

Puesta en Producción Segura

Práctica 2: DevSecOps

Objetivo

El objetivo de esta práctica es que el alumno comprenda los fundamentos de las DevOps seguras, también conocidas como DevSecOps.

Corrección de la práctica

El alumno deberá entregar una memoria con las soluciones a cada apartado y los pasos detallados que ha seguido en el proceso. Se valorará:

- La calidad de la memoria. Se deberán explicar y justificar los procedimientos y resultados, así como de dar soporte a lo descrito mediante capturas de pantalla.
- La corrección del proceso descrito.
- La corrección de los comandos empleados y de las opciones.
- La correcta justificación de los pasos seguidos.
- La propuesta de soluciones alternativas, documentando incluso aquellas que siendo razonables no han logrado su propósito.

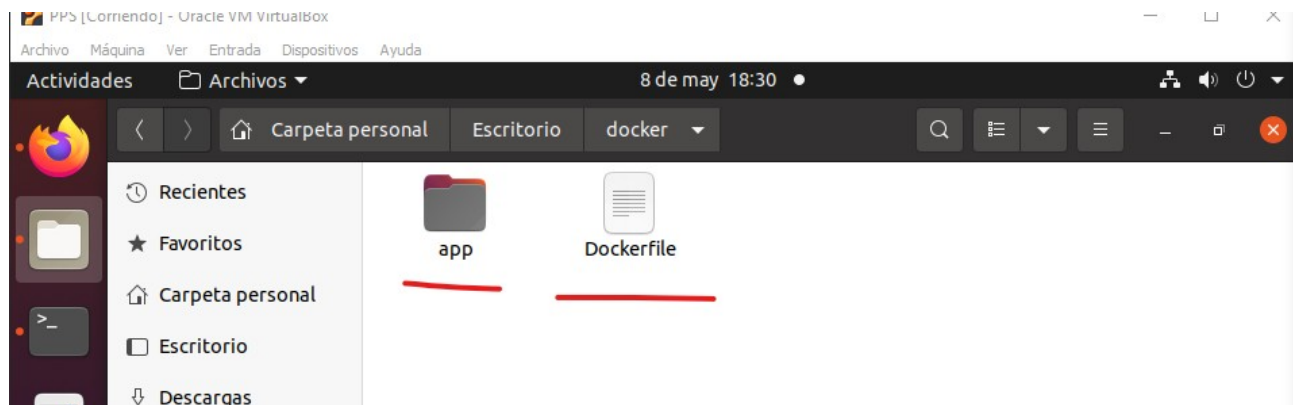
Forma de entrega: Vía Moodle. Se habilitará una tarea a tal efecto

