



UNIVERSIDAD DE GRANADA

Centro de Procesamiento de Datos

Práctica 2 – Contenedores Docker II

Arturo Alonso Carbonero

Apartado I

Para poder crear una página personalizada y acceder a la misma, es necesario crear un fichero para proporcionárselo al contenedor pertinente. En este caso, el fichero está situado en el mismo directorio que el *Dockerfile* para evitar concretar la ruta completa del mismo. En la imagen siguiente se muestra el contenido del fichero **index.html** con el que se trabajará.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ cat index.html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Practica 2 CPD</title>
  </head>
  <body>
    <h1>Acceso personalizado usando Dockerfile</h1>
    <a>Realizado por Arturo Alonso Carbonero</a>
  </body>
</html>
```

Fichero index.html

A continuación, debe ser creado el *Dockerfile*, que permite automatizar el proceso de creación de un contenedor. El contenido del *Dockerfile* para este ejercicio es el de la siguiente imagen, en el que se emplean las directrices a continuación listadas:

- **FROM debian** → Inicializa un nuevo montaje a partir de la imagen oficial de Debian.
- **MAINTAINER** → Establece el autor del *Dockerfile*. Esta orden está obsoleta.
- **RUN** → Establece los comandos a ejecutar dentro del contenedor.
- **ENV** → Establece la variable del entorno para el resto de órdenes.
- **EXPOSE** → Expone el puerto indicado.
- **ADD** → Permite añadir un fichero local al contenedor.
- **ENTRYPOINT** → Permite configurar un contenedor que se ejecutará como un ejecutable.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ cat Dockerfile
FROM debian
MAINTAINER alonsoarturo "alonsoarturo@correo.ugr.es"
RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2
EXPOSE 80
ADD ["index.html", "/var/www/html/"]
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Fichero Dockerfile

Una vez disponemos del fichero *Dockerfile*, situándonos en el mismo directorio, ejecutamos la orden **docker build** para lanzar el proceso de creación del contenedor. En este caso, lo hacemos con la opción **-t**, que permite nombrar y, opcionalmente, etiquetar el contenedor a crear.

```

alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker build
-t practica2_cpd .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.072kB
Step 1/9 : FROM debian
latest: Pulling from library/debian
167b8a53ca45: Pull complete
Digest: sha256:eaace54a93d7b69c7c52bb8ddf9b3fcb0c106a497bc1fdbb89a6299cf945c63
Status: Downloaded newer image for debian:latest
--> 2657a4a0a6d5
Step 2/9 : MAINTAINER alonsoarturo "alonsoarturo@correo.ugr.es"
--> Running in 3ecc69bcd5a4
Removing intermediate container 3ecc69bcd5a4
--> 8399e31aec1a
Step 3/9 : RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm
-rf /var/lib/apt/lists/*
--> Running in fa3b5cf73b0f
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]

```

Fragmento del resultado de la orden docker build

La orden **docker build** ejecuta el fichero *Dockerfile* de forma secuencial, descargando las imágenes necesarias y realizando paso a paso todos los establecidos para crear la imagen deseada. Para lanzar el contenedor, hacemos uso de la orden **docker run** y comprobamos si el proceso ha sido exitoso con la orden **docker ps**, tal y como en la práctica anterior.

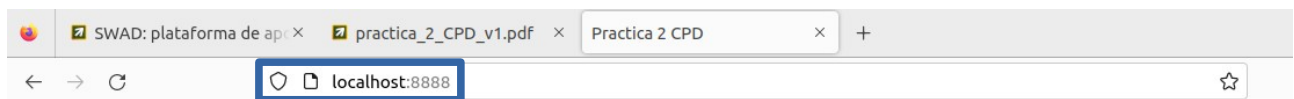
```

alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker run -
-name x1 -p8888:80 -d practica2_cpd
7d000c283a9038702c7871c43d0e19e476f9b506094b71083862ea61534c0371
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
7d000c283a90   practica2_cpd  "/usr/sbin/apache2ct..." 8 seconds ago  Up 6 secon
ds            0.0.0.0:8888->80/tcp, :::8888->80/tcp   x1

```

Resultado obtenido

Para comprobar que el fichero se ha subido correctamente al contenedor, accedemos desde un navegador al puerto indicado en la orden **docker run** para ver si el resultado es correcto.



Acceso personalizado usando Dockerfile

Realizado por Arturo Alonso Carbonero

Resultado en navegador

Además, la orden **curl** nos permite comprobar desde terminal si el contenedor se está ejecutando correctamente.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ curl -I localhost:8888
HTTP/1.1 200 OK
Date: Sun, 01 Oct 2023 17:25:28 GMT
Server: Apache/2.4.57 (Debian)
Last-Modified: Sun, 01 Oct 2023 17:05:09 GMT
ETag: "db-606aaa8f37b40"
Accept-Ranges: bytes
Content-Length: 219
Vary: Accept-Encoding
Content-Type: text/html

alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ curl localhost:8888
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Practica 2 CPD</title>
  </head>
  <body>
    <h1>Acceso personalizado usando Dockerfile</h1>
    <a>Realizado por Arturo Alonso Carbonero</a>
  </body>
</html>
```

Resultado con curl

Apartado II

Para poder subir una imagen creada al repositorio, es necesario disponer de una cuenta en el sitio oficial de Docker. En local, desde terminal, podemos ejecutar la orden **docker login** para acceder a la misma.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo 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: alonsoarturo
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Login desde terminal

Para desconectarse, basta con ejecutar **docker logout**.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker logout
Removing login credentials for https://index.docker.io/v1/
```

Logout desde terminal

Para este apartado, se empleará el mismo *Dockerfile* que en el apartado anterior, ejecutando además una orden **echo** para distinguirlo del anterior.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ cat Dockerfile
FROM debian
MAINTAINER alonsoarturo "alonsoarturo@correo.ugr.es"
RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*
RUN echo alonsoarturo

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2
EXPOSE 80
ADD ["index.html", "/var/www/html/"]
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Fichero Dockerfile

Ejecutamos la orden **docker build**, esta vez estableciendo una etiqueta (1.0), para crear la imagen. Además, en este caso, el nombre de la imagen, cuenta con el nombre de usuario empleado.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker build -t alonsoarturo/practica2_cpd:1.0 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 3.072kB
Step 1/10 : FROM debian
--> 2657a4a0a6d5
Step 2/10 : MAINTAINER alonsoarturo "alonsoarturo@correo.ugr.es"
--> Using cache
--> 8399e31aec1a
Step 3/10 : RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*
--> Using cache
--> 469a37b350c2
```

Fragmento del resultado de la orden docker build

Para subir la imagen al repositorio, hacemos uso de la orden **docker push**.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker push alonsoarturo/practica2_cpd:1.0
The push refers to repository [docker.io/alonsoarturo/practica2_cpd]
7bc1ae07483b: Pushed
2b086e95dc78: Pushed
7c85cfa30cb1: Mounted from library/debian
1.0: digest: sha256:6845544300021df292ed8dcf0c7189ca8498ce9c1bd2a1c78430533ec27edf3f size: 948
```

Resultado de la orden docker push

Podemos hacer uso de la orden **docker images** para asegurarnos de que la imagen se ha creado correctamente en caso de requerirlo.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
alonsoarturo/practica2_cpd	1.0	22f30b41160a	33 minutes ago	233MB
practica2_cpd	latest	22f30b41160a	33 minutes ago	233MB
imagenpractica1	latest	0298768d05da	5 days ago	194MB
nginx	latest	61395b4c586d	11 days ago	187MB
debian	latest	2657a4a0a6d5	11 days ago	116MB
fedora	latest	621310b5b7d8	2 weeks ago	191MB
ubuntu	latest	c6b84b685f35	6 weeks ago	77.8MB
alpine	latest	7e01a0d0a1dc	7 weeks ago	7.34MB
centos	latest	5d0da3dc9764	2 years ago	231MB

Imagen creada correctamente

Además, podemos descargar la imagen usando **docker pull**.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker pull alonsoarturo/practica2_cpd:1.0
1.0: Pulling from alonsoarturo/practica2_cpd
Digest: sha256:6845544300021df292ed8dcf0c7189ca8498ce9c1bd2a1c78430533ec27edf3f
Status: Downloaded newer image for alonsoarturo/practica2_cpd:1.0
docker.io/alonsoarturo/practica2_cpd:1.0
```

