image which can ...	13		
wordpressdevelop/php	PHP images for the WordPress local developme...	9		
wordpressdevelop/cli	WP-CLI images for the WordPress local develo...	7		
arm64v8/wordpress	The WordPress rich content management system...	7		
dalareo/wordpress-ldap	Wordpress images with LDAP support automatic...	7		[OK]
ansibleplaybookbundle/wordpress-ha-apb	An APB which deploys Wordpress HA	5		[OK]
wordpressdevelop/phpunit	PHPUnit images for the WordPress local devel...	5		

Búsqueda en el repositorio

La opción **-f** se puede usar para filtrar según diferentes contextos.

```
→ docker search -f is-official=true wordpress
```

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
wordpress	The WordPress rich content management system...	4665	[OK]	

```
→ docker search -f stars=40 wordpress
```

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
wordpress	The WordPress rich content management system...	4665	[OK]	
bitnami/wordpress	Bitnami Docker Image for WordPress	189		[OK]
bitnami/wordpress-nginx	Bitnami Docker Image for WordPress with NGINX	57		[OK]
tutum/wordpress	Out-of-the-box Wordpress docker image	41		

Búsqueda en el repositorio

La opción **-limit** limita el número de resultados (25 por defecto).

```
→ docker search --limit 4 wordpress
```

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
wordpress	The WordPress rich content management system...	4665	[OK]	
bitnami/wordpress	Bitnami Docker Image for WordPress	189		[OK]
bitnami/wordpress-nginx	Bitnami Docker Image for WordPress with NGINX	57		[OK]
bitnami/wordpress-intel		0		



Descarga

docker pull *REPOSITORIO[:TAG]*

```
→ docker pull nginx  
Using default tag: latest  
latest: Pulling from library/nginx  
c229119241af: Already exists  
2215908dc0a2: Pull complete  
08c3cb2073f1: Pull complete  
18f38162c0ce: Pull complete  
10e2168f148a: Pull complete  
c4ffe9532b5f: Pull complete  
Digest: sha256:2275af0f20d71b29391  
Status: Downloaded newer image for  
docker.io/library/nginx:latest
```

```
→ docker pull nginx:alpine  
alpine: Pulling from library/nginx  
40e059520d19: Already exists  
f206cf0d6188: Pull complete  
065a4ca9176e: Pull complete  
67124ec378c3: Pull complete  
b17ba2c5bc9f: Pull complete  
fed8f5509a6a: Pull complete  
Digest: sha256:44e208ac2000daeff77c27a409d1794d6bbdf52067de627c2da13e36c7d59582  
Status: Downloaded newer image for nginx:alpine  
docker.io/library/nginx:alpine
```

Descarga

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
wordpress	fpm-alpine	54b80d601342	5 days ago	290MB
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	fpm-alpine	7625aeffad8a	5 days ago	126MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	5 days ago	521MB
nginx	alpine	53722defe627	6 days ago	23.4MB
nginx	latest	12766a6745ee	6 days ago	142MB
postgres	14-alpine	114818c12d10	6 days ago	211MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB



Ver historial / capas

docker history *IMAGE[:TAG]*

→ docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
nginx	alpine	53722defe627	6 days ago	23.4MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

→ docker history hello-world

IMAGE	CREATED	CREATED BY	SIZE	COMMENT
feb5d9fea6a5	6 months ago	/bin/sh -c #(nop) CMD ["/hello"]	0B	
<missing>	6 months ago	/bin/sh -c #(nop) COPY file:50563a97010fd7ce...	13.3kB	

Ver historial / capas

→ docker history nginx:alpine

IMAGE	CREATED	CREATED BY	SIZE	COMMENT
53722defe627	6 days ago	/bin/sh -c #(nop) CMD ["nginx" "-g" "daemon...	0B	
<missing>	6 days ago	/bin/sh -c #(nop) STOPSIGNAL SIGQUIT	0B	
<missing>	6 days ago	/bin/sh -c #(nop) EXPOSE 80	0B	
<missing>	6 days ago	/bin/sh -c #(nop) ENTRYPOINT ["/docker-entr...	0B	
<missing>	6 days ago	/bin/sh -c #(nop) COPY file:09a214a3e07c919a...	4.61kB	
<missing>	6 days ago	/bin/sh -c #(nop) COPY file:0fd5fca330dcd6a7...	1.04kB	
<missing>	6 days ago	/bin/sh -c #(nop) COPY file:0b866ff3fc1ef5b0...	1.96kB	
<missing>	6 days ago	/bin/sh -c #(nop) COPY file:65504f71f5855ca0...	1.2kB	
<missing>	6 days ago	/bin/sh -c set -x && addgroup -g 101 -S ...	17.8MB	
<missing>	6 days ago	/bin/sh -c #(nop) ENV PKG_RELEASE=1	0B	
<missing>	6 days ago	/bin/sh -c #(nop) ENV NJS_VERSION=0.7.2	0B	
<missing>	6 days ago	/bin/sh -c #(nop) ENV NGINX_VERSION=1.21.6	0B	
<missing>	6 days ago	/bin/sh -c #(nop) LABEL maintainer=NGINX Do...	0B	
<missing>	6 days ago	/bin/sh -c #(nop) CMD ["/bin/sh"]	0B	
<missing>	6 days ago	/bin/sh -c #(nop) ADD file:3b5a33c96fd3c10d0...	5.57MB	