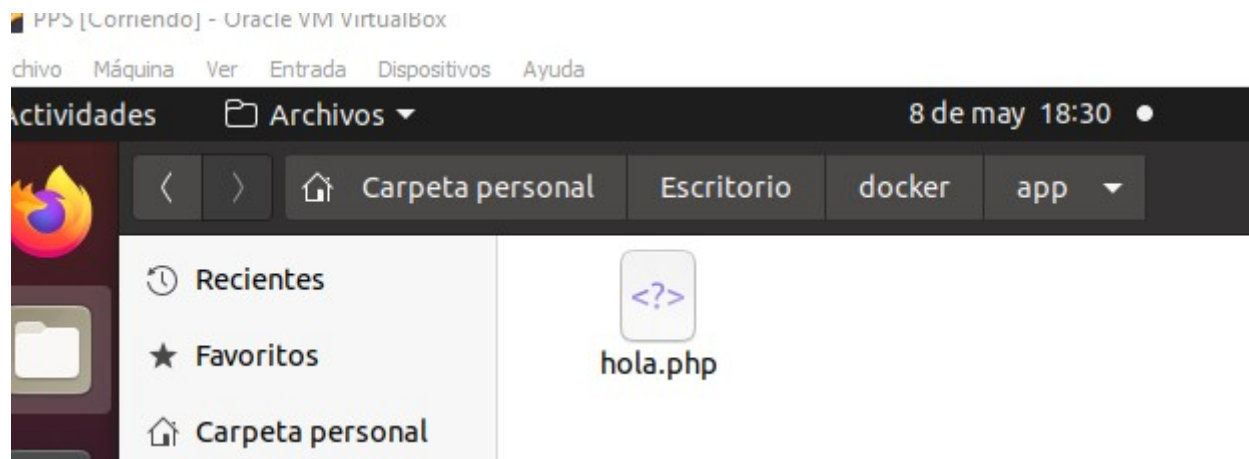
Docker

1. Crea un Dockerfile que partiendo de una imagen PHP genera una imagen que:

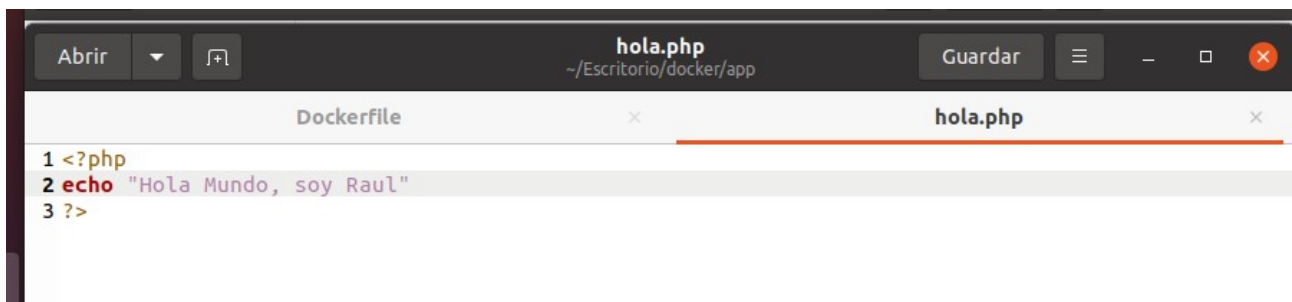
1. Copia una aplicación en PHP a un directorio del contenedor. Esta aplicación se debe copiar directamente desde un directorio del anfitrión. Para facilitar las cosas, debe de ser una aplicación sencilla que no emplee bases de datos (ya que si no también habría que instalar un MySQL).

Primero he creado un directorio con los siguientes ficheros:





Fichero **hola.php**



Configuración fichero Dockerfile:



Se construye el docker:

```
usuario@pps:~/Escritorio/docker$ docker build -t ejercicio2 .
```

```
Sending build context to Docker daemon 3.584kB
```

```
Step 1/5 : FROM php:7.4-apache
```

```
7.4-apache: Pulling from library/php
```

```
a603fa5e3b41: Pull complete
```

```
c428f1a49423: Pull complete
```

```
156740b07ef8: Pull complete
```

```
fb5a4c8af82f: Pull complete
```

```
25f85b498fd5: Pull complete
```

```
9b233e420ac7: Pull complete
```

```
fe42347c4ecf: Pull complete
```

```
d14eb2ed1e17: Pull complete
```

```
66d98f73acb6: Pull complete
```

```
d2c43c5efbc8: Pull complete
```

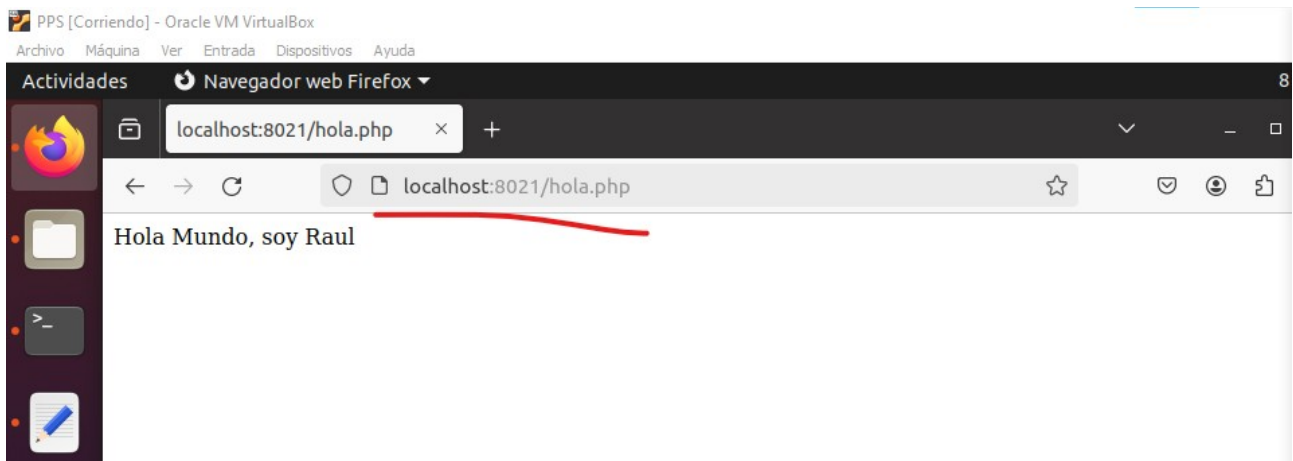
```
ab590b48ea47: Pull complete
```

```
80692ae2d067: Pull complete
```