Resultado de la orden docker pull

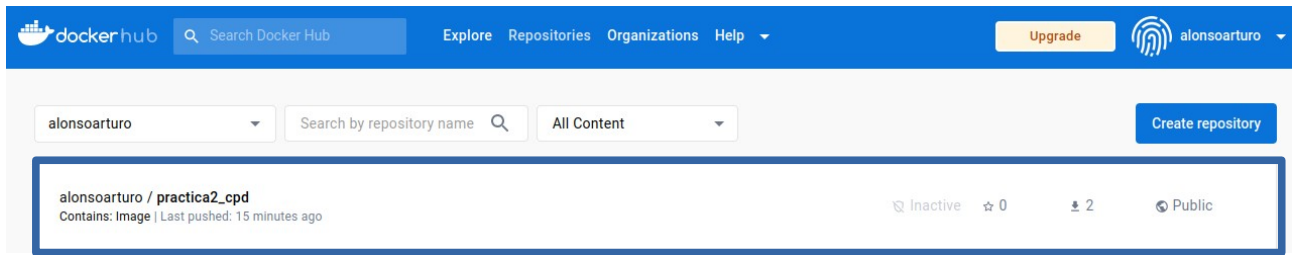
Además, si ejecutamos la orden **docker build** con la opción **--no-cache**, podemos evitar que se emplee memoria caché en la creación del fichero, de forma que no se creen las imágenes intermedias que, por defecto, genera la orden **docker build**. Así, aceleramos el proceso de creación en cada etapa.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2$ sudo docker build --no-cache -t alonsoarturo/practica2_cpd:1.0 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
             Install the buildx component to build images with BuildKit:
             https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.072kB
Step 1/10 : FROM debian
--> 2657a4a0a6d5
Step 2/10 : MAINTAINER alonsoarturo "alonsoarturo@correo.ugr.es"
--> Running in 278cba0d8ff0
Removing intermediate container 278cba0d8ff0
--> dbb550278938
```

Docker build con la opción --no-cache

Para comprobar que la imagen se ha subido correctamente al repositorio, podemos simplemente acceder a nuestra cuenta y consultar el apartado *Repositories*.



Resultado obtenido

Apartado III

Es posible que sea necesario crear soluciones más complejas con más de una aplicación. Para ello, hay que hacer uso de **docker-compose**, que permite combinar múltiples contenedores. Para ello, creamos en primer lugar un fichero **docker-compose.yml** que contendrá las directrices para la creación del entorno deseado. Para este caso (Wordpress + MySQL), hay que indicar los servicios que se van a emplear en la directiva **services**, así como la imagen, el volumen y el entorno, entre otros, de cada uno. Es necesario además que las credenciales que se definen para ambos servicios coincidan, para evitar conflictos a la hora de acceder a Wordpress.



```
alonsoarturo@alonsoarturo-VirtualBox: ~/Escrit...
GNU nano 6.2 docker-compose.yml *
version: '3.3'

services:
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: alonsoarturo
      MYSQL_PASSWORD: cpdwordpress

  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: alonsoarturo
      WORDPRESS_DB_PASSWORD: cpdwordpress
volumes:
  db_data: {}
```

Fcihero docker-compose.yml

Para lanzar el fichero anteriormente creado, hacemos uso de la orden **docker-compose up**, que realiza precisamente esto, en este caso con la opción **-d**, que ejecuta el contenedor en segundo plano.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Wordpress$ sudo docker-compose up -d
Creating network "wordpress_default" with the default driver
Creating volume "wordpress_db_data" with default driver
Pulling db (mysql:5.7)...
5.7: Pulling from library/mysql
70e9ff4420fb: Pull complete
7ca4383b183f: Pull complete
3e282e7651b1: Pull complete
1ffa0e0ca707: Pull complete
6eb790cf6382: Pull complete
b4b277ff2929: Pull complete
692fe4469429: Pull complete
c0d447d97bbd: Pull complete
99ee594517ba: Pull complete
```

Fragmento del resultado de la orden docker-compose up

```
Digest: sha256:a078d12a76827322cf95c1e3ae470d30bd518af6e0bdb42f4f49919b2b3ba74b
Status: Downloaded newer image for wordpress:latest
Creating wordpress_db_1 ... done
Creating wordpress_wordpress_1 ... done
```

