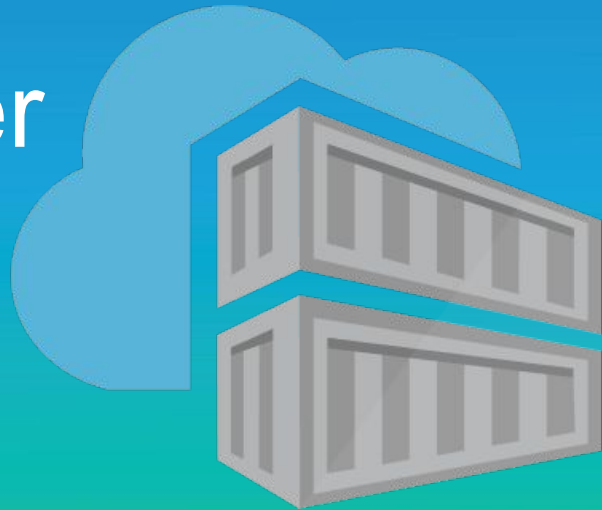
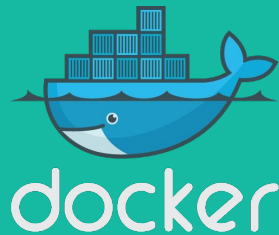


# Introducción a Docker



Pedromiguel Pinto { }

[pmpintogil@gmail.com](mailto:pmpintogil@gmail.com)



# Agenda

## Sección 1:

Introducción

¿Qué es Docker?

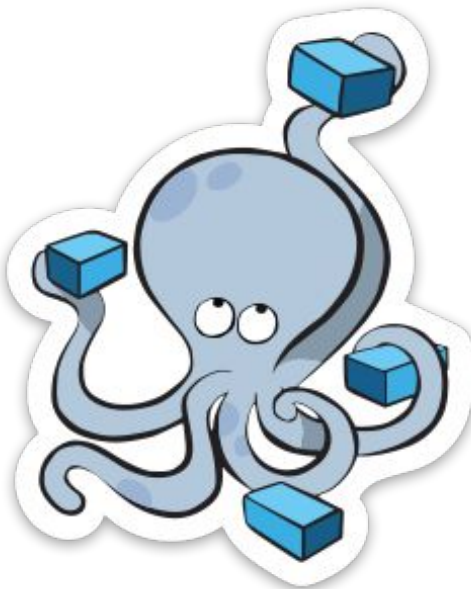
Instalando Docker (Demo)

## Sección 2:

Imágenes (Demo)

Contenedores (Demo)

Puertos (Demo)



## Sección 3:

Volúmenes  
(Demo)

## Sección 4:

Dockerfiles  
Builds

# Sección 1:

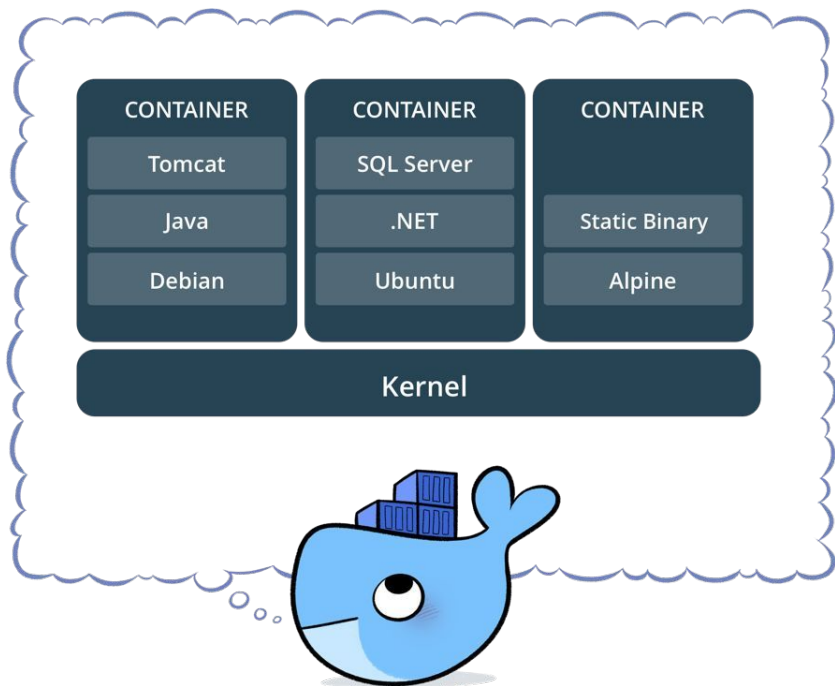
Introducción

¿Qué es Docker?