Eliminar imagen local

docker rmi *IMAGE[:TAG]*

```
→ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
wordpress     latest    ef25498ff6a9   5 days ago    605MB
phpmyadmin     fpm-alpine 7625aeffad8a   5 days ago    126MB
phpmyadmin     latest    88a6fdf429bf   5 days ago    508MB
mysql          8         667ee8fb158e   6 days ago    521MB
nginx          alpine    53722defe627   6 days ago    23.4MB
hello-world    latest    feb5d9fea6a5   6 months ago   13.3kB
```

```
→ docker rmi phpmyadmin:fpm-alpine
Untagged: phpmyadmin:fpm-alpine
Untagged: phpmyadmin@sha256:0d2a83968332cc00040e6f2a7ebfff6863d6c
Deleted: sha256:7625aeffad8a1744022b079bcb7684001ee899fbc512c398c
Deleted: sha256:a408c8eb7ae339953ea0ae12846e864e32a739183c8ea076c
Deleted: sha256:b2d73959f5006ece83187567487ac92621295377e393c3c9c
```

Eliminar imagen local

Solo se pueden eliminar imágenes que no tengan contenedor asociado, a menos que se use la opción **-f**.

```
→ docker rmi wordpress 19:51:15
Error response from daemon: conflict: unable to remove repository reference "wordpress" (must force) - container 45541b0f4f32 is using its referenced image ef25498ff6a9
```

```
→ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
45541b0f4f32	wordpress	"docker-entrypoint.s..."	4 days ago	Exited (0) 4 days ago		stoic_jones
f6558ab16dbc	wordpress	"docker-entrypoint.s..."	4 days ago	Exited (0) 4 days ago		wordpress_02
aa64663f981a	phpmyadmin	"/docker-entrypoint...."	4 days ago	Exited (0) 4 days ago		dbadmin2

Eliminar imagen local

```
→ docker rmi -f wordpress  
Untagged: wordpress:latest  
Untagged: wordpress@sha256:df2edd42c943f0925d4634718d1ed1171ea63e043a39201c0b6cbff9d470d571  
Deleted: sha256:ef25498ff6a9743827d71366f30d54ed1afcbe1ae9d65ef4325a635dea7edce8
```

Forzar el borrado de una imagen asociada a un contenedor no impide que el contenedor funcione.

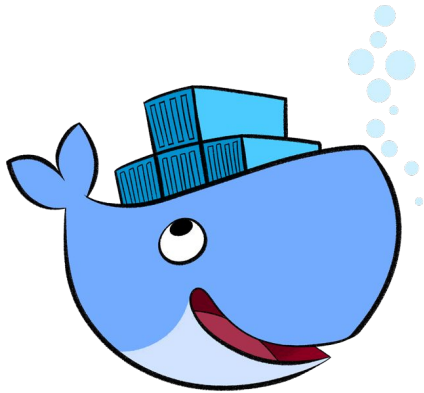


Eliminar imágenes y contenedores parados

docker system prune -a

```
→ docker system prune -a  
WARNING! This will remove:  
- all stopped containers  
- all networks not used by at least one container  
- all images without at least one container associated to them  
- all build cache  
  
Are you sure you want to continue? [y/N] y  
Deleted Containers:  
81b9551d89ea17be35f63af6b461527c7c0c415ace5710ed7f1a7406b898223b  
45541b0f4f32c6e7ada08b9d95040471914f5d1ca158b2023c0407510ce9f221  
f6558ab16dbc1be027e175027ed54690d01ff5581aa7d2502c6f1f8092e62d05
```

Creación de imágenes



Creación de imágenes

1. A partir de un contenedor
2. Exportando / Importando archivos
3. Usando el archivo Dockerfile



A partir de un contenedor

Si queremos hacer persistentes las modificaciones de un contenedor, tenemos que convertirlo en una imagen.

A partir de esa nueva imagen, se podrán crear otros contenedores que tendrán las modificaciones del inicial.

Para convertir un contenedor en una imagen, se utiliza el comando **docker commit**:

```
docker commit [opciones] id/name [REPOSITORY[:TAG]]
```

A partir de un contenedor

```
→ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
ad6662e312f2	hello-world	"/hello"	7 seconds ago	Exited (0) 6 seconds ago		<u>sleepy_montalcini</u>
20f8c843487d	phpmyadmin	"/docker-entrypoint.s..."	2 minutes ago	Exited (0) 23 seconds ago		dbadmin
dead1ecb588d	mysql:8	"docker-entrypoint.s..."	2 minutes ago	Exited (0) 23 seconds ago		mysql
08eea60ef6c6	wordpress	"docker-entrypoint.s..."	2 minutes ago	Exited (0) 23 seconds ago		wordpress_01

```
→ docker commit sleepy_montalcini  
sha256:4cc5c4d14a8deb3aa7ba6e15e633ff4b9673f023c323df1c84a4f071630917a0
```