Fragmento del resultado de la orden docker-compose up

Para comprobar que el contenedor se ha creado correctamente, podemos hacer uso de la opción **docker-compose ps**.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Wordpress$ sudo docker-compose ps
```

Name	Command	State	Ports
wordpress_db_1	docker-entrypoint.sh mysqld	Up	3306/tcp, 33060/tcp
wordpress_wordpress_1	docker-entrypoint.sh apach ...	Up	0.0.0.0:8000->80/tcp, :::8000->80/tcp

Contenedor correctamente creado

Para acceder al servicio, nos conectamos desde un navegador al *host* local a través del puerto indicado en el fichero *.yaml*.



localhost:8000/wp-admin/install.php?step=1



Hola

¡Este es el famoso proceso de instalación de WordPress en cinco minutos! Simplemente completa la información siguiente y estarás a punto de usar la más enriquecedora y potente plataforma de publicación personal del mundo.

Información necesaria

Por favor, proporciona la siguiente información. No te preocupes, siempre podrás cambiar estos ajustes más tarde.

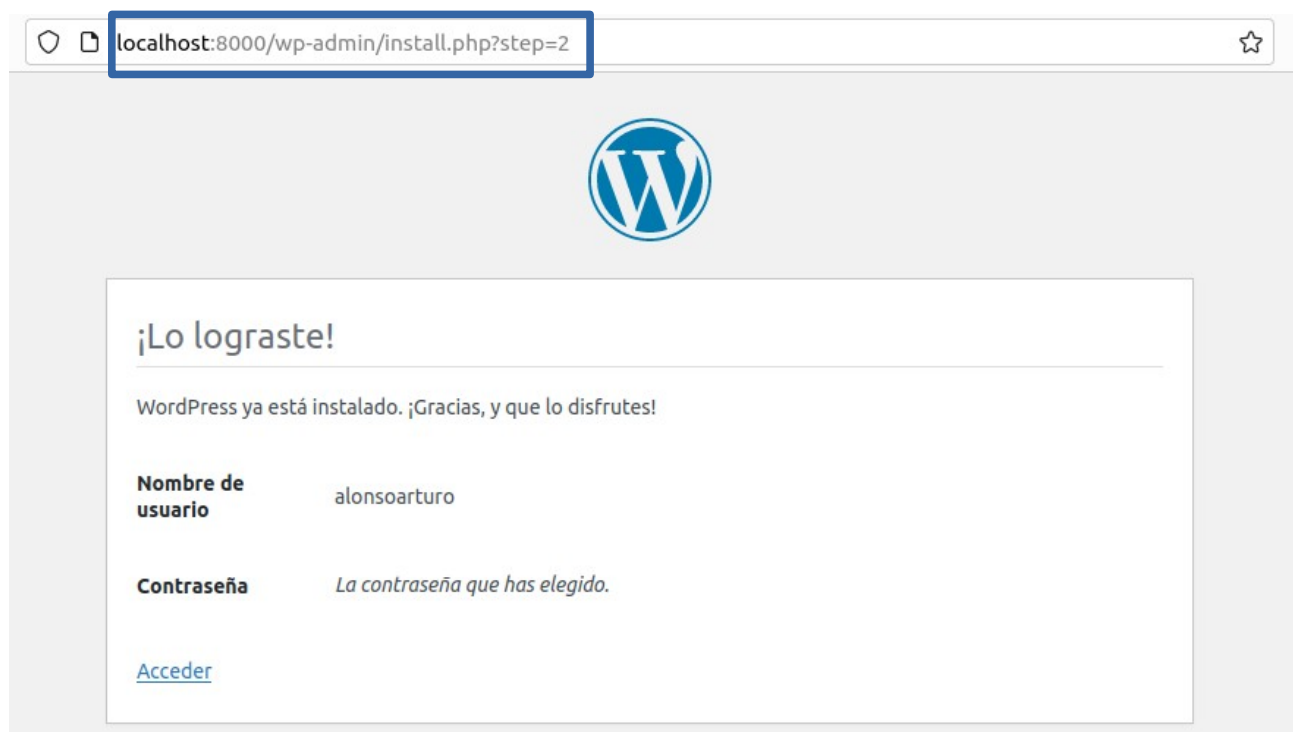
Título del sitio

Nombre de usuario


Los nombres de usuario pueden tener únicamente caracteres alfanuméricos, espacios,

Configuración de Wordpress

Tras completar el proceso de instalación de Wordpress, podemos comprobar que el resultado es el deseado.



localhost:8000/wp-admin/install.php?step=2



¡Lo lograste!

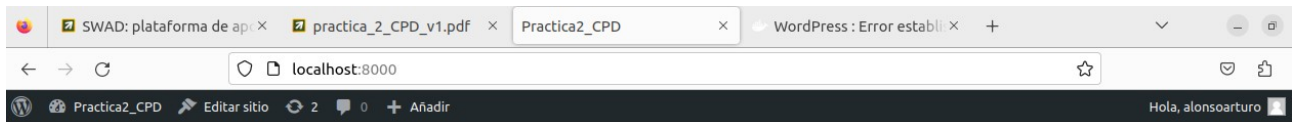
WordPress ya está instalado. ¡Gracias, y que lo disfrutes!

Nombre de usuario alonsoarturo

Contraseña La contraseña que has elegido.

[Acceder](#)

Resultado final



Practica2_CPD

Página de ejemplo

Mindblown: a blog about philosophy.

¡Hola, mundo!

Fichero de ejemplo por defecto

Apartado IV (opcional)

Para ejecutar los tests de rendimiento de este apartado, he optado por crear un contenedor sencillo a partir de la imagen básica de Ubuntu. Para poder realizar los tests, el *Dockerfile* instala directamente **sysbench**, que permite precisamente realizar pruebas de rendimiento en un equipo.

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Opcional$ cat Dockerfile
FROM ubuntu

RUN apt-get update && apt-get install -y sysbench
```

