

Docker & Cloud

Una breve introducción a los contenedores enfocado especialmente en [Docker](#).

[Principio de los contenedores](#)

[Evolución](#)

[Colocation](#)

[Virtualization](#)

[Serverless](#)

[Aplicaciones monolíticas vs microservicios](#)

[Comparativa](#)

[Datos Relevantes](#)

[¿Por qué Docker?](#)

[Problemas comunes](#)

[Ventajas](#)

[Componentes](#)

[Hoja de trucos para Docker CLI](#)

[ECS, Fargate, Compute Engine](#)

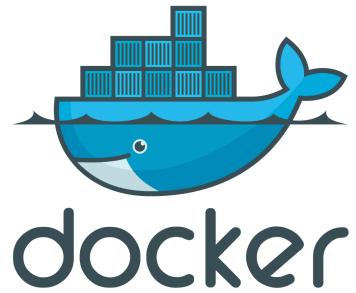
[Referencias y links útiles](#)

Principio de los contenedores

Un contenedor es una caja metálica sellada,
rígida y reutilizable que se utiliza para contener

mercancías que requieren transporte por barco, camión o ferrocarril.

El contenedor debe estar construido para un uso repetido, fácil de llenar o vaciar y especialmente diseñado para facilitar el transporte y separación de mercancías.



Evolución

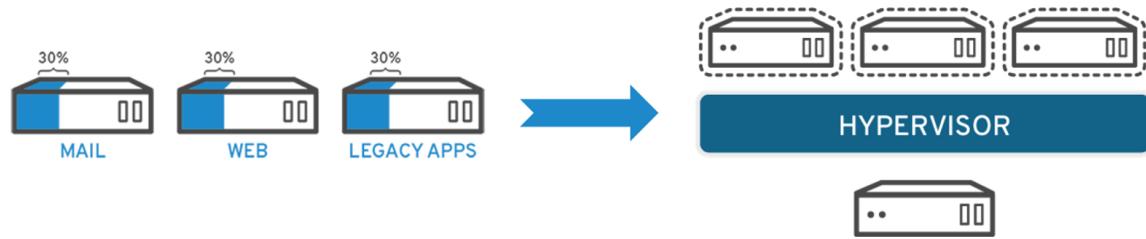
How did we get here? Where are we going?