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	4cc5c4d14a8d	5 seconds ago	13.3kB
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

A partir de un contenedor

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	<u>4cc5c4d14a8d</u>	5 seconds ago	13.3kB
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

```
~
```

```
→ docker run 4cc5c4d14a8d
```

```
Hello from Docker!
```

```
This message shows that your installation appears to be working correctly.
```

```
→ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3bdabb274a67	<u>4cc5c4d14a8d</u>	"/hello"	11 seconds ago	Exited (0) 9 seconds ago		dreamy_meitner
ad6662e312f2	hello-world	"/hello"	5 minutes ago	Exited (0) 5 minutes ago		sleepy_montalcini

A partir de un contenedor

Sin usar opciones, la imagen se crea sin repositorio ni etiqueta, lo que es poco útil más allá de una prueba.

Lo normal es asignar:

- Autor
- Comentario
- Repositorio
- Etiqueta

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	4cc5c4d14a8d	5 seconds ago	13.3kB
wordpress	latest	ef25498ff6a9	5 days ago	605MB

A partir de un contenedor

docker commit -a "Autor" -m "Mensaje" id/name REPOSITORY[:TAG]]

```
→ docker commit -a "Pepe" -m "Prueba de imagen" sleepy_montalcini pepe/hello-mod:001  
sha256:9ffb433aff99e74bac82d3d26c3986919bec0c62903bd7715a552f5aecdb63d7
```

~

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
pepe/hello-mod	001	9ffb433aff99	8 seconds ago	13.3kB
<none>	<none>	4cc5c4d14a8d	12 minutes ago	13.3kB
wordpress	latest	ef25498ff6a9	5 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	5 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

A partir de un contenedor

```
→ docker run pepe/hello-mod:001
```

```
Hello from Docker!
```

```
This message shows that your installation appears to be working correctly.
```

```
→ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7b11ccf8ddbb	<u>pepe/hello-mod:001</u>	"/hello"	3 minutes ago	Exited (0) 3 minutes ago		practical_lamarr
3bdabb274a67	4cc5c4d14a8d	"/hello"	14 minutes ago	Exited (0) 14 minutes ago		dreamy_meitner
ad6662e312f2	hello-world	"/hello"	19 minutes ago	Exited (0) 19 minutes ago		sleepy_montalcini



Exportando / Importando archivos

```
docker save IMAGE > archivo.tar  
docker save -o archivo.tar IMAGE
```

El comando **save** permite exportar una imagen a un archivo -tar.

```
→ docker save pepe/hello-mod:001 > pepe_hello_mod.001.tar  
~  
→ ls -l pepe_hello_mod.001.tar  
-rw-rw-r-- 1 pepe pepe 23552 abr  4 20:55 pepe_hello_mod.001.tar
```



Exportando / Importando archivos

docker load < archivo.tar
docker load -i archivo.tar

El comando **load** permite importar una imagen exportada con el comando **save**.

```
→ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
wordpress     latest    ef25498ff6a9   5 days ago    605MB
phpmyadmin     latest    88a6fdf429bf   5 days ago    508MB
mysql          8         667ee8fb158e   6 days ago    521MB
hello-world    latest    feb5d9fea6a5   6 months ago   13.3kB

→ docker load -i pepe_hello_mod.001.tar
Loaded image: pepe/hello-mod:001

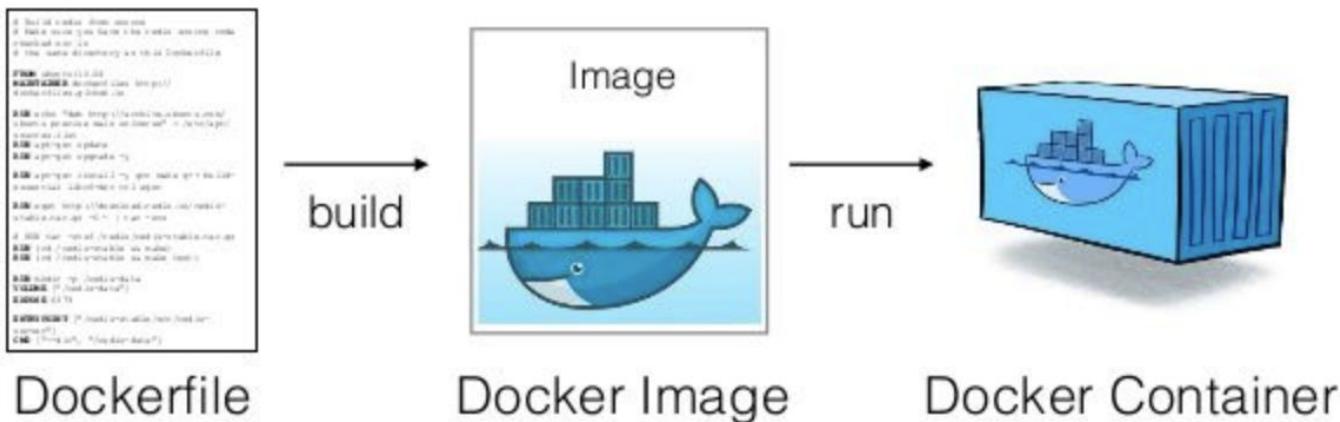
→ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
pepe/hello-mod 001       9ffb433aff99   23 minutes ago 13.3kB
wordpress     latest    ef25498ff6a9   5 days ago    605MB
phpmyadmin     latest    88a6fdf429bf   5 days ago    508MB
mysql          8         667ee8fb158e   6 days ago    521MB
hello-world    latest    feb5d9fea6a5   6 months ago   13.3kB
```



