

Activitat 1. Desplegament d'una web estàtica amb NGinx i Docker

Index

Activitat 1. Desplegament d'una web estàtica amb NGinx i Docker.....	1
1. Introducció.....	1
2. Desplegament de la imatge oficial de Docker NGinx.....	1
2.1 Client de Docker.....	2
2.1.1 Gestió de Contenidors.....	2
2.1.2 Gestió d'imatges.....	3
2.1.3. Arxiu de registre del servidor.....	5
3. Desplegament d'una Web Estàtica.....	5
4. Bibliografia / Webgrafia.....	9

1. Introducció

A fi d'introduir els diferents elements que formen l'arquitectura docker, Durem a terme el desplegament d'una web fent ús d'una imatge base oficial **de nginx**.

Com ja sabem, Nginx és un dels **servidors web** més populars del món i és **responsable d'allotjar alguns dels llocs més grans i amb major trànsit d'Internet**. En la majoria dels casos, té **més recursos que Apache**.

2. Desplegament de la imatge oficial de Docker NGinx

En primer lloc, hem de comprovar que tenim instal·lada la plataforma de 'docker'. En cas contrari seguirem les instruccions de la [pàgina oficial](#).

```
$ docker -v;
```

```
→batoi:~ docker -version
```

```
Docker version 24.0.7, build afdd53b
```

En aquesta activitat farem servir la imatge oficial d'nginx (https://hub.docker.com/_/nginx). Crearem un contenidor sense fer cap modificació sobre la mateixa:

```
$ docker run -dp 8080:80 --name prova-nginx nginx:latest
```

Amb aquest comandament hem indicat a Docker que:

- Cree un contenidor a partir de la imatge **nginx:latest**. Més concretament, la imatge de

nginx etiquetada com `latest`.

- **Redirigeixi** totes les peticions del **port 8080** del **nostre PC** al **port 80** del **contenedor** de docker. (`-dp 8080:80`)
- Que **assigne el nom** `prova-nginx` al **contenedor** creat.

Si obrim un navegador, i accedim a localhost, al port 80, podrem veure la pàgina d'inici de NGinx:



2.1 Client de Docker

2.1.1 Gestió de Contenedors

Si volem **veure** quins **contenedors Docker** **tenim en funcionament**, podem executar el següent comandament:

```
$ docker container ls
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bdcdf9288ed	nginx:latest	"/docker-entrypoint..."	2 minutes ago	Up 2 minutes	0.0.0.0:8080->80/tcp	prova-nginx

Per a **parar un contenidor** podem fer servir el comandament `docker container stop {CONTAINER_ID}` o `docker container stop {NAME}`

```
$ docker container stop bdcdf9288ed
```

Per a **tornar a iniciar un contenidor** que hem **parat** primer haurem de consultar els que tenim

parats en el sistema operatiu i després, **fent servir el id o el nom** executar el següent comandament.

```
$ docker container start nginx
```

```
➤ ~ docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS               NAMES
bdcdf9288ed   nginx:latest   "/docker-entrypoint..." 6 minutes ago   Exited (0)    About a minute ago
```

```
➤ ~ docker container start bdcdf9288ed
bdcdf9288ed
```

```
➤ ~ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS               NAMES
bdcdf9288ed   nginx:latest   "/docker-entrypoint..." 10 minutes ago   Up 3 seconds   0.0.0.0:8080->80/tcp   prova-nginx
```

Per a **eliminar el contenidor del nostre sistema** podem fer servir el següent comandament (Hem de tenir en compte que el contenidor ha d'estar parat):

```
$ docker container rm nginx;
```

```
➤ ~ docker container stop bdcdf9288ed
bdcdf9288ed
```

```
➤ ~ docker container rm bdcdf9288ed
bdcdf9288ed
```

```
➤ ~ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS               NAMES
```

Podem eliminar tots els contenidor que tenim parats amb una única ordre