Instalando Docker (Demo)



# ¿Que es Docker?



**Docker** es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos. (Wikipedia)

- ❑ Empaquetado estandarizado para software y dependencias
- ❑ Aisla aplicaciones entre sí
- ❑ Comparte el mismo kernel del sistema operativo
- ❑ Funciona para todas las principales distribuciones de Linux
- ❑ Contenedores nativos de Windows Server 2016

# Instalando Docker (Demo)

## Actualizamos:

```
$ sudo apt update  
$ sudo apt upgrade
```

## Paquetes y requisitos previos:

```
$ sudo apt-get install curl apt-transport-https ca-certificates  
$ software-properties-common
```

## Agregar los repositorios de Docker:

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
$ sudo apt update  
$ apt-cache policy docker-ce
```

## Instalar Docker:

```
$ sudo apt install docker-ce  
$ sudo systemctl status docker  
$ sudo docker run hello-world
```

# Comandos Básicos de Docker

```
$ docker image pull node:latest
```

```
$ docker image ls
```

```
$ docker container run -d -p 5000:5000 --name node node:latest
```

```
$ docker container ps
```

```
$ docker container stop node(or <container id>)
```

```
$ docker container rm node (or <container id>)
```

```
$ docker image rmi (or <image id>)
```

```
$ docker build -t node:2.0 .
```

```
$ docker image push node:2.0
```

```
$ docker --help
```

# Sección 2:

Imágenes (Demo)

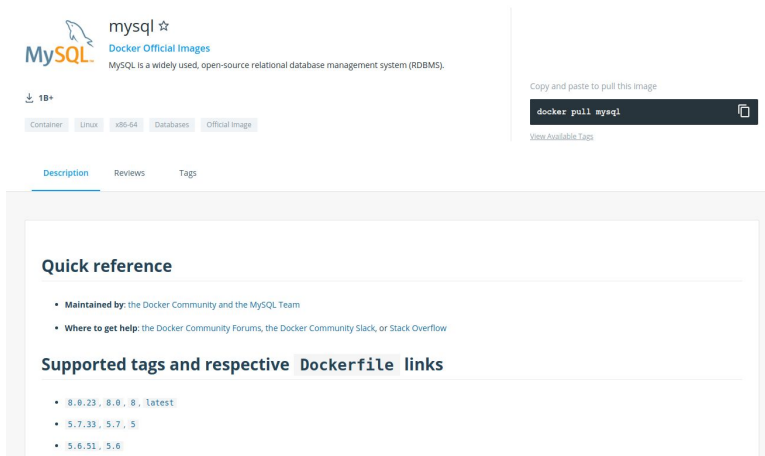
Contenedores (Demo)

Puertos (Demo)



# Imágenes:

Una imagen de Docker es una plantilla de solo lectura que contiene un conjunto de instrucciones para crear un contenedor que se puede ejecutar en la plataforma Docker. Proporciona una forma conveniente de empaquetar aplicaciones y entornos de servidores preconfigurados, que se puede usar para su propio uso privado o compartir públicamente con otros usuarios de Docker.



The screenshot shows the Docker Hub interface for the 'mysql' image. At the top, it displays the 'mysql' logo with a star, indicating it is an official image. Below the logo, it states 'MySQL is a widely used, open-source relational database management system (RDBMS)'. A download icon and '10+' are shown. A horizontal bar