Usando el archivo Dockerfile

La última opción para crear imágenes, y quizá la más utilizada, es el archivo **Dockerfile**.

Este archivo contiene los **comandos** necesarios para crear paso a paso una imagen a partir de otra.



Usando el archivo Dockerfile

```
FROM ubuntu:latest  
  
RUN apt update && apt install -y nano  
  
# Aquí un comentario  
  
CMD /bin/bash
```

Dockerfile

Dockerfile es el nombre por defecto que debe tener el archivo.



Usando el archivo Dockerfile

docker build [opciones] PATH

Para crear la imagen, se usa el comando **docker build**:

```
~/Documentos/docker-examples  
→ ls  
Dockerfile
```

```
~/Documentos/docker-examples  
→ docker build .  
Sending build context to Docker daemon 2.048kB  
Step 1/3 : FROM ubuntu:latest  
latest: Pulling from library/ubuntu  
4d32b49e2995: Pull complete  
Digest: sha256:bea6d19168bbfd6af8d77c2cc3c572114eb5d113e6f422573c93cb605a0e2ffb  
Status: Downloaded newer image for ubuntu:latest  
---> ff0fea8310f3  
Step 2/3 : RUN apt update && apt install -y nano  
---> Running in 49167e9cf4de
```

Usando el archivo Dockerfile

Sin usar opciones, la imagen se crea sin repositorio ni etiqueta, lo que es poco útil más allá de una prueba.

```
~/Documentos/docker-examples
```

```
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	4db5e6df3016	4 seconds ago	108MB
pepe/hello-mod	001	9ffb433aff99	20 hours ago	13.3kB

Usando el archivo Dockerfile

La opción **-t** permite especificar nombre del repositorio y etiqueta:

Docker detecta que la imagen ya estaba y la sustituye.

```
~/Documentos/docker-examples
→ docker build -t ubuntu_pepe:v1.0 .
Sending build context to Docker daemon 2.048kB
Step 1/3 : FROM ubuntu:latest
---> ff0fea8310f3
Step 2/3 : RUN apt update && apt install -y nano
---> Using cache
---> 9f911dda0bbb
Step 3/3 : CMD /bin/bash
---> Using cache
---> 4db5e6df3016
Successfully built 4db5e6df3016
Successfully tagged ubuntu_pepe:v1.0

~/Documentos/docker-examples
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu_pepe	v1.0	4db5e6df3016	23 minutes ago	108MB
pepe/hello-mod	001	9ffb433aff99	21 hours ago	13.3kB
wordpress	latest	ef25498ff6a9	6 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	6 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
ubuntu	latest	ff0fea8310f3	2 weeks ago	72.8MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

Usando el archivo Dockerfile

La opción **-f** permite especificar la ruta al archivo Dockerfile (nombre incluido)

1 Copiamos el Dockerfile con otro nombre

2 Construimos la imagen con otra etiqueta

3 Docker detecta que es la misma imagen, pero al tener otra etiqueta, crea otra imagen.

```
~/Documentos/docker-examples
→ cp Dockerfile Dockerfile.local

~/Documentos/docker-examples
→ docker build -t ubuntu_pepe:v1.1 -f Dockerfile.local .
Sending build context to Docker daemon 3.072kB
Step 1/3 : FROM ubuntu:latest
--> ff0fea8310f3
Step 2/3 : RUN apt update && apt install -y nano
--> Using cache
--> 9f911dda0bbb
Step 3/3 : CMD /bin/bash
--> Using cache
--> 4db5e6df3016
Successfully built 4db5e6df3016
Successfully tagged ubuntu_pepe:v1.1

~/Documentos/docker-examples
→ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu_pepe	v1.0	4db5e6df3016	37 minutes ago	108MB
ubuntu_pepe	v1.1	4db5e6df3016	37 minutes ago	108MB
pepe/hello-mod	001	9ffb433aff99	21 hours ago	13.3kB
wordpress	latest	ef25498ff6a9	6 days ago	605MB
phpmyadmin	latest	88a6fdf429bf	6 days ago	508MB
mysql	8	667ee8fb158e	6 days ago	521MB
ubuntu	latest	ff0fea8310f3	2 weeks ago	72.8MB
hello-world	latest	feb5d9fea6a5	6 months ago	13.3kB

Usando el archivo Dockerfile

El archivo Dockerfile admite muchos comandos. Los más importantes son:

- FROM
- RUN
- CMD
- ENTRYPOINT
- EXPOSE
- ADD/COPY
- USER
- WORKDIR
- ENV

A continuación se presenta un breve resumen de cada comando. Para detalles de su utilización, debe consultarse la [documentación oficial](#).



Usando el archivo Dockerfile

FROM se usa para especificar la imagen base. Debe ser la primera instrucción del Dockerfile.