```
$ docker container prune
```

```
➤ ~ docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
8fe2522a64df7ad24480b847f774c4d28d44ce2a02228de38ea7d3a16cde021a
d9cbe202ebcf0112138eb42afe189bf6dea82b7161506849487e9547fbf183cf
```

2.1.2 Gestió d'Imatges

Com hem vist a la sessió de teoria, **Docker crea els contenidors a partir d'imatges**. Les imatges són una espècie de plantilles que contenen tot el programari que necessita l'aplicació per a posar-se en marxa de forma que tots els contenidors creats a partir d'una imatge

contenen el mateix programari, encara que en el moment de la seua creació es poden personalitzar alguns detalls.

Quan **tractem de llençar un contenidor** a partir d'una imatge, el **dimoni de Docker busca** si eixa **imatge** existeix al **nostre ordinador**, en cas contrari la buscarà al repositori oficial i la baixarà. D'aquesta forma, al igual que hem fet amb els contenidors

Podeu obtindre la referència als diferents comandaments a la documentació oficial: <https://docs.docker.com/engine/reference/commandline/docker/>

<code>docker image ls</code>	Mostra totes les imatges que tenim al nostre ordinador
<code>docker image rm {REPOSITORY}</code> <code>docker image rm {IMAGE_ID}</code>	Muestra las imágenes descargadas y disponibles en nuestra máquina

```

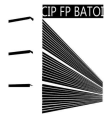
$ docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest a8758716bb6a 2 months ago 187MB
hello-world latest d2c94e258dcb 8 months ago 13.3kB
$ docker image rm d2c94e258dcb
Untagged: hello-world:latest
Untagged: hello-world@sha256:4bd78111b6914a99dbc560e6a20eab57ff6655aea4a80c50b0c5491968cbc2e6
Deleted: sha256:d2c94e258dcb3c5ac2798d32e1249e42ef01cba4841c2234249495f87264ac5a
Deleted: sha256:ac28800ec8bb38d5c35b49d45a6ac4777544941199075dff8c4eb63e093aa81e
$ docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest a8758716bb6a 2 months ago 187MB

```

Activitat 1

Contesta breument i amb les teues paraules a les següents **qüestions**:

- 1.1.- Explica com funciona el paràmetre run de Docker executat al punt 2.
- 1.2.- Crea un nou contenidor a partir de la **imatge base nginx** que estiga a la escolta del **port 8090** del teu PC, i amb el nom '**nginx-8090**'. Accedeix amb el navegador i comprova que funciona correctament.
- 1.3.- Fes una captura de pantalla dels contenidors Docker que tens en execució a la teua màquina.
- 1.4.- Para el contenidor docker y fes una captura de pantalla de tots els contenidors (incloent els que tens parats).
- 1.5.- Elimina de forma definitiva el contenidor amb nom '**nginx-8090**'. Pega un captura en la



que demostres que s'ha eliminat correctament.

1.6.- Fes una captura de totes les imatges que tens al servidor i elimina la de nginx que has utilitzat al punt anterior.

2.1.3. Arxiu de registre del servidor

Per a poder accedir als logs de Nginx, Docker ens proporciona el comandament **docker container logs {id o nom del contenidor}** :

