

ACTIVIDAD EVALUABLE 3 - GIT Y DOCKER

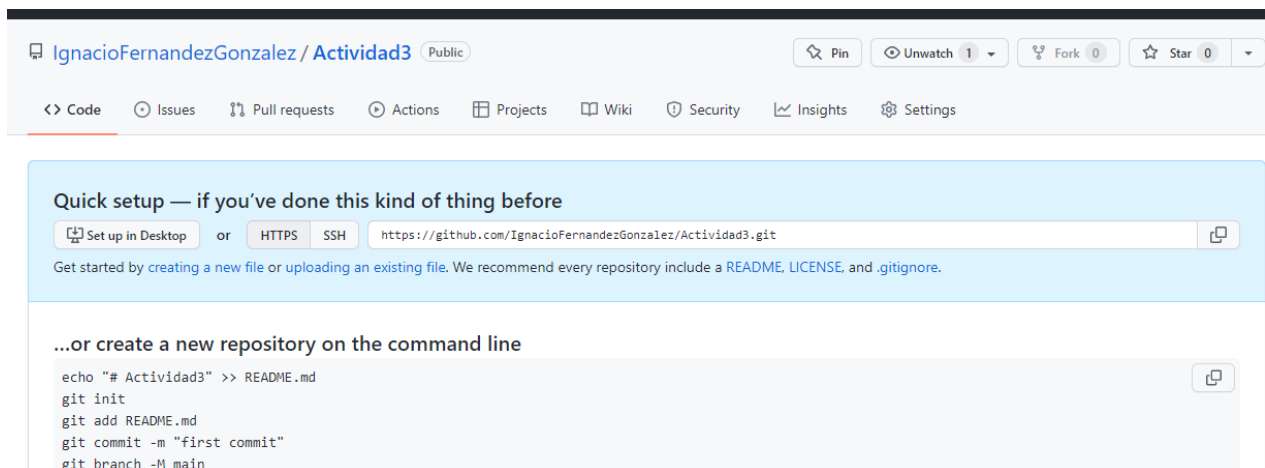
ACTIVIDAD EVALUABLE 3 - GIT Y DOCKER

INTRODUCCION

Ejercicio 2 - almacenamiento

INTRODUCCION

Para empezar a organizar la entrega de la tarea se va a crear un repositorio GitHub ([\[https://github.com/IgnacioFernandezGonzalez/Actividad3\]](https://github.com/IgnacioFernandezGonzalez/Actividad3):
) para alojar las distintas soluciones de los ejercicios que componen la tarea.



A continuación he generado un directorio en mi disco local para alojar mis ficheros y poder trasladar los cambios a GitHub

Ejercicio 2 - almacenamiento

Para la resolución de este segundo ejercicio lo primero que tenemos que crear es una carpeta "saludo", yo la haré en el directorio /ejercicio2/saludo

```
mkdir saludo
```

Una vez estemos en el directorio creado, crearemos un fichero `index.html` con el siguiente contenido

```
ignaciofernandez@clienteLinux: ~/ejercicio2/salud
GNU nano 4.8 index.html
<h1>HOLA SOY IGNACIO FERNANDEZ GONZALEZ</h1>
```

Como siempre usando el editor nano, con el comando

```
sudo nano index.html
```

El siguiente paso será arrancar uno de los contenedores que se nos pide "c1", haciendo un bind mount de la carpeta saludo en la carpeta `/var/html/www` de dicho contenedor, y haciendo que podamos acceder a el contenido por el puerto 8181

Lo haremos con el comando

```
docker container run --name c1 -v /ejercicio2/saludo:/var/www/html --publish 8181:80 --detach --restart=always php:7.4-apache
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ sudo docker container run --name c1 -v ~/ejercicio2/saludo:/var/www/html --publish 8181:80 --detach --restart=always php:7.4-apache
[sudo] contraseña para ignaciofernandez:
ed723b7961d155bc333446a2caf93366d1df9077a5e4550dfccf509574c39335
```