Se corre el docker en los puertos correspondientes:

```
usuario@pps:~/Escritorio/docker$ docker run -d -p 8021:80 --name perros2 ejercicio2  
56f31667e4626097fba5f6fba585f5e03d4c038c31588bbb6ebf18529e2ba913  
usuario@pps:~/Escritorio/docker$
```

Se entra a un navegador al puerto indicado (8021):

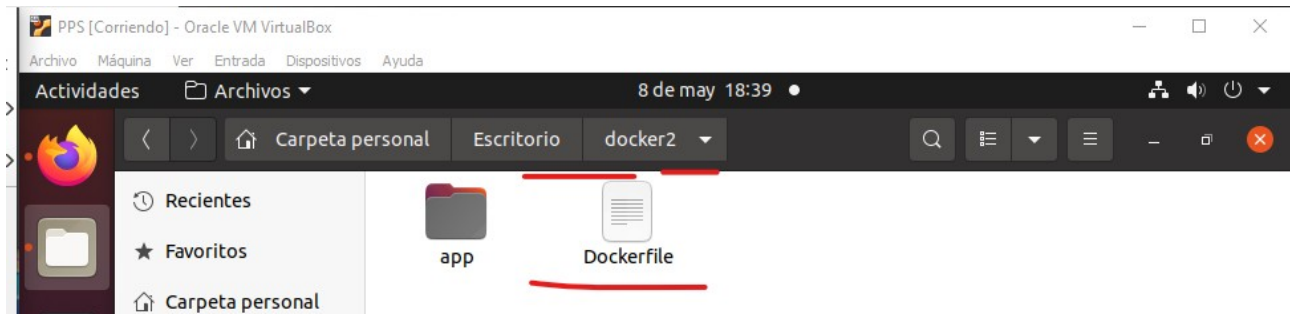


Estaría desplegado el fichero **hola.php**

2. Crea un Dockerfile que partiendo de una imagen Ubuntu genera una imagen que:

1. Instala Apache, de forma que se exponga el puerto 80.
2. Instala PHP.
3. Copia una aplicación web en PHP al directorio de Apache que expone las páginas web. Esta aplicación se debe descargar automáticamente mediante algún comando como git clone o curl. Para facilitar las cosas, debe de ser una aplicación sencilla que no emplee bases de datos (ya que si no también habría que instalar un MySQL).

Trabajaré en un directorio nuevo.



Configuración del Dockerfile:



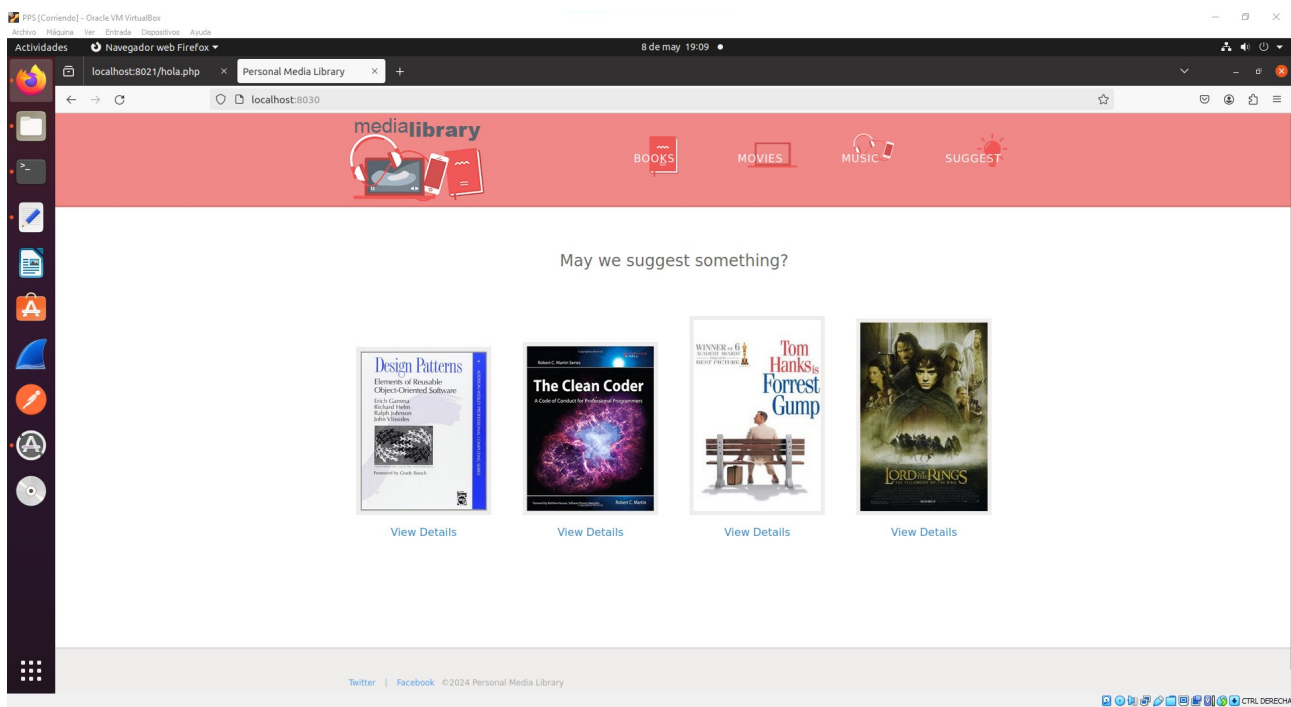
Se construye :

```
usuario@pps:~/Escritorio/docker2$ docker build -t ejercicio7 .
Sending build context to Docker daemon 3.072kB
Step 1/6 : FROM ubuntu:latest
----> bf3dc08bfed0
Step 2/6 : RUN apt-get update && apt-get install -y apache2 php git
----> Using cache
----> 8eb00100e987
Step 3/6 : RUN rm -rf /var/www/html/*
----> Using cache
----> ce52620017c4
Step 4/6 : EXPOSE 80
----> Using cache
----> 51f0cb0af7c0
Step 5/6 : RUN git clone https://github.com/aeimskei/basic-php-website /var/www/html
----> Running in f2585e204ded
Cloning into '/var/www/html'...
Removing intermediate container f2585e204ded
----> ba664d0d9465
Step 6/6 : CMD ["apache2ctl", "-D", "FOREGROUND"]
----> Running in de2167ecdbf7
Removing intermediate container de2167ecdbf7
----> 7d61434fbec5
Successfully built 7d61434fbec5
Successfully tagged ejercicio7:latest
```

Se corre de nuevo en los puertos deseados:

```
usuario@pps:~/Escritorio/docker2$ docker run -d -p 8030:80 --name clonado6 ejercicio7
bea92c645ffd19d5b1e9d2b777006274c1cd2972e6453967bd501ad4ed2b9cd0
usuario@pps:~/Escritorio/docker2$
```

La página será lanzada en ese puerto (8030):



3. Crea un contenedor para cada una de esas imágenes y verifica que funciona. Para y borra dicho contenedor.

Listo los contenedores con **docker ps**: (los clonados son los que funcionaron)