Fichero Dockerfile

Realizamos un proceso similar al de los apartados anteriores para subir la imagen al repositorio.

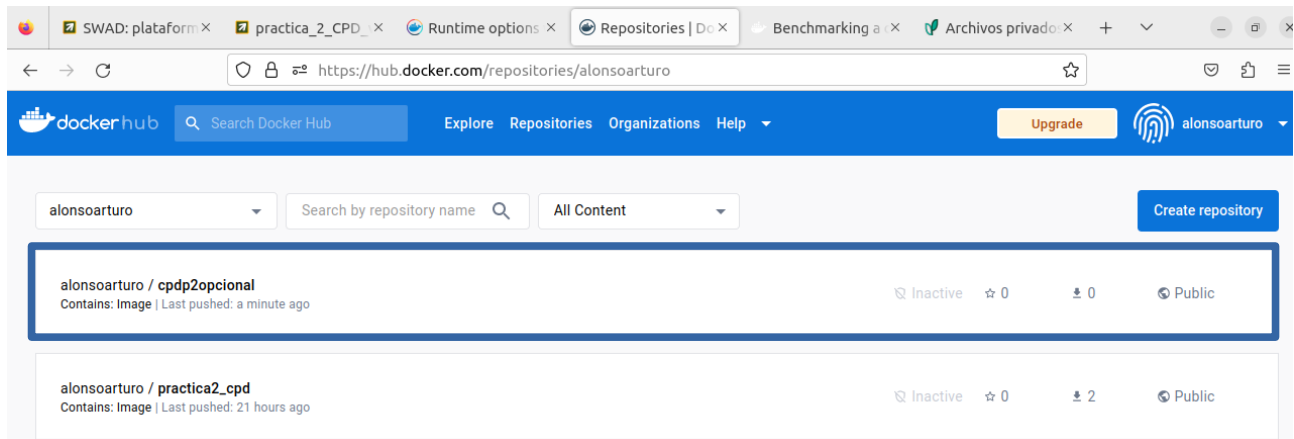
```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Opcional$ sudo docker build -t
alonsoarturo/cdp2opcional:1.0 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
             Install the buildx component to build images with BuildKit:
             https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM ubuntu
--> c6b84b685f35
Step 2/2 : RUN apt-get update && apt-get install -y sysbench
--> Running in 52944acc60dc
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
```