contains filters: 'Container', 'Linux', 'x86-64', 'Databases', and 'Official image'. On the right, a dark button says 'docker pull mysql' with a copy icon. Below this, a link says 'View Available Tags'. The main content area has tabs for 'Description', 'Reviews', and 'Tags'. Under 'Description', there is a 'Quick reference' section with two bullet points: 'Maintained by: the Docker Community and the MySQL Team' and 'Where to get help: the Docker Community Forums, the Docker Community Slack, or Stack Overflow'. Below that is a section titled 'Supported tags and respective Dockerfile links' with a list of tags: '8.0.23', '8.0.8', '8', 'latest', '5.7.33', '5.7', '5', and '5.6.51', '5.6'.

mysql ☆  
Docker Official Images  
MySQL is a widely used, open-source relational database management system (RDBMS).

10+

Container Linux x86-64 Databases Official image

Copy and paste to pull this image  
docker pull mysql

View Available Tags

Description Reviews Tags

### Quick reference

- **Maintained by:** the Docker Community and the MySQL Team
- **Where to get help:** the Docker Community Forums, the Docker Community Slack, or Stack Overflow

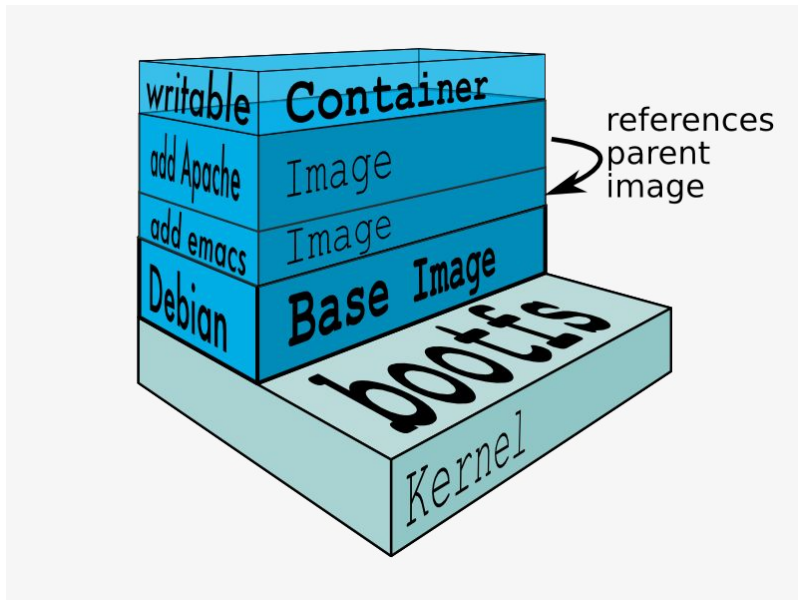
### Supported tags and respective Dockerfile links

- 8.0.23, 8.0.8, 8, latest
- 5.7.33, 5.7, 5
- 5.6.51, 5.6



# Contenedores:

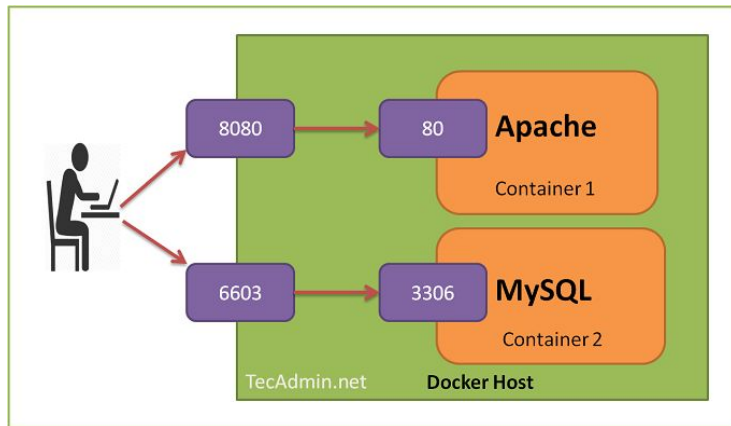
Una imagen de contenedor de Docker es un paquete de software ligero, independiente y ejecutable que incluye todo lo necesario para ejecutar una aplicación: código, tiempo de ejecución, herramientas del sistema, bibliotecas del sistema y configuración.



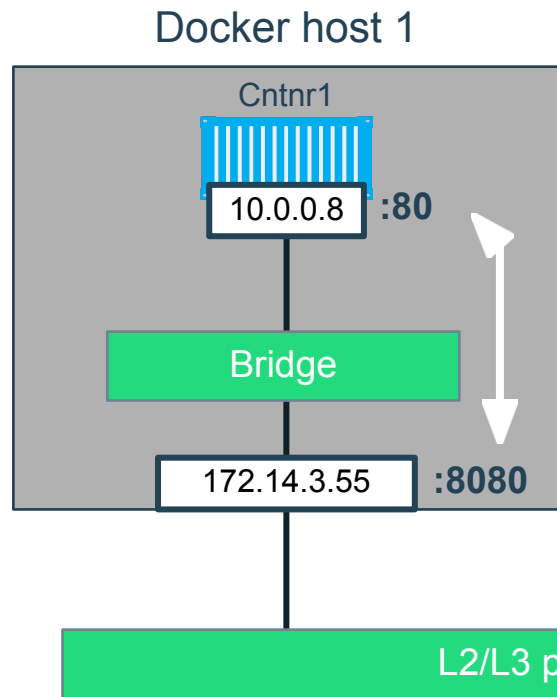
# Puertos:

Esto vincula el puerto 8080 del contenedor al puerto TCP 80 en 127.0.0.1 de la máquina host. También puede especificar puertos udp y sctp.

Tenga en cuenta que los puertos que no están vinculados al host (es decir, -p 80:80 en lugar de -p 127.0.0.1:80:80) serán accesibles desde el exterior. Esto también se aplica si configuró UFW para bloquear este puerto específico, ya que Docker administra sus propias reglas de iptables.



# Docker Bridge Networking and Port Mapping



Host port      Container port