<https://techprodezza.code.blog/2020/06/14/google-cloud-platform-gcp/>

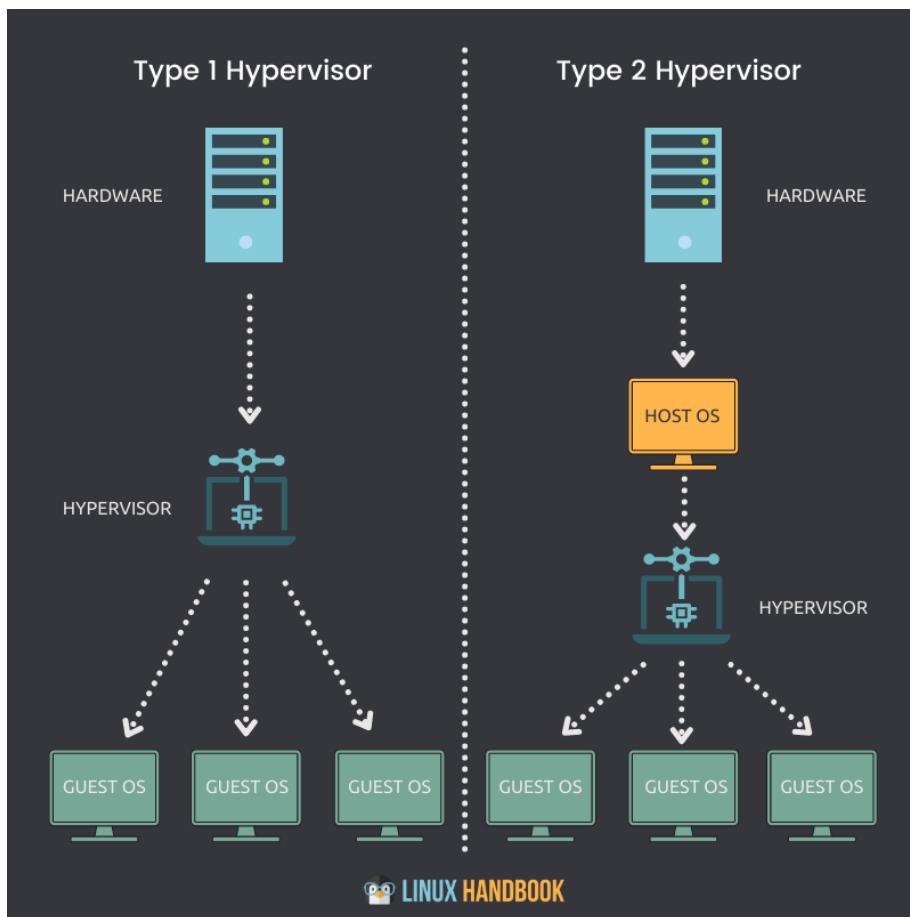
Colocation

Dedicated Servers vs Virtual Machines

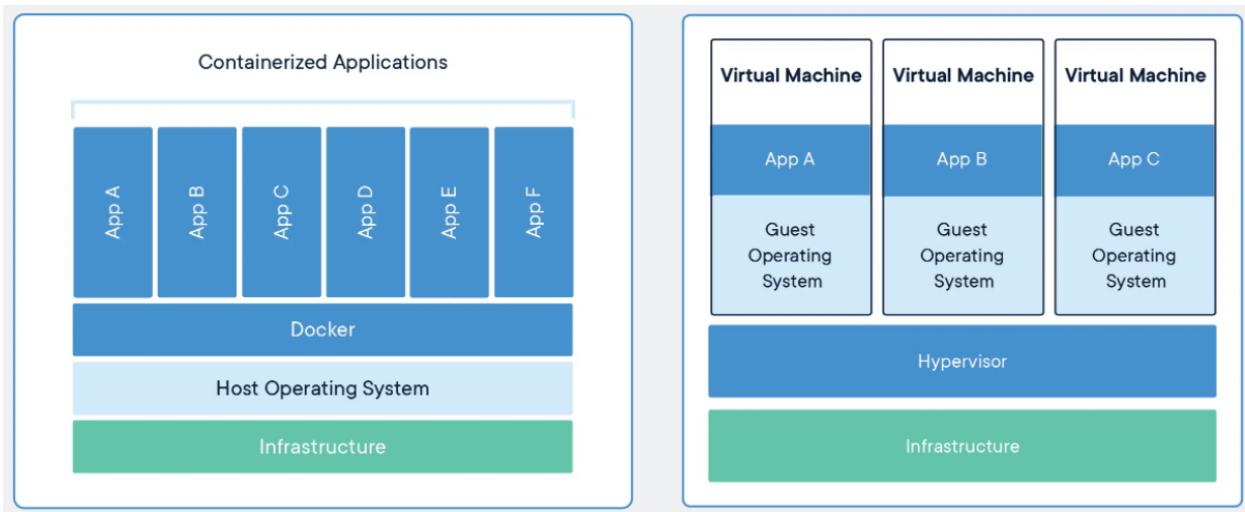


<https://blog.equinix.com/blog/2020/02/25/how-to-speak-like-a-data-center-geek-bare-metal-vs-virtualization-vs-serverless/>

Virtualization



<https://linuxhandbook.com/what-is-hypervisor/>



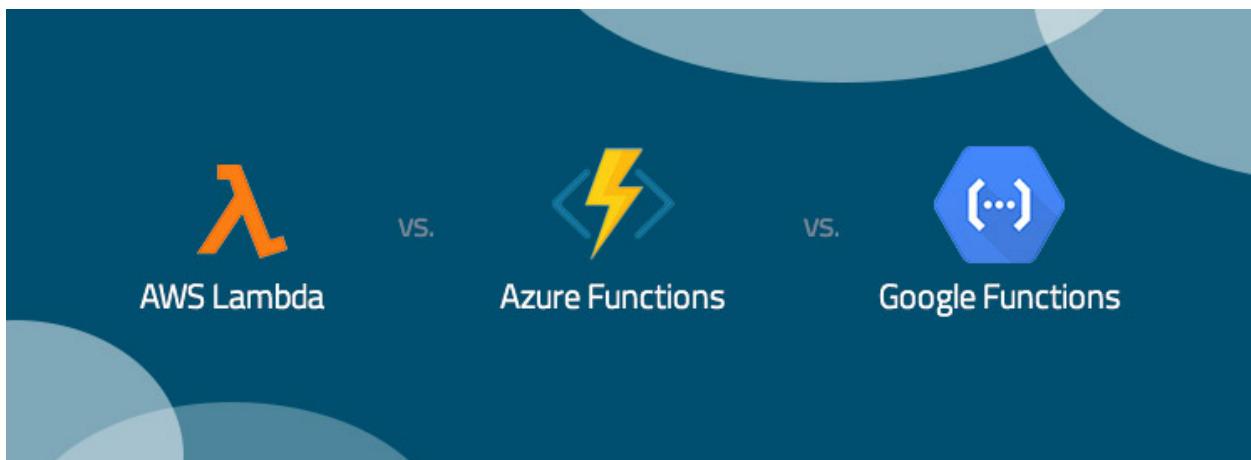
Comparación y explicación de los contenedores vs VMs