Vemos que una vez accedimos al contenedor c1, podemos visualizar el contenido de `index.html`

```
ignaciofernandez@clienteLinux: ~/ejercicio2/salud
GNU nano 4.8 index.html
<h1>HOLA SOY IGNACIO FERNANDEZ GONZALEZ</h1>
```

Una vez montado podemos acceder por el navegador a nuestro fichero .html en el puerto 8181



Para continuar, arrancaremos otro contenedor de nombre "c2", haciendo un bind mount de la carpeta saludo en la carpeta `/var/html/www` de dicho contenedor, y haciendo que podamos acceder a el contenido por el puerto 8282

Lo haremos con el comando

```
docker container run --name c2 -v /ejercicio2/saludo:/var/www/html --publish 8282:80 --detach --restart=always php:7.4-apache
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ sudo docker container run --name c2 -v ~/ejercicio2/saludo:/var/www/html --publish 8282:80 --detach --restart=always php:7.4-apache
a102d535e411e369976b3e189ecd5af9e58051357106521d43e39c7431dee638
```

Ahora con el contenedor "c2" arrancado modificaremos el contenido del fichero `index.html`, vemos que se puede seguir accediendo a `index.html` con el contenedor c2 arrancado.

```
ignaciofernandez@clienteLinux: ~/ejercicio2/salud
GNU nano 4.8 index.html
<h1>HOLA SOY IGNACIO FERNANDEZ GONZALEZ y he arrancado el contenedor c2</h1>
```

Vamos a comprobar que efectivamente tenemos acceso a este index.html por el puerto 8282



A continuación vamos a comprobar que podemos seguir accediendo a los dos contenedores, sin necesidad de reiniciarlos

Primero comprobamos que están creados y en funcionamiento, con el comando

```
docker ps
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
a182d535e411   php:7.4-apache "docker-php-entrypoi..." 31 minutes ago Up 31 minut
es            c2
ed723b7961d1   php:7.4-apache "docker-php-entrypoi..." 38 minutes ago Up 38 minut
es            c1
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$
```

Efectivamente están en funcionamiento, y ahora vamos a acceder a cada uno de ellos con el comando

```
docker exec -i -t c1 /bin/bash
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker exec -i -t c1 /bin/bash
root@ed723b7961d1:/var/www/html#
```

Vemos que efectivamente accedemos al shell del contenedor c1

Ahora haremos lo mismo con el contenedor c2

```
docker exec -i -t c2 /bin/bash
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker exec -i -t c2 /bin/bash
root@a182d535e411:/var/www/html#
```

Como paso final borraremos ambos contenedores, para lo que antes tendremos que pararlos con el comando

```
docker stop c1 c2
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker stop c1 c2
c1
c2
```

Para a continuación eliminarlos con el comando

```
docker rm c1 c2
```

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker rm c1 c2
c1
c2
```

Si hacemos

```
docker ps -a
```

Vemos que ya no existen estos contenedores

```
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED
STATUS        PORTS
71a178510300   ejemplol:v1   "/usr/sbin/apache2ct..." 3 days ago
Exited (255)   80/tcp, 0.0.0.0:49153->8282/tcp, :::49153->8282/tcp
plo_v2
fec9da107175   ejemplol:v1   "/usr/sbin/apache2ct..." 3 days ago
Exited (255)   0.0.0.0:80->80/tcp, :::80->80/tcp
ced472a16180   ubuntu-python3 "python3 hola.py"         3 days ago
Exited (0)
y_agnesi
276eb359fc0e   ignaciofernandez/ubuntu-nodejs "bash"                     7 days ago
Exited (0)
cray
6780be5867e7   ubuntu        "bash"                     7 days ago
Exited (0)
ing_cannon
96a211f1c9f7   hello-world   "/hello"                   7 days ago
Exited (0)
ated_aryabhata
ignaciofernandez@clienteLinux:~/ejercicio2/saludo$
```