```
$ docker container run -p 8080:80 ...
```

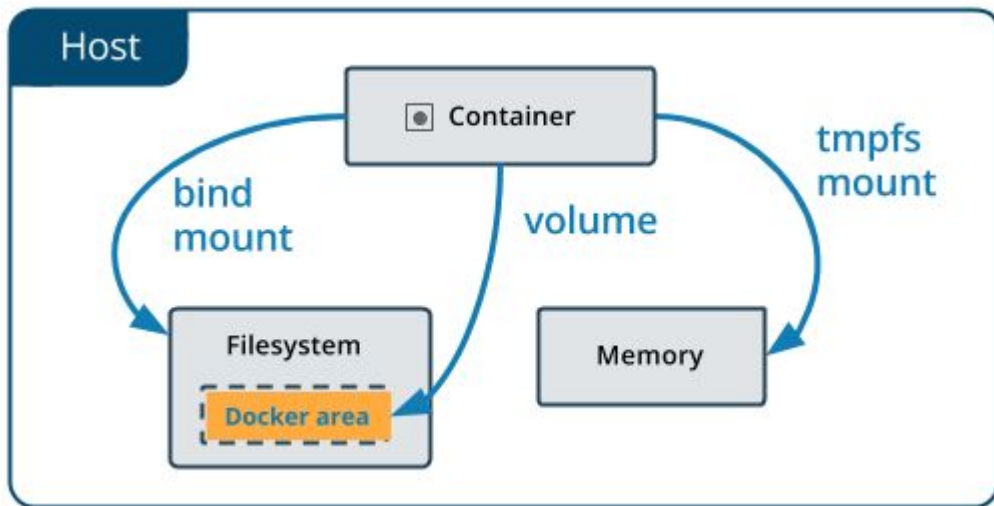
# Sección 3:

## Volúmenes (Demo)



# Volúmenes

Los volúmenes son el mecanismo preferido para conservar los datos generados y utilizados por los contenedores de Docker. Si bien los montajes de enlace dependen de la estructura del directorio y el sistema operativo de la máquina host, Docker administra completamente los volúmenes.



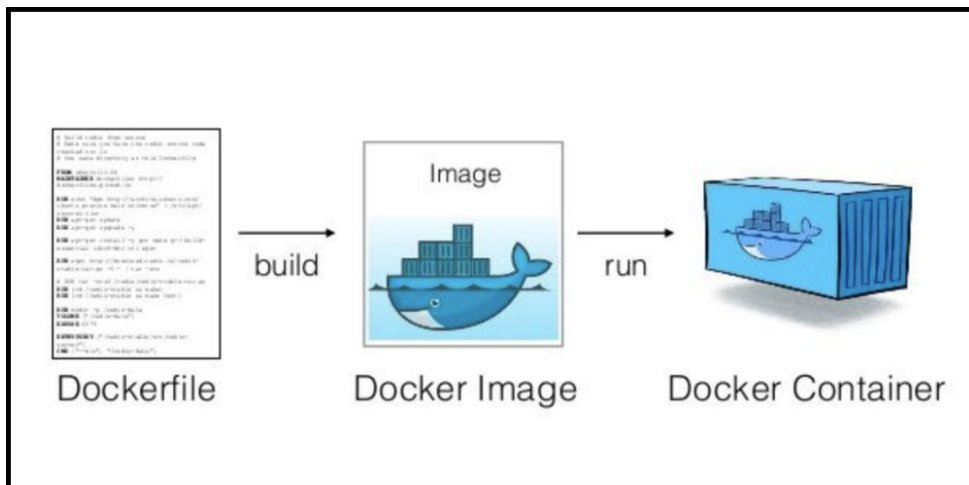
# Sección 4:

## Dockerfile Builds




# Dockerfile:

Un Dockerfile es un documento de texto que contiene todos los comandos que un usuario puede llamar en la línea de comandos para ensamblar una imagen. Al usar la compilación de Docker, los usuarios pueden crear una compilación automatizada que ejecute varias instrucciones de línea de comandos en sucesión. Esta página describe los comandos que puede usar en un Dockerfile.



# Formato de un Dockerfile

 Dockerfile x