Serverless

Serverless and container-based applications deploy the fastest

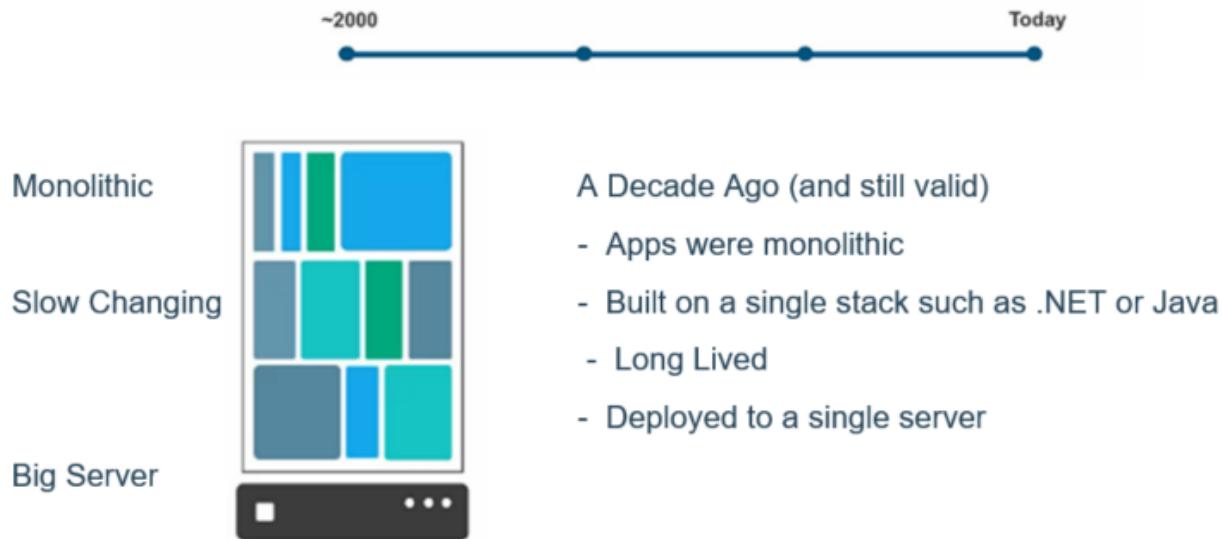


<https://www.cloudflare.com/es-es/learning/serverless/serverless-vs-containers/>



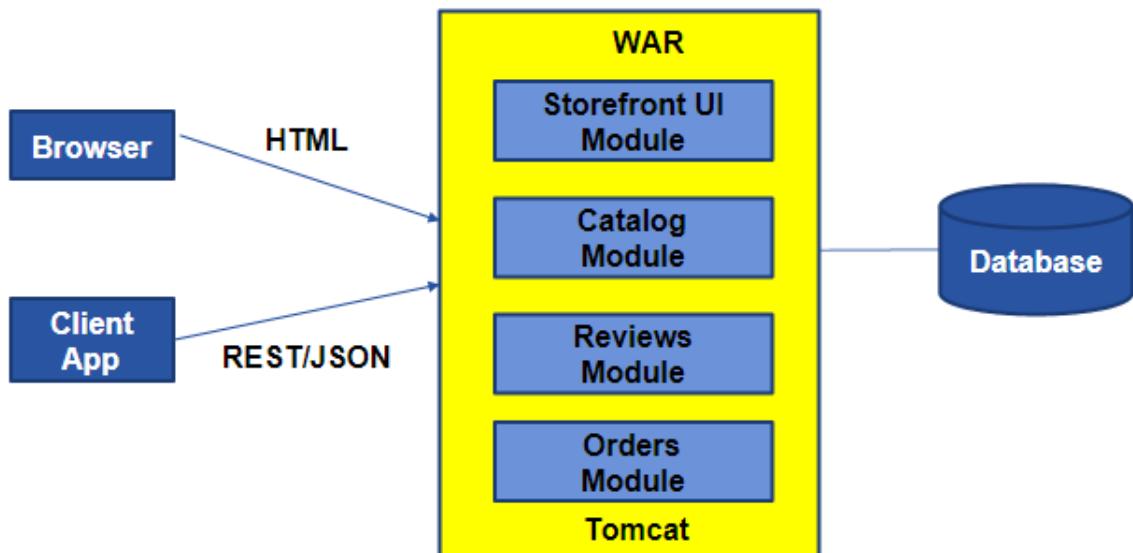
Aplicaciones monolíticas vs microservicios