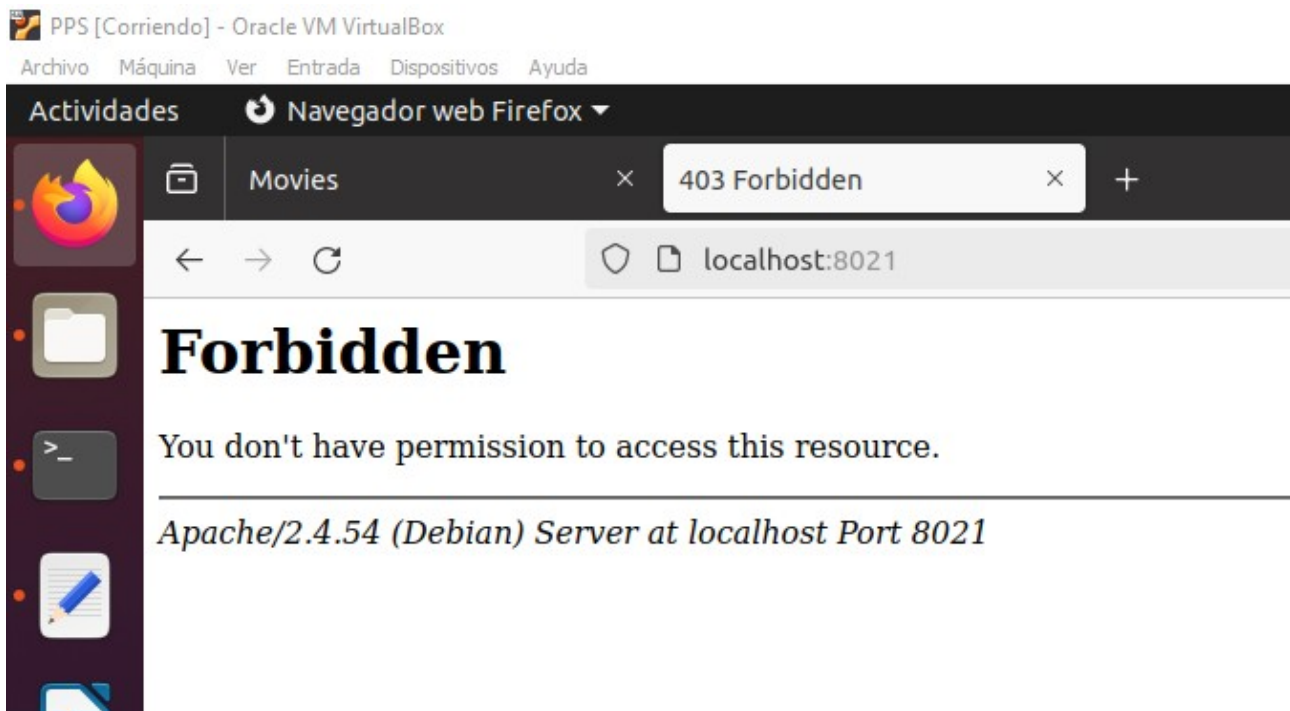
```
usuario@pps:~/Escritorio/docker2$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bea92c645ffd	ejercicio7	"apache2ctl -D FOREG..."	15 minutes ago	Up 15 minutes	0.0.0.0:8030->80/tcp, :::8030->80/tcp	clonado6
07dd417bd56e	ejercicio6	"apache2ctl -D FOREG..."	17 minutes ago	Up 17 minutes	0.0.0.0:8029->80/tcp, :::8029->80/tcp	clonado5
83892f15efa6	ejercicio5	"apache2ctl -D FOREG..."	18 minutes ago	Up 18 minutes	0.0.0.0:8028->80/tcp, :::8028->80/tcp	clonado4
e6e66db04300	ejercicio4	"apache2ctl -D FOREG..."	23 minutes ago	Up 23 minutes	0.0.0.0:8026->80/tcp, :::8026->80/tcp	clonado2
b61bf09581b1	ejercicio3	"apache2ctl -D FOREG..."	29 minutes ago	Up 29 minutes	0.0.0.0:8025->80/tcp, :::8025->80/tcp	clonado
56f31667e462	ejercicio2	"docker-php-entrypoi..."	About an hour ago	Up About an hour	0.0.0.0:8021->80/tcp, :::8021->80/tcp	perros2
3266b4010485	ejercicio1	"docker-php-entrypoi..."	About an hour ago	Up About an hour	0.0.0.0:8020->80/tcp, :::8020->80/tcp	perros

```
usuario@pps:~/Escritorio/docker2$
```

Paro el docker perros (puerto 8021) y compruebo que ya no está activo:

```
usuario@pps:~/Escritorio/docker2$ docker stop perros
perros
usuario@pps:~/Escritorio/docker2$
```



Ahora lo elimino con **docker rm**:

```
usuario@pps:~/Escritorio/docker2$ docker rm perros
perros
usuario@pps:~/Escritorio/docker2$ docker ps
```

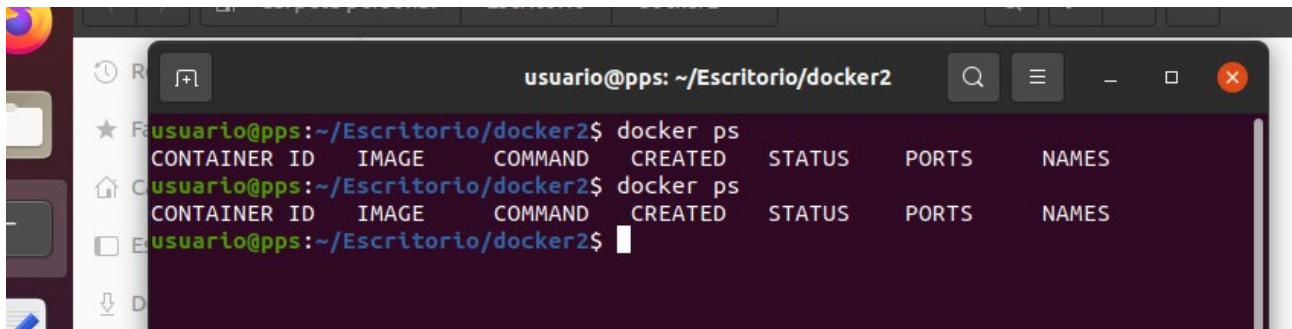
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bea92c645ffd	ejercicio7	"apache2ctl -D FOREG..."	25 minutes ago	Up 25 minutes	0.0.0.0:8030->80/tcp, :::8030->80/tcp	clonado6
07dd417bd56e	ejercicio6	"apache2ctl -D FOREG..."	27 minutes ago	Up 27 minutes	0.0.0.0:8029->80/tcp, :::8029->80/tcp	clonado5
83892f15efa6	ejercicio5	"apache2ctl -D FOREG..."	28 minutes ago	Up 28 minutes	0.0.0.0:8028->80/tcp, :::8028->80/tcp	clonado4
e6e66db04300	ejercicio4	"apache2ctl -D FOREG..."	32 minutes ago	Up 32 minutes	0.0.0.0:8026->80/tcp, :::8026->80/tcp	clonado2
b61bf09581b1	ejercicio3	"apache2ctl -D FOREG..."	39 minutes ago	Up 39 minutes	0.0.0.0:8025->80/tcp, :::8025->80/tcp	clonado
56f31667e462	ejercicio2	"docker-php-entrypoi..."	About an hour ago	Up About an hour	0.0.0.0:8021->80/tcp, :::8021->80/tcp	perros2

```
usuario@pps:~/Escritorio/docker2$
```

Como se observa en la imagen al listar ya no está perros

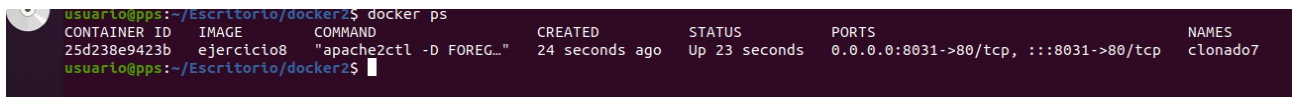
4. Emplea un comando para lanzar 20 contenedores de la segunda imagen, cada uno mapeado en un puerto distinto del anfitrión. Cuando veas que funcionan, para y borra dichos contenedores.

Primero compruebo que no tengo ninguno porque he apagado la máquina:



```
usuario@pps: ~/Escritorio/docker2
usuario@pps:~/Escritorio/docker2$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
usuario@pps:~/Escritorio/docker2$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
usuario@pps:~/Escritorio/docker2$
```

Volví a lanzar uno de los de antes:



```
usuario@pps:~/Escritorio/docker2$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
25d238e9423b   ejercicio8 "apache2ctl -D FOREG..." 24 seconds ago Up 23 seconds 0.0.0.0:8031->80/tcp, :::8031->80/tcp clonado7
usuario@pps:~/Escritorio/docker2$
```

Ahora los lanzo con:

for ((i=0; i<20; i++)); do docker run -d -p \$((8080+\$i)):80 --name contenedor\$i ejercicio8; done

Aquí se muestra como se lanzan los 20:



```
usuario@pps:~/Escritorio/docker2$ for ((i=0; i<20; i++)); do docker run -d -p $((8080+$i)):80 --
name contenedor$i ejercicio8; done
5764ea3985ac550710dcb07be87d9bb0fbf22697ece05cac01a5d2c606d4f84c
c27d3c3b88439192c9a3a7077da52bd30317f2c600207de66fd05811de98bbf
03c122da2fd8e0a14aa41c55c02f0e97e263be065e097486d2cbc2454b0222bc
63f38edb989ca815bca50d62ccc7aacfa3f2332bb5fea2f9da5aa6b266496953
09e86dcc113a2c1f67c6157211e1da35f9e634053a7aedbfa6a6864d2bfb19c
f91d45f0162258b5cd42f5e92bff1cf9a74a69105e24c1e17a6801dec17de781
8d2a61922ae3f50da3f5cf1c18de2989516b0343c9c2503f720ef3a31a987712
cf0e1d53016235a914afc42985d70d1c4261928c6863cec74017dfd1adc19219
857e25fcd01efbe1560cfdd2752e5e5275f8cc00108610cee1bc19f2b66d34d2
0f45f359eb0490a09cf2e9416d1a2921ed16779961c11e58ba362a15cd28c52a
5548b9ef963f67869aa38d8a45da7011f2e66670c711fee36089a8960b59de4d
57ae1b34ffd978d00f2a1b440e9b4fe142628b40cc4d935cbca5ff34549daa63
02dd0910578c3663d63c10ad997ac5da0c1e22631c709dd9f17fca91ce1e20fd
b02f149f5b020538f1ec50c57c598035f7233c2dd7c90ab851a5d5b5f401d4250
62c447fc994f5fb61a683fd57fa63f8540f6e56501eded49c2f017b4218307df
b903112efe0ff2cd43d0aacc12b5ed1178dc416154a9e257d488dce56e5e9973
9543b5c06222426e525307cea192db9ab370ec3e565bcfd919cb17430080e580
78169e7ffc6d8085bd773022d34f08e56fb700c65602a78521866a571c60f1d
74d1805809e9a5a319cce866c89438371f818ae4606ae94dddfc05291b92b7cd
ac9ccdf9d381c7a6848560953fe31b4e7f8b4c65a7a6ae3f818fd2ea0408ed90
usuario@pps:~/Escritorio/docker2$
```


Se muestra aquí que se lanzan del 0 al 19:

```
usuario@pps:~/Escritorio/docker2$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
ac9ccdf9d381   ejercicio8 "apache2ctl -D FOREG..." 49 seconds ago Up 48 seconds 0.0.0.0:8099->80/tcp, :::8099->80/tcp contenedor19
74d1805809e9   ejercicio8 "apache2ctl -D FOREG..." 49 seconds ago Up 48 seconds 0.0.0.0:8098->80/tcp, :::8098->80/tcp contenedor18
78169e7ffc6d   ejercicio8 "apache2ctl -D FOREG..." 49 seconds ago Up 49 seconds 0.0.0.0:8097->80/tcp, :::8097->80/tcp contenedor17
9543b5c06222   ejercicio8 "apache2ctl -D FOREG..." 50 seconds ago Up 49 seconds 0.0.0.0:8096->80/tcp, :::8096->80/tcp contenedor16
b903112efe0f   ejercicio8 "apache2ctl -D FOREG..." 50 seconds ago Up 50 seconds 0.0.0.0:8095->80/tcp, :::8095->80/tcp contenedor15
62c447fc994f   ejercicio8 "apache2ctl -D FOREG..." 51 seconds ago Up 50 seconds 0.0.0.0:8094->80/tcp, :::8094->80/tcp contenedor14
b02f149f5b02   ejercicio8 "apache2ctl -D FOREG..." 51 seconds ago Up 50 seconds 0.0.0.0:8093->80/tcp, :::8093->80/tcp contenedor13
02dd0910578c   ejercicio8 "apache2ctl -D FOREG..." 52 seconds ago Up 51 seconds 0.0.0.0:8092->80/tcp, :::8092->80/tcp contenedor12
57ae1b34ffd9   ejercicio8 "apache2ctl -D FOREG..." 52 seconds ago Up 51 seconds 0.0.0.0:8091->80/tcp, :::8091->80/tcp contenedor11
5548b9ef963f   ejercicio8 "apache2ctl -D FOREG..." 52 seconds ago Up 52 seconds 0.0.0.0:8090->80/tcp, :::8090->80/tcp contenedor10
0f45f359eb04   ejercicio8 "apache2ctl -D FOREG..." 53 seconds ago Up 52 seconds 0.0.0.0:8089->80/tcp, :::8089->80/tcp contenedor9
857e25fcd01e   ejercicio8 "apache2ctl -D FOREG..." 53 seconds ago Up 52 seconds 0.0.0.0:8088->80/tcp, :::8088->80/tcp contenedor8
cf0e1d530162   ejercicio8 "apache2ctl -D FOREG..." 54 seconds ago Up 53 seconds 0.0.0.0:8087->80/tcp, :::8087->80/tcp contenedor7
8d2a61922ae3   ejercicio8 "apache2ctl -D FOREG..." 54 seconds ago Up 53 seconds 0.0.0.0:8086->80/tcp, :::8086->80/tcp contenedor6
f91d45f01622   ejercicio8 "apache2ctl -D FOREG..." 54 seconds ago Up 54 seconds 0.0.0.0:8085->80/tcp, :::8085->80/tcp contenedor5
09e86dccc113a   ejercicio8 "apache2ctl -D FOREG..." 55 seconds ago Up 54 seconds 0.0.0.0:8084->80/tcp, :::8084->80/tcp contenedor4
63f38edb989c   ejercicio8 "apache2ctl -D FOREG..." 55 seconds ago Up 54 seconds 0.0.0.0:8083->80/tcp, :::8083->80/tcp contenedor3
03c122da2fd8   ejercicio8 "apache2ctl -D FOREG..." 56 seconds ago Up 55 seconds 0.0.0.0:8082->80/tcp, :::8082->80/tcp contenedor2
c27d3c3b8843   ejercicio8 "apache2ctl -D FOREG..." 56 seconds ago Up 55 seconds 0.0.0.0:8081->80/tcp, :::8081->80/tcp contenedor1
5764ea3985ac   ejercicio8 "apache2ctl -D FOREG..." 56 seconds ago Up 56 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp contenedor0
25d238e9423b   ejercicio8 "apache2ctl -D FOREG..." 3 minutes ago Up 3 minutes 0.0.0.0:8031->80/tcp, :::8031->80/tcp clonado7
usuario@pps:~/Escritorio/docker2$
```

Los paro reutilizando el comando de antes:

for ((i=0; i<20; i++)); do docker stop contenedor\$i; done

```
usuario@pps:~/Escritorio/docker2$ for ((i=0; i<20; i++)); do docker stop contenedor$i; done
contenedor0
contenedor1
contenedor2
contenedor3
contenedor4
contenedor5
contenedor6
contenedor7
contenedor8
contenedor9
contenedor10
contenedor11
contenedor12
contenedor13
contenedor14
contenedor15
contenedor16
contenedor17
contenedor18
contenedor19
usuario@pps:~/Escritorio/docker2$
```

Una vez pausados todos uso el comando con **rm** para eliminarlos:

for ((i=0; i<20; i++)); do docker rm contenedor\$i; done

```
usuario@pps:~/Escritorio/docker2$ for ((i=0; i<20; i++)); do docker rm contenedor$i; done
contenedor0
contenedor1
contenedor2
contenedor3
contenedor4
contenedor5
contenedor6
contenedor7
contenedor8
contenedor9
contenedor10
contenedor11
contenedor12
contenedor13
contenedor14
contenedor15
contenedor16
contenedor17
contenedor18
contenedor19
usuario@pps:~/Escritorio/docker2$
```

Como vemos ya no están:

```
contenedor19
usuario@pps:~/Escritorio/docker2$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
25d238e9423b   ejercicio8  "apache2ctl -D FOREG..." 12 minutes ago Up 12 minutes  0.0.0.0:8031->80/tcp, :::8031->80/tcp clonado7
usuario@pps:~/Escritorio/docker2$
```

Git

Mantén un repositorio de Git público con:

- Memoria de la práctica.
- El fichero Dockerfile.

Haz un *commit* y un *push* del proyecto cada vez que superes uno de los ítems de la sección anterior.