```
FROM ubuntu:latest  
  
RUN apt update && apt install -y nano  
  
# Aquí un comentario  
  
CMD /bin/bash
```



Usando el archivo Dockerfile

RUN se usa para ejecutar un comando y crear una nueva capa en la imagen.



```
FROM ubuntu:latest  
  
RUN apt update && apt install -y nano  
  
# Aquí un comentario  
  
CMD /bin/bash
```



Usando el archivo Dockerfile

CMD se usa para especificar el comando que se ejecutará al ejecutar el contendor.

Solo puede haber un CMD en el Dockerfile. Si hay varios, solo el último tiene efecto.

```
FROM ubuntu:latest

RUN apt update && apt install -y nano

# Aquí un comentario

→ CMD /bin/bash
```



Usando el archivo Dockerfile

ENTRYPOINT se usa para especificar el comando que se ejecutará al ejecutar el contenedor.

Solo puede haber un ENTRYPOINT en el Dockerfile. Si hay varios, solo el último tiene efecto.

ENTRYPOINT y **CMD** interactúan para determinar el comando que se ejecutará al ejecutarse el contenedor. El resultado depende de si se han declarado o no, y de qué formato se ha usado.

Usando el archivo Dockerfile

En la siguiente tabla se veel resultado de diferentes combinaciones:

	No ENTRYPOINT	ENTRYPOINT exec_entry p1_entry	ENTRYPOINT ["exec_entry", "p1_entry"]
No CMD	<i>error, not allowed</i>	/bin/sh -c exec_entry p1_entry	exec_entry p1_entry
CMD ["exec_cmd", "p1_cmd"]	exec_cmd p1_cmd	/bin/sh -c exec_entry p1_entry	exec_entry p1_entry exec_cmd p1_cmd
CMD ["p1_cmd", "p2_cmd"]	p1_cmd p2_cmd	/bin/sh -c exec_entry p1_entry	exec_entry p1_entry p1_cmd p2_cmd
CMD exec_cmd p1_cmd	/bin/sh -c exec_cmd p1_cmd	/bin/sh -c exec_entry p1_entry	exec_entry p1_entry /bin/sh -c exec_cmd p1_cmd



Usando el archivo Dockerfile

EXPOSE se usa para indicar los puertos que escucha un contenedor.

Se utiliza a modo de **documentación**, ya que los puertos expuestos son los susceptibles de ser mapeados (publicados) con la opción **-p de docker run**.

FROM ubuntu:latest

RUN apt update && apt install -y apache2

EXPOSE 80

ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]



Usando el archivo Dockerfile

COPY se usa para copiar archivos entre el host y el contenedor.

Solo permite copiar **archivos locales**.

```
FROM ubuntu:latest
```

```
RUN apt update && apt install -y apache2
```

```
EXPOSE 80
```

```
COPY ["index.html", "/var/www/html/"]
```

```
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```



Usando el archivo Dockerfile

ADD se usa para copiar archivos entre el host y el contenedor.

Permite copiar **archivos locales y remotos** a través de la url.

También permite **descomprimir** archivos tar comprimidos.

```
FROM ubuntu:latest
```

```
RUN apt update && apt install -y apache2
```

```
EXPOSE 80
```

```
→ ADD ["index.html", "/var/www/html/"]
```

```
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```



Usando el archivo Dockerfile

USER especifica el usuario y/o grupo con los que se ejecutarán el contenedor y los comandos RUN, CMD y ENTRYPOINT.

Por defecto es el root.

```
FROM ubuntu:latest
```

```
RUN apt update && apt install -y apache2
```

```
EXPOSE 80
```

```
USER pepe
```

```
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```



Usando el archivo Dockerfile

WORKDIR se usa para cambiar el directorio actual para cualquier comando RUN, CMD, ENTRYPOINT, COPY y ADD posterior.

Si no existe, se crea.

```
FROM ubuntu:latest

RUN apt update && apt install -y apache2
EXPOSE 80

USER pepe
WORKDIR /home/pepe
ADD ["files.zip","archivos"]

ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```



Usando el archivo Dockerfile - *ENV*

ENV se usa para crear variables de entorno.