Applications have changed dramatically



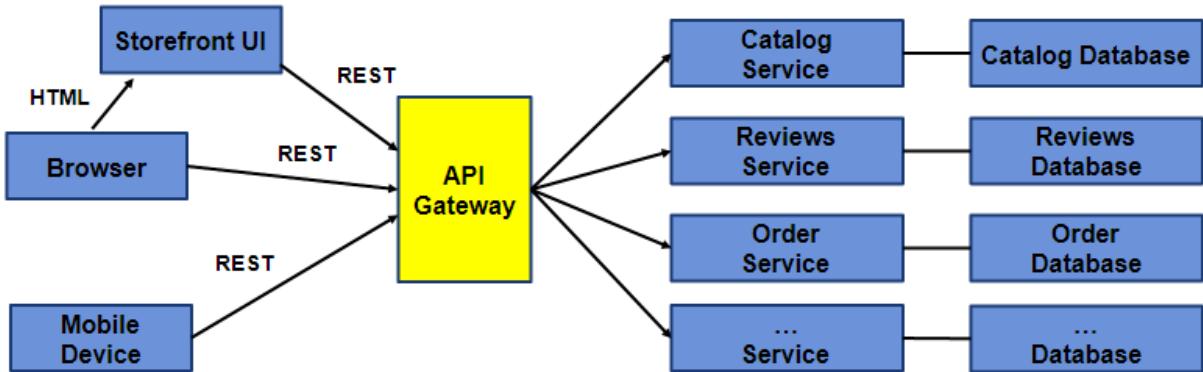
https://dockerlabs.collabnix.com/docker/Docker_VIT_Intro/Docker_VIT_Intro.html

A Close Look at Monolithic



https://dockerlabs.collabnix.com/docker/Docker_VIT_Intro/Docker_VIT_Intro.html

Microservice Architecture



https://dockerlabs.collabnix.com/docker/Docker_VIT_Intro/Docker_VIT_Intro.html

Comparativa

	BARE METAL	VIRTUAL MACHINE	CONTAINER	SERVERLESS
Boot Time	~5-10 min	~2 min	~2 secs	~0.0003 secs
App Deployment Lifecycle	Deploy in weeks Live for years	Deploy in minutes Live for weeks	Deploy in seconds Live for minutes/hours	Deploy in milliseconds Live for seconds
Deployment Complexity	Need to know: 1. Hardware 2. OS 3. Runtime environment 4. Application code	Need to know: 1. OS 2. Runtime environment 3. Application code	Need to know: 1. Runtime environment 2. Application code	Need to know: 1. Application code
Investment	Buy/rent dedicated server	Rent a dedicated VM on a shared server	Rent containers and pay for actual runtime	Pay for compute resources used during runtime
Scaling	Can take months* Should be approved by a panel of experts	Takes hours Should be approved by administrators	Takes seconds Policy driven scaling	Takes milliseconds Event driven scaling

*For on-premises bare metal. Bare metal cloud services can be dynamically provisioned in minutes

<https://blog.equinix.com/blog/2020/02/25/how-to-speak-like-a-data-center-geek-bare-metal-vs-virtualization-vs-serverless/>

Datos Relevantes

Fundado en el año 2010, lanzado en 2011 y liberado como proyecto open-source en Marzo de 2013. El lanzamiento de Docker inició una revolución en el desarrollo de aplicaciones, al democratizar los contenedores de software. Docker desarrolló una tecnología de contenedor de Linux, una que es portátil, flexible y fácil de implementar, además permite construir, distribuir y ejecutar cualquier aplicación en cualquier lado.

Accelerate how you build, share, and run modern applications.