```
$ docker container logs prova-nginx;
```

```
* ~ docker container logs prova-nginx;
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
192.168.65.1 - - [18/Jan/2024:11:37:10 +0000] "GET /favicon.ico HTTP/1.1" 404 153 "http://127.0.0.1:8080/" "Mozilla/5.0 (Macintosh; Intel
Mac OS X 10.15; rv:121.0) Gecko/20100101 Firefox/121.0" "-"
2024/01/18 11:37:10 [error] 29#29: *1 open() "/usr/share/nginx/html/favicon.ico" failed (2: No such file or directory), client: 192.168.6
5.1, server: localhost, request: "GET /favicon.ico HTTP/1.1", host: "127.0.0.1:8080", referrer: "http://127.0.0.1:8080/"
```

Activitat 2

Llença un nou contenidor amb la imatge nginx i realitza les següents accions:

2.1.- Accedeix diverses vegades a la pàgina d'inici del servidor NGinx desplegat amb Docker, fent servir diferents navegadors, i posteriorment visualitza els logs del contenidor. Fes una captura de pantalla.

2.2.- ¿Des de quin navegador web s'ha visitat la pàgina d'inici? Relaciona cadascuna de les entrades del log amb el navegador corresponent.

3. Desplegament d'una Web Estàtica

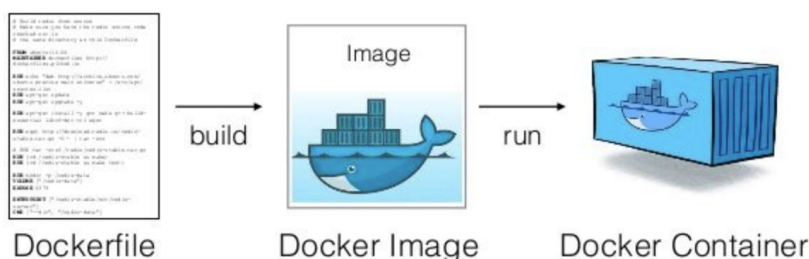
Fins ara hem vist com crear contenidor amb un servidor nginx, com gestionar-los i com accedir a ells. No obstant això, el que necessitem és poder desplegar les nostres web en un contenidor.

Per fer-ho haurem de **crear una nova imatge a partir de la imatge que conté el servidor web nginx en la que s'incloga el web que volem desplegar**. La construcció d'imatges es realitza amb un arxiu anomenat **Dockerfile**. Aquest arxiu ens permet **afegir noves capes a una imatge base**. Cada una d'aquestes capes són accions que volem que es facen sobre la

imatge la base, Exemples d'aquestes accions són:

- **Copiar una arxiu o carpeta al contenidor.** En el nostre cas, tendrem que copiar la web al document root del contenidor.
- Instal·lar algun mòdul o servei.
- Modificar un arxiu de configuració.
- Etc.

Una vegada tenim el fitxer Dockerfile construirem la nostra imatge i li assignarem un nom per a tindre-la disponible per a llençar tots els contenidor que necessitem.







Construcció d'una imatge pròpia amb el web a desplegar

Si observem la documentació oficial de la imatge docker de Nginx podem veure quin és el **DocumentRoot** que utilitza per a desplegar el web i que tenim a la ruta **/usr/share/nginx/html**.

Per tant, per a desplegar la nostra web al servidor NGinx en un contenidor a partir de dita imatge de copiar els fitxers de la nostra web a la carpeta **/usr/share/nginx/html** abans de que es llanci.

Encara que trobem diverses opcions per a desplegar la nostra web al contenidor, en aquesta activitat hem optat per **afegir una capa** a la imatge de nginx.

En primer lloc, crearem una carpeta ud06a01, i dins, [clonarem el projecte de la nostra web](#). També crearem un document de text amb el nom 'Dockerfile'

Nombre	Estado	Fecha de modificación	Tipo	Tamaño
 ddaw-ud1-p1		27/09/2022 17:31	Carpeta de archivos	
 Dockerfile		27/09/2022 17:43	Archivo	1 KB

A continuació, editem el fitxer Dockerfile, afegint el següent contingut:

```
FROM /nginx:latest
COPY ddaw-ud1-p1 /usr/share/nginx/html
```

En el fitxer Dockerfile hem indicat que, partint de la imatge Docker de nginx amb la etiqueta latest, volem copiar la carpeta `ddaw-ud1-p1` del nostre maquinari a la carpeta `/usr/share/nginx/html` del contenidor.

Guardem el document Dockerfile, i executem el següent comandament, a un terminal posicionat a la mateixa carpeta en la que es troba el fitxer Dockerfile:

```
$ docker build -t ud06a01:v1 .
```

```
→ ud06a01 docker build -t ud06a01:v1 .

[+] Building 0.1s (7/7) FINISHED
docker:desktop-linux
=> [internal] load .dockerignore
0.0s
=> => transferring context: 2B
0.0s
=> [internal] load build definition from Dockerfile
0.0s
=> => transferring dockerfile: 94B
0.0s
=> [internal] load metadata for docker.io/library/nginx:latest
0.0s
=> [internal] load build context
0.0s
=> => transferring context: 2.99kB
0.0s
=> [1/2] FROM docker.io/library/nginx:latest
0.0s
=> CACHED [2/2] COPY ddaw-ud1-p1 /usr/share/nginx/html
0.0s
=> exporting to image
0.0s
```

```
=> => exporting layers
0.0s
=> => writing image
sha256:612c6368e43aad49f0203b50ca8fcb9fa2969e593933c9a7ba01753257a96379
0.0s
=> => naming to docker.io/library/ud06a01:v1
```

Després de construir la nostra imatge personalitzada, tindrem en el nostre maquinari una nova imatge amb l'etiqueta ud06a01 amb la versió v1. (Aquesta nova imatge sols és accessible des del nostre maquinari.