```
FROM ubuntu:latest
```

```
RUN apt update && apt install -y apache2
```

```
EXPOSE 80
```

```
USER pepe
```

```
ENV V1="valor1" V2="valor2"
```

```
ENV MY_NAME="John Doe"
```

```
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Contenedores e imagens

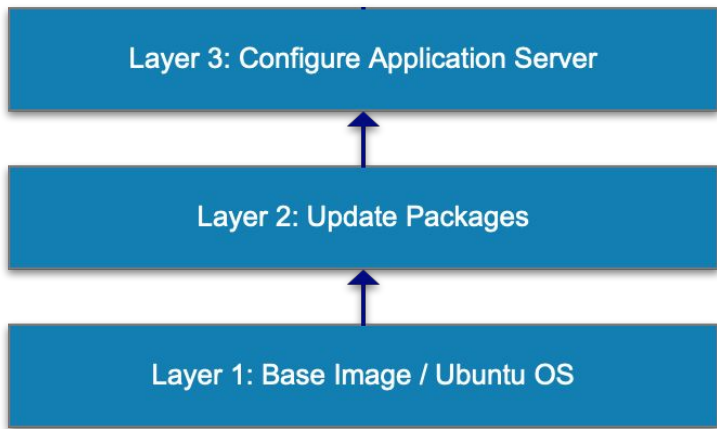


Contenedores e imágenes

Las imágenes se crean **por capas**.

Cada instrucción RUN, COPY y ADD añade **una nueva capa**.

Docker **comparte** las capas iguales entre imágenes para ahorrar espacio.