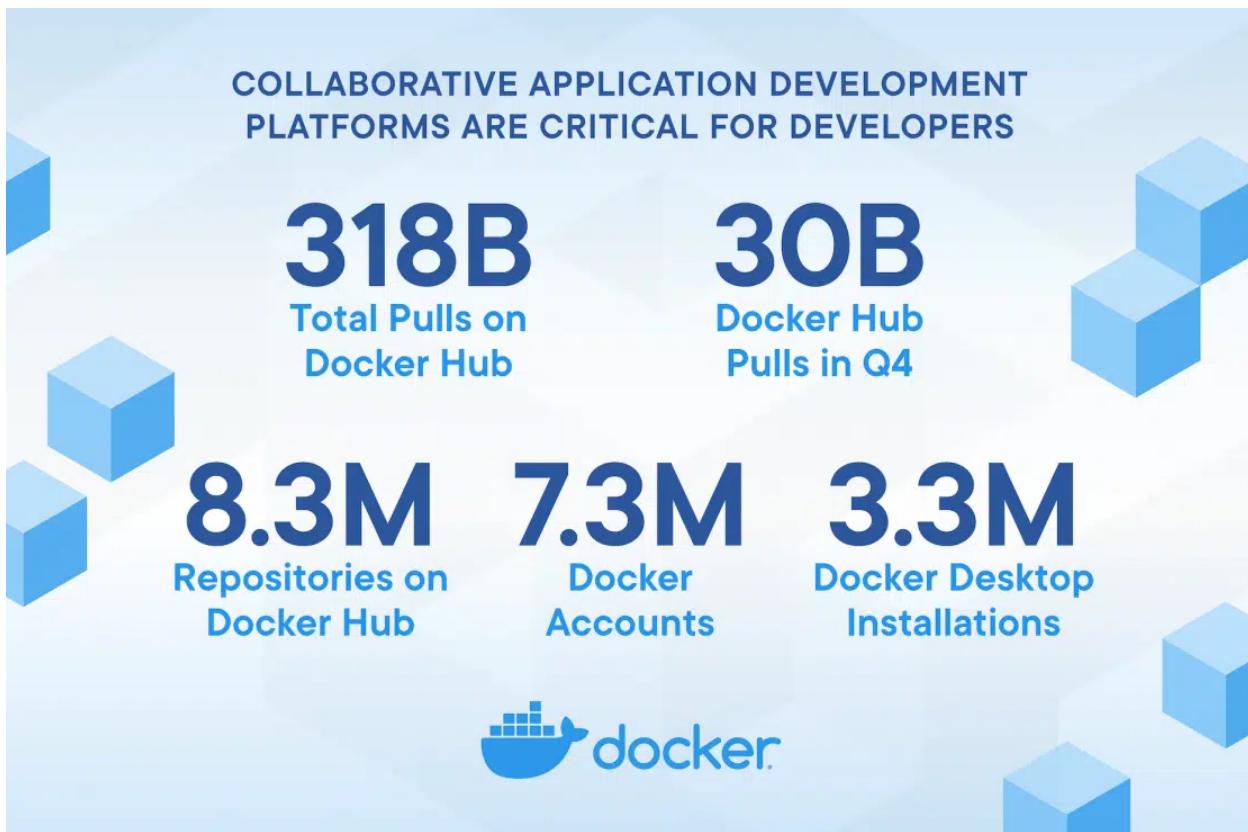
13 million +
developers

7 million +
applications

13 billion +
monthly image downloads

See who uses Docker





<https://www.docker.com/blog/docker-index-shows-continued-massive-developer-adoption-and-activity-to-build-and-share-apps-with-docker/>

¿Por qué Docker?

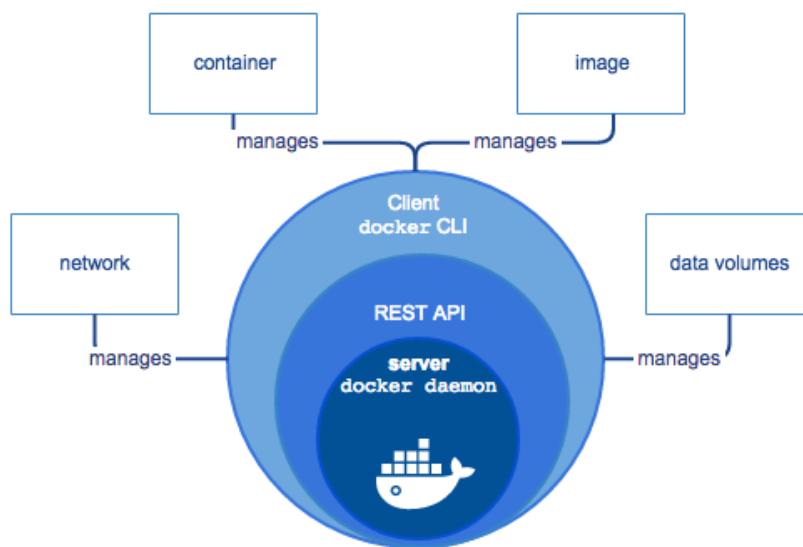
Problemas comunes

- Construir
- Distribuir
- Ejecutar
- Dependencias
- Entornos de ejecución (Hardware)
- Versionamiento
- Costos

Ventajas

- Flexible
- Distribuir
- Liviano
- Portable
- Escalable
- Efímero
- Seguros

Componentes



<https://www.recursion.mx/docker/501/>

Docker daemon: la que permite crear, destruir, escalar, es el que va a interactuar con el OS.