```
ud06a01 docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
ud06a01 v1 612c6368e43a 2 hours ago 189MB
nginx latest a8758716bb6a 2 months ago 187MB
```



Per poder-la fer servir a un altre maquinari deuríem de pujar-la a un docker registri. Aquesta part la aprendrem més endavant.

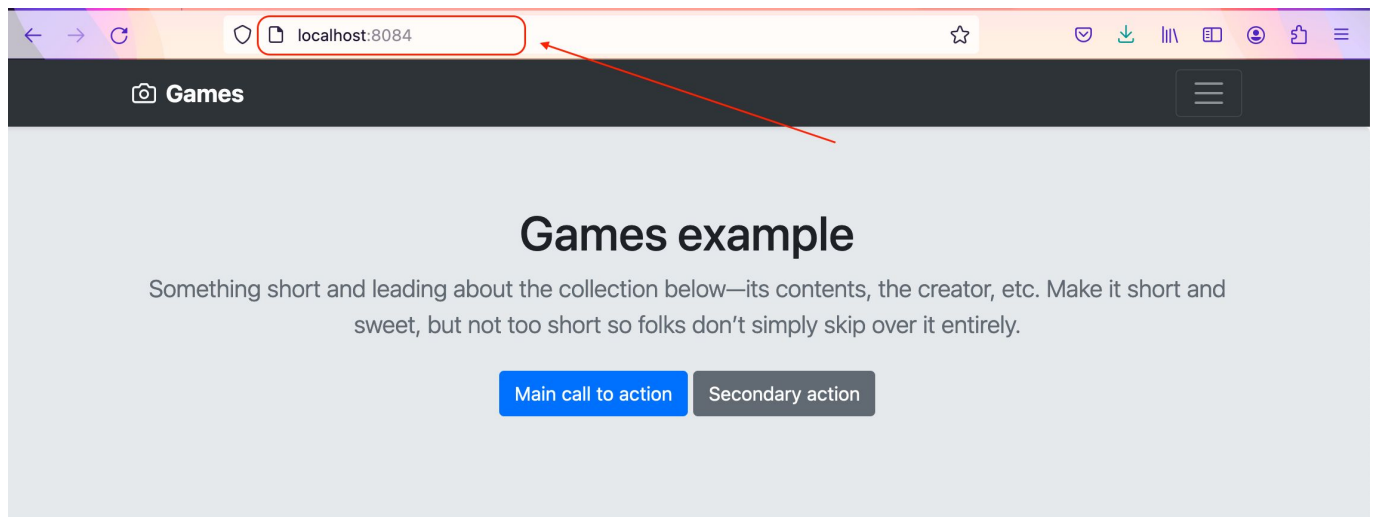
Instanciació d'un contenidor amb l'imatge creada.

Per finalitzar, arrancarem un contenidor amb la nostra imatge, executant el següent comandament:

```
$ docker run -dp 8084:80 --name ud06a01 ud06a01:v1
```

```
ud06a01 docker container ls
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
40233bc91cb9 ud06a01:v1 "/docker-entrypoint..." 11 seconds ago Up 10 seconds 80/tcp, 0.0.0.0:80->8080/tcp ud01a06
```

Podrem observar com s'ha instanciat un nou contenidor amb la nostra imatge, mapejan el port 80 del contenidor al port 8084 del nostre maquinari. D'aquesta forma, si accedim al port 8082 del nostre maquinari amb el navegador, podrem accedir a la web desplegada:



Activitat 3.- Publicació d'una web estàtica

3.1 Crea una imatge docker personalitzada a partir de la imatge docker `nginx:latest` que inclogui la pàgina web estàtica ubicada en el [següent repositori](#). La imatge Docker deu crear-se amb la etiqueta `ud06a01-x-y`, a on **X** e **Y** seran els noms dels integrants del grup. Per finalitzar, desplega dos contenidors a partir de l'imatge Docker creada (recorda que han de ser diferents ports) i accedeix a ella des del host amfitrió. Per demostrar que has fet correctament el desplegament de la web hauràs de (com a mínim) adjuntar captures de pantalla que demostrin:

- El contingut del Dockerfile
- El comandament utilitzat per crear la imatge personalitzada.
- El comandament utilitzat per desplegar el contenidor a partir de la imatge personalitzada.
- Una captura del navegador, mostrant la web desplegada i l'URL per a cada contenidor
- Una captura de l'accés log del contenidor, a on puguem veure l'accés a la web de cada contenidor

4. Bibliografia / Webgrafia

- Documentació de la imatge docker Nginx. https://hub.docker.com/_/nginx
- Docker cli reference. <https://docs.docker.com/engine/reference/run/>