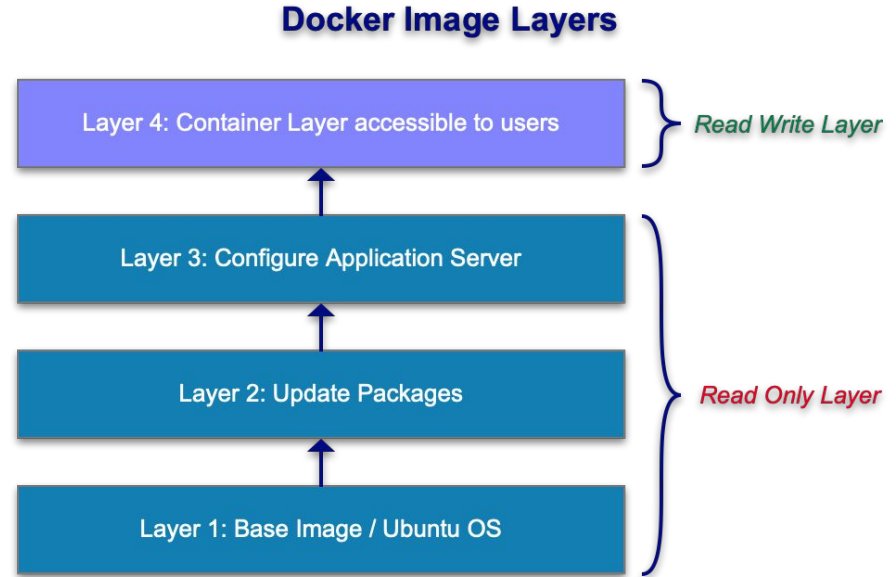
Contenedores e imágenes



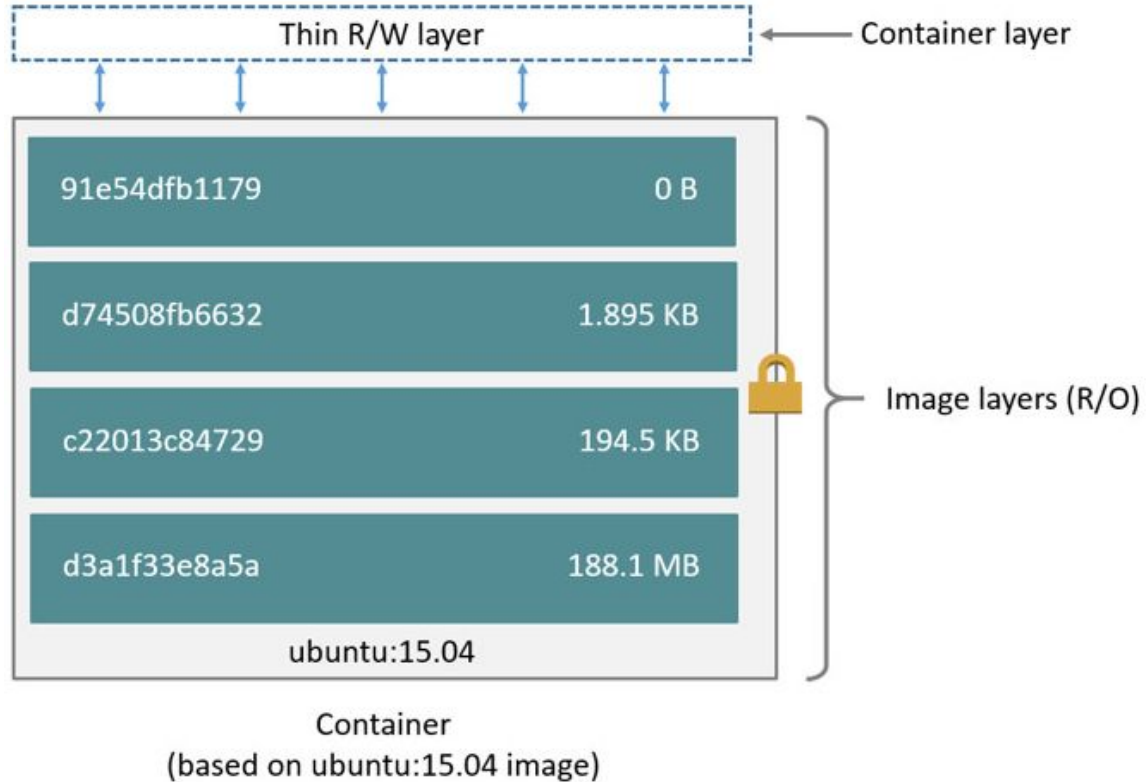
Contenedores e imágenes

Cuando se crea un contenedor, se usan las capas de la imagen como capas de **solo lectura**.

Encima se añade una capa de **lectura y escritura** que contienen las modificaciones hechas por el contenedor.

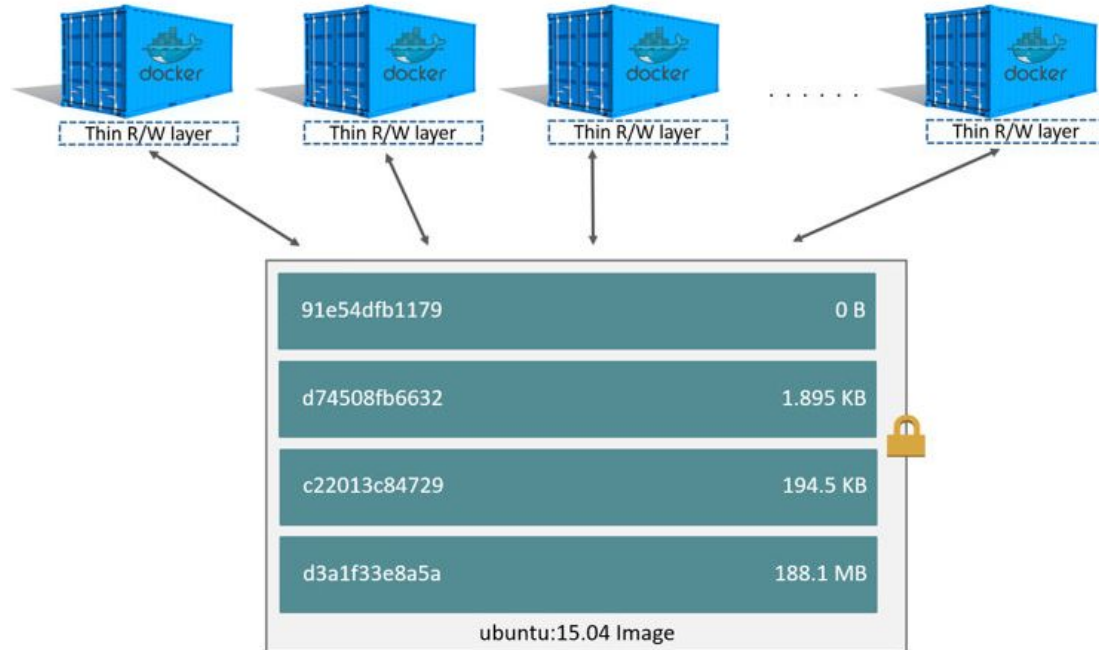


Contenedores e imágenes



Contenedores e imágenes

Varios contenedores creados a partir de una misma imagen, comparten las capas de dicha imagen, ahorrando espacio en disco.



Contenedores e imágenes

Cuando se crea una imagen a partir de un contenedor, usando **docker commit**, la capa de lectura y escritura se convierte en una capa de solo lectura.

docker history nos permite ver las capas:

```
→ docker history hello-world:latest
IMAGE          CREATED          CREATED BY          SIZE      COMMENT
feb5d9fea6a5   6 months ago    /bin/sh -c #(nop)  CMD ["/hello"]     0B
<missing>      6 months ago    /bin/sh -c #(nop)  COPY file:50563a97010fd7ce... 13.3kB
```

```
→ docker history pepe/hello-mod:001
IMAGE          CREATED          CREATED BY          SIZE      COMMENT
9ffb433aff99   45 hours ago    /hello             0B        Prueba de imagen
<missing>      6 months ago    /bin/sh -c #(nop)  CMD ["/hello"]     0B
<missing>      6 months ago    /bin/sh -c #(nop)  COPY file:50563a97010fd7ce... 13.3kB
```

Ejercicio 1

Ejercicio 1

Se tiene una carpeta html con una web estática (index.html) y se pide crear una imagen Docker con el servidor web Apache que muestre la web.