```
1  # Create image based on the official Node 6 image from dockerhub
2  FROM node:latest
3
4  # Create a directory where our app will be placed
5  RUN mkdir -p /usr/src/app
6
7  # Change directory so that our commands run inside this new directory
8  WORKDIR /usr/src/app
9
10 # Copy dependency definitions
11 COPY package.json /usr/src/app
12
13 # Install dependencies
14 RUN npm install
15
16 # Get all the code needed to run the app
17 COPY . /usr/src/app
18
19 # Expose the port the app runs in
20 EXPOSE 4200
21
22 # Serve the app
23 CMD ["npm", "start"]
```

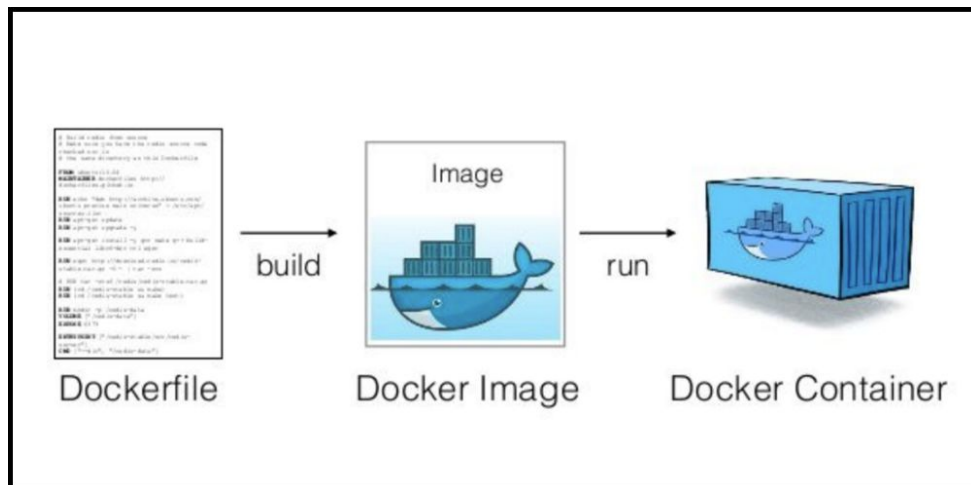


# Cada línea crea una capa



# Docker build:

El comando `docker build` crea imágenes de Docker a partir de un `Dockerfile` y un "contexto". El contexto de una compilación es el conjunto de archivos ubicados en la RUTA o URL especificada. El proceso de construcción puede hacer referencia a cualquiera de los archivos del contexto. Por ejemplo, su compilación puede usar una instrucción `COPY` para hacer referencia a un archivo en el contexto.





docker