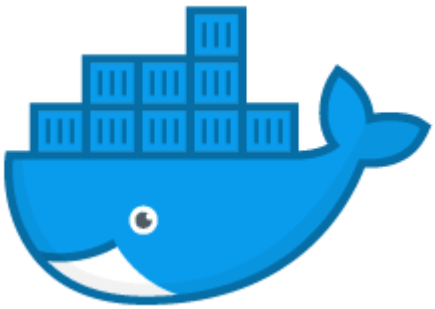
Nimatron Studios

Presenta

Docker para principiantes



Docker es un software gratuito que permite la creación aislada e independiente de entornos de desarrollo para implementar y desplegar aplicaciones. Facilitando enormemente la ejecución de proyectos en diferentes maquinas sin problemas de compatibilidad. Los entornos de desarrollo se denominan Contenedores. Docker simplifica y acelera el flujo de trabajo del desarrollo con un pipeline integrado y mediante consolidación de componentes por aplicación.

La función de los contenedores Docker es la de ejecutar instancias de imágenes Docker. Mediante las imágenes se comparte una aplicación o servicios con absolutamente todas las dependencias en diferentes entornos. Permitiendo de esta forma que el desarrollador tenga la capacidad de implementar rápidamente cualquier aplicación y tener más control sobre el versionado.

Recomiendo ingresar a su web y conocer un poco más: Docker

¿Cómo se utiliza?

Para responder a esta pregunta crearemos un programa simple en Python. La facilidad que ofrece Docker es que no será necesario tener Python instalado en tu computadora para que funcione.

Primero necesitamos instalar Docker,

Si utilizas Windows puedes hacerlo dando clic en el siguiente enlace.

Instalar en Windows

Si utilizas MacOS puedes hacerlo dando clic en el siguiente enlace.

Instalar en MacOS

Si utilizas Ubuntu puedes hacerlo ejecutando la siguiente línea de código en consola:

```
$ sudo apt install docker.io
```

Para comprobar su correcta instalación ejecutamos:

```
$ sudo docker run hola-mundo
```

Nota: Para este tutorial utilizaremos Windows y VScode

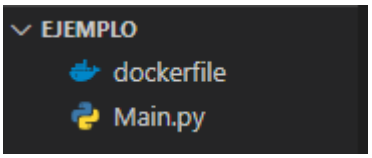
Creamos la carpeta que contendrá nuestro pequeño programa en Python.

Crearemos dos archivos:

Main.py

dockerfile

(Dockerfile es un archivo de texto plano donde guardaremos las instrucciones para crear nuestras imágenes Docker)



Dentro del archivo Main.py escribiremos la siguiente línea de código:

```
print("Mis primeros pasos con Docker")
```

Ahora llegó el momento de configurar nuestro Dockerfile, en este caso necesitamos simplemente que pueda ejecutar nuestro pequeño programa, por lo que solo necesitará tener Python instalado.

Docker nos facilita mucho más las cosas, existen imágenes ya configuradas para la ejecución de proyectos, todo esto podemos encontrarlo en el siguiente enlace: Docker Hub

Dentro buscaremos la imagen para ejecutar un proyecto de Python.

Nuestro dockerfile debe quedar de la siguiente manera:

```
dockerfile > ...
1  # A dockerfile must always start by importing the base image.
2  # We use the keyword 'FROM' to do that.
3  # In our example, we want import the python image.
4  # So we write 'python' for the image name and 'latest' for the version.
5  FROM python:latest
6
7  # In order to launch our python code, we must import it into our image.
8  # We use the keyword 'COPY' to do that.
9  # The first parameter 'main.py' is the name of the file on the host.
10 # The second parameter '/' is the path where to put the file on the image.
11 # Here we put the file at the image root folder.
12 COPY Main.py /
13
14 # We need to define the command to launch when we are going to run the image.
15 # We use the keyword 'CMD' to do that.
16 # The following command will execute "python ./main.py".
17 CMD [ "python", "./Main.py" ]
```

Ahora crearemos nuestra imagen Docker a partir del dockerfile, entonces en consola ejecutamos:

docker build -t python-app

```
[+] Building 71.6s (7/7) FINISHED
 => [internal] load build definition from Dockerfile                                    0.0s
 => => transferring dockerfile: 826B                                                    0.0s
 => [internal] load .dockerignore                                                       0.0s
 => => transferring context: 2B                                                         0.0s
 => [internal] load metadata for docker.io/library/python:latest                        7.9s
 => [internal] load build context                                                       0.0s
 => => transferring context: 71B                                                        0.0s
 => [1/2] FROM docker.io/library/python:latest@sha256:3d3a5f478d2ba0e025e78e84c1dd468cff4d2f99a12d79  61.1s
 => => resolve docker.io/library/python:latest@sha256:3d3a5f478d2ba0e025e78e84c1dd468cff4d2f99a12d79d  0.0s
 => => sha256:bd8f6a7501ccbe80b95c82519ed6fd4f7236a41e0ae59ba4a8df76af24629efc 50.43MB / 50.43MB  20.7s
 => => sha256:efe9738af0cb2184ee8f3fb3dcb130455385aa428a27d14e1e07a5587ff16e64 10.00MB / 10.00MB  10.1s
 => => sha256:2c3f921940344546a5695d11a8118998f011de6e38d36f4edd99253330429d59 2.22kB / 2.22kB  0.0s
 => => sha256:a57971b4d22cd930652c246ba56311a6a8cc10ca9d153fb4e803f12dfc48c02a 8.34kB / 8.34kB  0.0s
 => => sha256:44718e6d535d365250316b02459f98a1b0fa490158cc53057d18858507504d60 7.83MB / 7.83MB  9.0s
 => => sha256:3d3a5f478d2ba0e025e78e84c1dd468cff4d2f99a12d79d37a89294fc6d7ae41 2.36kB / 2.36kB  0.0s
 => => sha256:f37aabde37b87d272286df45e6a3145b0884b72e07e657bf1a2a1e74a92c6172 51.84MB / 51.84MB  28.8s
 => => sha256:3923d444ed0552ce73ef51fa235f1b45edafdec096abda6abab710637dac7ec6 192.35MB / 192.35MB  50.5s
 => => extracting sha256:bd8f6a7501ccbe80b95c82519ed6fd4f7236a41e0ae59ba4a8df76af24629efc  2.5s
 => => sha256:1ecef690e281c31d68c3cd0628207e8b02fb0e60575604fc5436404d1007a75e 6.15MB / 6.15MB  23.0s
 => => sha256:0649c5bd98518385dfa6b98f398c409f54f3c2744297d7cab30b03ecba9fcfb5 19.15MB / 19.15MB  28.8s
 => => extracting sha256:44718e6d535d365250316b02459f98a1b0fa490158cc53057d18858507504d60  0.3s
 => => extracting sha256:efe9738af0cb2184ee8f3fb3dcb130455385aa428a27d14e1e07a5587ff16e64  0.3s
 => => sha256:d7a9fa72f19290251b0f5df9144f9da3b63189e321d4b0f330344b464f1168c8 233B / 233B  29.4s
 => => extracting sha256:f37aabde37b87d272286df45e6a3145b0884b72e07e657bf1a2a1e74a92c6172  2.6s
 => => sha256:9de7a5c126e61030af7b1e277797a4daf67bb8dd9fedd0622f52baa176a518e0 2.31MB / 2.31MB  30.2s
 => => extracting sha256:3923d444ed0552ce73ef51fa235f1b45edafdec096abda6abab710637dac7ec6  8.5s
 => => extracting sha256:1ecef690e281c31d68c3cd0628207e8b02fb0e60575604fc5436404d1007a75e  0.3s
 => => extracting sha256:0649c5bd98518385dfa6b98f398c409f54f3c2744297d7cab30b03ecba9fcfb5  0.7s
 => => extracting sha256:d7a9fa72f19290251b0f5df9144f9da3b63189e321d4b0f330344b464f1168c8  0.0s
 => => extracting sha256:9de7a5c126e61030af7b1e277797a4daf67bb8dd9fedd0622f52baa176a518e0  0.2s
 => [2/2] COPY Main.py /                                                                2.4s
 => exporting to image                                                                  0.1s
 => => exporting layers                                                                 0.0s
 => => writing image sha256:d525fe9fb4f92c915f78fa752f860cf266124d3d17061245367924fce5f3816e  0.0s
 => => naming to docker.io/library/python-app                                           0.0s
```

Al concluir podremos ejecutar nuestra imagen Docker con:

```
docker run python-app
```

Podemos observar que nuestro programa se ejecuta sin inconvenientes.

Conclusión

Docker es super útil para todo tipo de desarrollador de software. Las facilidades que ofrece para las diferentes necesidades y requerimientos permiten acelerar de forma exponencial el flujo de trabajo.

Extras

Comandos más utilizados:

- Mostrar los registros de un contenedor

  ```
  docker logs [NombreContenedor]
  ```

- Estadísticas de un contenedor (uso de CPU, memoria, etc).

  ```
  docker stats
  ```

- Cancelar un contendor Docker

  ```
  docker kill [NombreContenedor]
  ```

- Eliminar todos los contenedores detenidos

  ```
  docker rm $(docker ps -a -q)
  ```

- Eliminar contenedor especifico

  ```
  docker rm [NombreContenedor]
  ```

- Detener todos los contenedores

  ```
  docker stop $(docker ps -a -q)
  ```

- Detener contenedor especifico

  ```
  docker stop [NombreContenedor]
  ```

- Contenedores existentes

  docker ps -a

- Eliminar todas las imágenes

  docker image rm $(docker images -a -q)

- Eliminar imagen especifica

  docker image rm [NombreImagen]

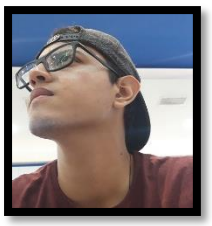
- Enumerar imágenes

  docker image ls

- Proceso superior de un contenedor

  docker top [NombreContenedor]

Autor



Alexander Alzate Quintero

Desarrollador de Software

[Sitio web] [GitHub] [LinkedIn]