Requisitos:

- Imagen base: debian
- Comandos para instalar Apache:
 - apt-get update
 - apt-get install -y apache2
 - apt-get clean
 - rm -rf /var/lib/apt/lists/*
- Debe exponer el puerto 80
- Comando para ejecutar Apache: `/usr/sbin/apache2ctl -D FOREGROUND`

index.html

```
<h1>Ejercicio 1</h1>  
<p>Hola! Todo OK!</p>
```


Ejercicio 2

Ejercicio 2

Repita el ejercicio 1 pero usando como imagen base la oficial de Apache.

Ejercicio 3

Ejercicio 3

Repita el ejercicio 2 pero usando como imagen base la oficial de Nginx.

Ejercicio 4

Ejercicio 4

Se tiene una carpeta app con una página web dinámica con php (index.php) y se pide crear una imagen Docker con el servidor web Apache y php que muestre la web.

Requisitos:

- Imagen base: debian
- Comandos para instalar Apache:
 - apt-get update
 - apt-get install -y apache2 libapache2-mod-php7.3 php7.3
 - apt-get clean
 - rm -rf /var/lib/apt/lists/*
 - rm /var/www/html/index.html
- Debe exponer el puerto 80
- Comando para ejecutar Apache: /usr/sbin/apache2ctl -D FOREGROUND

index.php

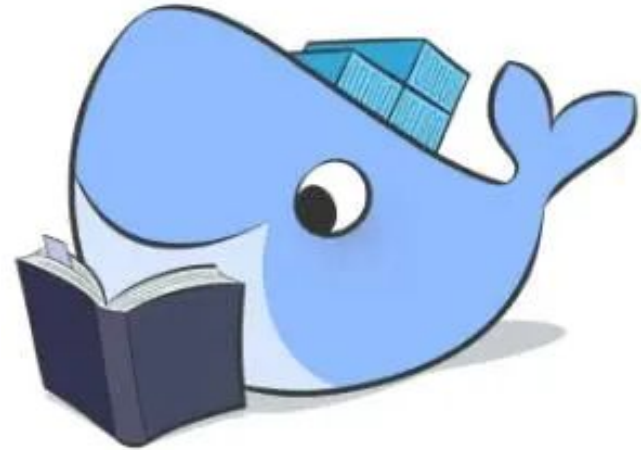
```
<?php echo phpinfo();?>
```

Ejercicio 5

Ejercicio 5

Repita el ejercicio 4 pero usando como imagen base `php:apache`, que es una imagen de oficial de PHP con Apache ya instalado.

Publicación de imágenes



Publicación de imágenes

Se pueden publicar imágenes en el repositorio para ser utilizadas posteriormente.

Para publicar en **Docker Hub**, es necesario tener **una cuenta**.

El procedimiento tiene 2 pasos:

1. Iniciar sesión con **docker login**
2. Subir la imagen con **docker push**



Publicación de imágenes

docker login

```
→ docker login
```

```
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
```

```
Username: pposca
```

```
Password:
```

```
WARNING! Your password will be stored unencrypted in /home/pepe/.docker/config.json.
```

```
Configure a credential helper to remove this warning. See
```

```
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

```
Login Succeeded
```



Publicación de imágenes

docker push


```
→ docker push pposca/ejer5:v1  
The push refers to repository [docker.io/pposca/ejer5]  
1bca420e374b: Pushed  
78d1dc086f2e: Mounted from library/php  
c67db8aa2359: Mounted from library/php  
9108a3392c29: Mounted from library/php  
fd7fb2d8362b: Mounted from library/php  
be00d6205d59: Mounted from library/php  
d3ac239ebf10: Mounted from library/php  
d1a9206e71d3: Mounted from library/php  
70c7e28ebf9b: Mounted from library/php  
8da8ace4fad7: Mounted from library/php  
a9a0b6ccdc20: Mounted from library/php  
1fef65785df5: Mounted from library/php  
ea61222fc24f: Mounted from library/php  
608f3a074261: Mounted from library/php  
v1: digest: sha256:bd57620325d291741a6091383f15cf2bd7abfc81bcfb4e46dca190933dabb468 size: 3242
```

Publicación de imágenes

El nombre de la imagen tienen que ser: **username/imagen:tag**

[Repositories](#) [Starred](#) [Contributed](#)

Displaying 1 of 1 repository



pposca/ejer5

By [pposca](#) • Updated an hour ago

Container

0
Stars

Ejercicio 6

Ejercicio 6

Crea una cuenta en Docker Hub, crea una imagen y súbela.

Puede utilizar una imagen de las creadas en los ejercicios anteriores.

Recuerda que la primera parte del nombre de la imagen tiene que coincidir con el nombre de usuario de Docker Hub.

Luego borra la imagen local y usa `docker run` para que se la descargue del repositorio, cree un contenedor y lo ejecute.

Documenta el proceso con explicaciones y capturas de pantalla.