REST API: para comunicarse consigo mismo, remotamente, por medio de HTTP, para darle instrucciones al Docker daemon.

Docker CLI: para interactuar directamente con Docker por medio del cliente de Docker.

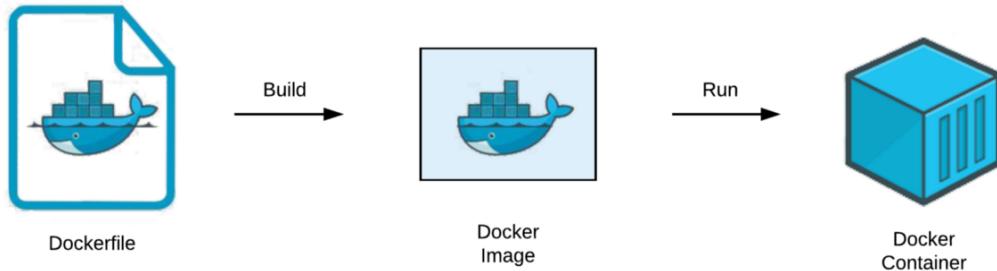
Container: donde se correrán los proyectos.

Images: los artefactos para empaquetar los contenedores y poder compartirlos.

Volumes: permite acceder al sistema de archivos de los contenedores.

Network: permite a los distintos contenedores comunicarse entre sí y con el mundo exterior.

Dockerfile: para construir imágenes personalizadas a partir de plantillas



```
# syntax=docker/dockerfile:1
FROM ubuntu:18.04
COPY . /app
RUN make /app
CMD python /app/app.py
```

Docker Compose: compose es una herramienta para definir y ejecutar aplicaciones de varios contenedores, utiliza YAML para configurar los servicios y luego con un solo comando crea e inicia todos los servicios para la aplicación, ejm, Apache + PHP + MySQL

```
version: "3"

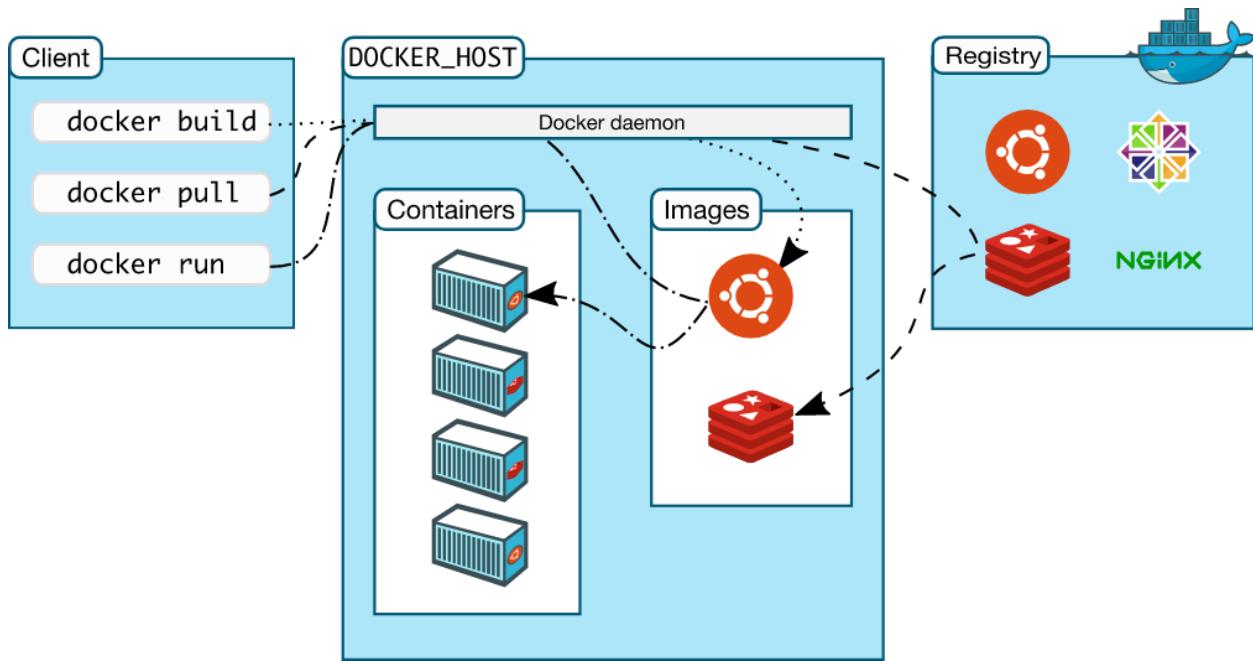
services:
  mariadb:
    image: mariadb:10.7
    container_name: mariadb
    expose:
      - 3306
    restart: always
    volumes_from:
      - mariadb-data
    environment:
      MYSQL_ROOT_PASSWORD: secret
      MYSQL_DATABASE: mydb
      MYSQL_USER: myuser
      MYSQL_PASSWORD: password
  mariadb-data:
    image: mariadb:10.7
    container_name: mariadb-data
```

```

volumes:
  - /var/lib/mysql
  command: "true"
nginx:
  image: nginx:latest
  container_name: "nginx"
  build: ./nginx
  restart: always
  depends_on:
    - php
    - app-data
  ports:
    - "80:80"
    - "443:443"
  links:
    - php
  volumes_from:
    - app-data
setup:
  build: ./php/
  volumes_from:
    - app-data
  command: "composer install -d /var/www/ --prefer-dist --no-progress"
php:
  build: ./php/
  container_name: "php"
  restart: always
  depends_on:
    - setup
  expose:
    - 9000
  links:
    - mariadb
  volumes_from:
    - app-data
app-data:
  image: php:7.4-fpm-alpine
  container_name: app-data
  volumes:
    - ./app:/var/www/

