
«Dockerizar» una web estática y publicarla en Docker Hub

Implantación de Aplicaciones Web

Índice general

1	«Dockerizar» una web estática y publicarla en Docker Hub	1
1.1	Conocimientos previos	1
1.2	Tareas a realizar	1
1.3	Requisitos del archivo Dockerfile	1
1.4	Creación de la imagen Docker a partir del archivo Dockerfile	2
1.5	Publicar la imagen en Docker Hub	3
1.6	Publicar la imagen automáticamente en Docker Hub con GitHub Actions	3
1.7	Entregables	4
2	Referencias	5
3	Licencia	6

1 «Dockerizar» una web estática y publicarla en Docker Hub

En esta práctica tendremos que crear un archivo [Dockerfile](#) para crear una imagen [Docker](#) que contenga una **aplicación web estática**. Posteriormente deberá publicar la imagen en [Docker Hub](#) y realizar la implantación del sitio web en [Amazon Web Services \(AWS\)](#) haciendo uso de contenedores [Docker](#) y de la herramienta [Docker Compose](#).

1.1. Conocimientos previos

Se recomienda la lectura de la sección [Dockerfile](#) del siguiente tutorial de [introducción a Docker](#).

1.2. Tareas a realizar

A continuación se describen **muy brevemente** algunas de las tareas que tendrá que realizar.

1. Crea un archivo [Dockerfile](#) para crear una imagen que contenga el servicio de **Nginx** con la siguiente aplicación web estática:
 - <https://github.com/josejuansanchez/2048>
2. Publica la imagen en [Docker Hub](#).
3. Crea una máquina virtual [Amazon EC2](#).
4. Instala y configura [Docker](#) y [Docker compose](#) en la máquina virtual.
5. Crea un archivo **docker-compose.yml** para poder desplegar la aplicación web estática en la máquina virtual de AWS.
6. Busque cuál es la dirección IP pública de su instancia y compruebe que puede acceder a la aplicación web desde un navegador web.

1.3. Requisitos del archivo Dockerfile

Tendrá que crear un archivo [Dockerfile](#) con los siguientes requisitos:

- Como imagen base deberá utilizar la última versión de `ubuntu`.
- Instala el software necesario para poder clonar el repositorio de GitHub donde se encuentra la aplicación web estática.
- Clona el repositorio de GitHub donde se encuentra la aplicación web estática en el directorio `/var/www/html/`, que es el directorio que utiliza Nginx, por defecto, para servir el contenido.
- El puerto que usará la imagen para ejecutar el servicio de Nginx será el puerto 80.
- El comando que se ejecutará al iniciar el contenedor será el comando `CMD ["nginx", "-g", "daemon off;"]`.

1.4. Creación de la imagen Docker a partir del archivo Dockerfile



Para crear la imagen de Docker a partir del archivo `Dockerfile` deberá ejecutar el siguiente comando.

```
1 docker build -t nginx-2048 .
```

Para comprobar que la imagen se ha creado correctamente podemos ejecutar el comando:

```
1 docker images
```

Para publicar la imagen en [Docker Hub](#) es necesario que en el nombre de la imagen aparezca **nuestro nombre de usuario de Docker Hub**. Por ejemplo, si mi nombre de usuario es `josejuansanchez` la imagen debería llamarse `josejuansanchez/nginx-2048`.

También es una buena práctica asignarle una etiqueta a la imagen. Por ejemplo, en este caso vamos a asignarle las etiquetas `1.0` y `latest`.

```
1 docker tag nginx-2048 josejuansanchez/nginx-2048:1.0
```

```
1 docker tag nginx-2048 josejuansanchez/nginx-2048:latest
```

Comprobamos que la imagen tiene el nombre y las etiquetas correctas:

```
1 docker images
```

1.5. Publicar la imagen en Docker Hub



Una vez que le hemos asignado un nombre correcto a la imagen y le hemos añadido las etiquetas, podemos publicarla en [Docker Hub](#).

En primer lugar, tendremos que iniciar la sesión en [Docker Hub](#) con el comando:

```
1 docker login
```

Para iniciar la sesión nos preguntará el nombre de usuario y la contraseña de [Docker Hub](#). En lugar de introducir nuestra contraseña podemos crear un token en [Docker Hub](#) y utilizarlo para iniciar la sesión.

Una vez iniciada la sesión, podemos publicar la imagen con el comando `docker push`. Tenemos que publicar la imagen con las dos etiquetas que hemos creado.

```
1 docker push josejuansanchez/nginx-2048:1.0
```

```
1 docker push josejuansanchez/nginx-2048:latest
```

1.6. Publicar la imagen automáticamente en Docker Hub con GitHub Actions

En este apartado vamos a aprender cómo podemos configurar [GitHub Actions](#) para publicar una imagen automáticamente en un Registry como Docker Hub, cada vez que se realice un *push* al repositorio de GitHub.

Se recomienda la lectura del apartado «[Publicación de imágenes de Docker](#)» de la documentación oficial de GitHub Actions.

Puede encontrar un ejemplo de cómo se puede configurar GitHub Actions para publicar una imagen de Docker en el siguiente repositorio de GitHub:

- <https://github.com/josejuansanchez/2048-github-actions>

Para utilizar este ejemplo, deberá crear dos *secrets* en su repositorio para las acciones de GitHub Actions. Estos *secrets* almacenarán los siguientes valores:

- `DOCKERHUB_USERNAME`: Nombre de usuario en Docker Hub.
- `DOCKERHUB_TOKEN`: Token de acceso a Docker Hub, que tendrá que crear en la sección de *Security* de su cuenta de Docker Hub.

1.7. Entregables

En esta práctica habrá que entregar un **documento técnico** con la descripción de los pasos que se han llevado a cabo durante todo el proceso.

El documento debe incluir **como mínimo** lo siguientes contenidos:

- URL del repositorio de GitHub donde se ha alojado el documento técnico escrito en [Markdown](#).
- Descripción detallada de las acciones que ha realizado durante durante el desarrollo de esta práctica.

2 Referencias

- [Docker](#)
- [Docker Compose](#)
- [Dockerfile Reference](#)

3 Licencia

Esta página forma parte del curso Implantación de Aplicaciones Web de José Juan Sánchez Hernández y su contenido se distribuye bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.