Creación de la imagen

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Opcional$ sudo docker push alonsoarturo/cdp2opcional:1.0
The push refers to repository [docker.io/alonsoarturo/cdp2opcional]
d1d67ab0008f: Pushed
dc0585a4b8b7: Mounted from library/ubuntu
1.0: digest: sha256:7866df7631f14c9e70202a7a247f53560cca950d2e09c16a3a06799583a28243 size: 741
```

Subida de la imagen al repositorio



Comprobación

A continuación, se muestran los resultados de ejecutar la orden **docker run** con diferentes opciones para controlar los recursos que el contenedor puede utilizar. Puesto que se trata del contenedor más sencillo posible, el tiempo no varía, pero hay ciertos valores que se pueden destacar.

Ejecución básica

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Opcional$ sudo docker
run -it alonsoarturo/cpdp2opcional:1.0 /bin/sh
# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the c
ommand line without any options.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!
```

Ejecución básica

```
CPU speed:
  events per second:   440.36

General statistics:
  total time:                   10.0017s
  total number of events:       4405

Latency (ms):
  min:                        2.16
  avg:                        2.27
  max:                        8.06
  95th percentile:           2.61
  sum:                        9992.71

Threads fairness:
  events (avg/stddev):       4405.0000/0.00
  execution time (avg/stddev): 9.9927/0.00
```

Resultado

Ejecución con opción --cpus

```
alonsoarturo@alonsoarturo-VirtualBox: ~/Escritorio/CPD/Práctica2/Opcional$ sudo docker
run -it --cpus=".5" alonsoarturo/cdp2opcional:1.0 /bin/sh
# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the c
ommand line without any options.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!
```

Ejecución limitando el uso de CPU

```
CPU speed:
  events per second:   190.97

General statistics:
  total time:                   10.0003s
  total number of events:       1910

Latency (ms):
  min:                          2.16
  avg:                          5.23
  max:                          252.58
  95th percentile:             10.09
  sum:                          9993.37

Threads fairness:
  events (avg/stddev):       1910.0000/0.00
  execution time (avg/stddev): 9.9934/0.00
```

Resultado

En comparación el resultado anterior, el número de eventos es aproximadamente la mitad. Además, la latencia máxima es mucho mayor.

Ejecución con opción --cpuset-cpus

```
alonsoarturo@alonsoarturo-VirtualBox: ~/Escritorio/CPD/Práctica2/Opcional$ sudo docker
run -it --cpuset-cpus="0-1" alonsoarturo/cdp2opcional:1.0 /bin/sh
# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the c
ommand line without any options.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!
```

Ejecución limitando el número de CPUs disponibles

```
CPU speed:
  events per second:   440.65

General statistics:
  total time:                   10.0018s
  total number of events:       4408

Latency (ms):
  min:                        2.16
  avg:                        2.27
  max:                        4.31
  95th percentile:           2.57
  sum:                        9992.46

Threads fairness:
  events (avg/stddev):       4408.0000/0.00
  execution time (avg/stddev): 9.9925/0.00
```

Resultado

El resultado es similar a la ejecución básica, pero si disminuye en la mitad la latencia máxima.

Ejecución con opción --cpu-shares

```
alonsoarturo@alonsoarturo-VirtualBox:~/Escritorio/CPD/Práctica2/Opcional$ sudo docker
run -it --cpu-shares="2048" alonsoarturo/cdp2opcional:1.0 /bin/sh
# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the c
ommand line without any options.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!
```

Ejecución aumento el doble el peso del contenedor

```
CPU speed:
  events per second:   435.61

General statistics:
  total time:                   10.0022s
  total number of events:       4358

Latency (ms):
  min:                          2.16
  avg:                          2.29
  max:                          18.22
  95th percentile:             2.71
  sum:                          9994.42

Threads fairness:
  events (avg/stddev):       4358.0000/0.00
  execution time (avg/stddev): 9.9944/0.00
```

Resultado

El mayor peso del contenedor hace que la latencia máxima aumente en este caso.

Referencias

Apartado I

- <https://docs.docker.com/engine/reference/builder/#from>
- <https://docs.docker.com/engine/reference/commandline/build/>

Apartado II

- <https://docs.docker.com/engine/reference/commandline/pull/>

Apartado III

- <https://docs.docker.com/compose/>
- <https://docs.docker.com/compose/features-uses/>
- https://docs.docker.com/engine/reference/commandline/compose_up/
- <https://forums.docker.com/t/wordpress-error-establishing-a-database-connection/103843/2>

Apartado IV (opcional)

- https://docs.docker.com/config/containers/resource_constraints/#configure-the-default-cfs-scheduler
- <https://forums.docker.com/t/benchmarking-a-docker-container/23943/2>