```

Docker Hub: es el repositorio proporcionado por Docker para buscar y compartir imágenes de contenedores con el equipo.



Docker Swarm: es una herramienta de orquestación de contenedores, lo que significa que permite al usuario administrar múltiples contenedores implementados en múltiples máquinas host.

Hoja de trucos para Docker CLI



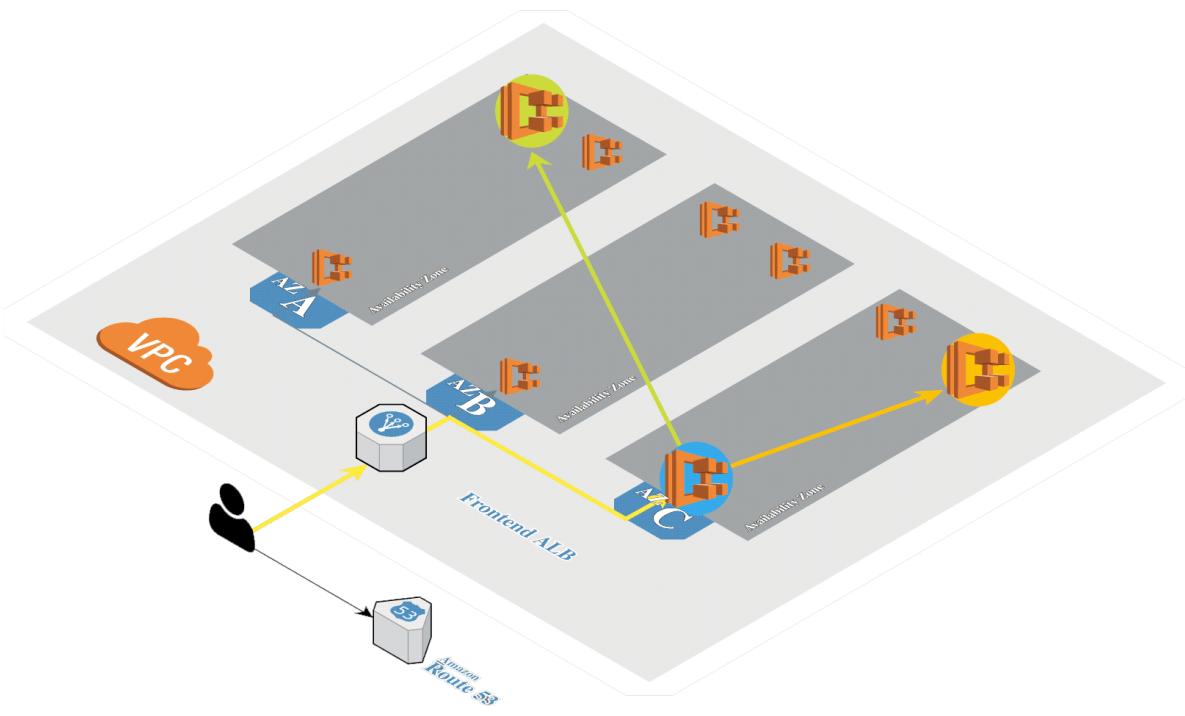
Cheatsheet for Docker CLI

Run a new Container	Manage Containers	Manage Images	Info & Stats
<p>Start a new Container from an Image <code>docker run IMAGE</code> <code>docker run nginx</code></p> <p>...and assign it a name <code>docker run --name CONTAINER IMAGE</code> <code>docker run --name web nginx</code></p> <p>...and map a port <code>docker run -p HOSTPORT:CONTAINERPORT IMAGE</code> <code>docker run -p 8080:80 nginx</code></p> <p>...and map all ports <code>docker run -P IMAGE</code> <code>docker run -P nginx</code></p> <p>...and start container in background <code>docker run -d IMAGE</code> <code>docker run -d nginx</code></p> <p>...and assign it a hostname <code>docker run --hostname HOSTNAME IMAGE</code> <code>docker run --hostname srv nginx</code></p> <p>...and add a dns entry <code>docker run --add-host HOSTNAME:IP IMAGE</code></p> <p>...and map a local directory into the container <code>docker run -v HOSTDIR:TARGETDIR IMAGE</code> <code>docker run -v ./:/usr/share/nginx/html nginx</code></p> <p>...but change the entrypoint <code>docker run -it --entrypoint EXECUTABLE IMAGE</code> <code>docker run -it --entrypoint bash nginx</code></p>	<p>Show a list of running containers <code>docker ps</code></p> <p>Show a list of all containers <code>docker ps -a</code></p> <p>Delete a container <code>docker rm CONTAINER</code> <code>docker rm web</code></p> <p>Delete a running container <code>docker rm -f CONTAINER</code> <code>docker rm -f web</code></p> <p>Delete stopped containers <code>docker container prune</code></p> <p>Stop a running container <code>docker stop CONTAINER</code> <code>docker stop web</code></p> <p>Start a stopped container <code>docker start CONTAINER</code> <code>docker start web</code></p> <p>Copy a file from a container to the host <code>docker cp CONTAINER:SOURCE TARGET</code> <code>docker cp web:/index.html index.html</code></p> <p>Copy a file from the host to a container <code>docker cp TARGET CONTAINER:SOURCE</code> <code>docker cp index.html web:/index.html</code></p> <p>Start a shell inside a running container <code>docker exec -it CONTAINER EXECUTABLE</code> <code>docker exec -it web bash</code></p> <p>Rename a container <code>docker rename OLD_NAME NEW_NAME</code> <code>docker rename 096 web</code></p> <p>Create an image out of container <code>docker commit CONTAINER</code> <code>docker commit web</code></p>	<p>Download an image <code>docker pull IMAGE[:TAG]</code> <code>docker pull nginx</code></p> <p>Upload an image to a repository <code>docker push IMAGE</code> <code>docker push myimage:1.0</code></p> <p>Delete an image <code>docker rmi IMAGE</code></p> <p>Show a list of all Images <code>docker images</code></p> <p>Delete dangling images <code>docker image prune</code></p> <p>Delete all unused images <code>docker image prune -a</code></p> <p>Build an image from a Dockerfile <code>docker build DIRECTORY</code> <code>docker build .</code></p> <p>Tag an image <code>docker tag IMAGE NEWIMAGE</code> <code>docker tag ubuntu ubuntu:18.04</code></p> <p>Build and tag an image from a Dockerfile <code>docker build -t IMAGE DIRECTORY</code> <code>docker build -t myimage .</code></p> <p>Save an image to .tar file <code>docker save IMAGE > FILE</code> <code>docker save nginx > nginx.tar</code></p> <p>Load an image from a .tar file <code>docker load -i TARFILE</code> <code>docker load -i nginx.tar</code></p>	<p>Show the logs of a container <code>docker logs CONTAINER</code> <code>docker logs web</code></p> <p>Show stats of running containers <code>docker stats</code></p> <p>Show processes of container <code>docker top CONTAINER</code> <code>docker top web</code></p> <p>Show installed docker version <code>docker version</code></p> <p>Get detailed info about an object <code>docker inspect NAME</code> <code>docker inspect nginx</code></p> <p>Show all modified files in container <code>docker diff CONTAINER</code> <code>docker diff web</code></p> <p>Show mapped ports of a container <code>docker port CONTAINER</code> <code>docker port web</code></p>

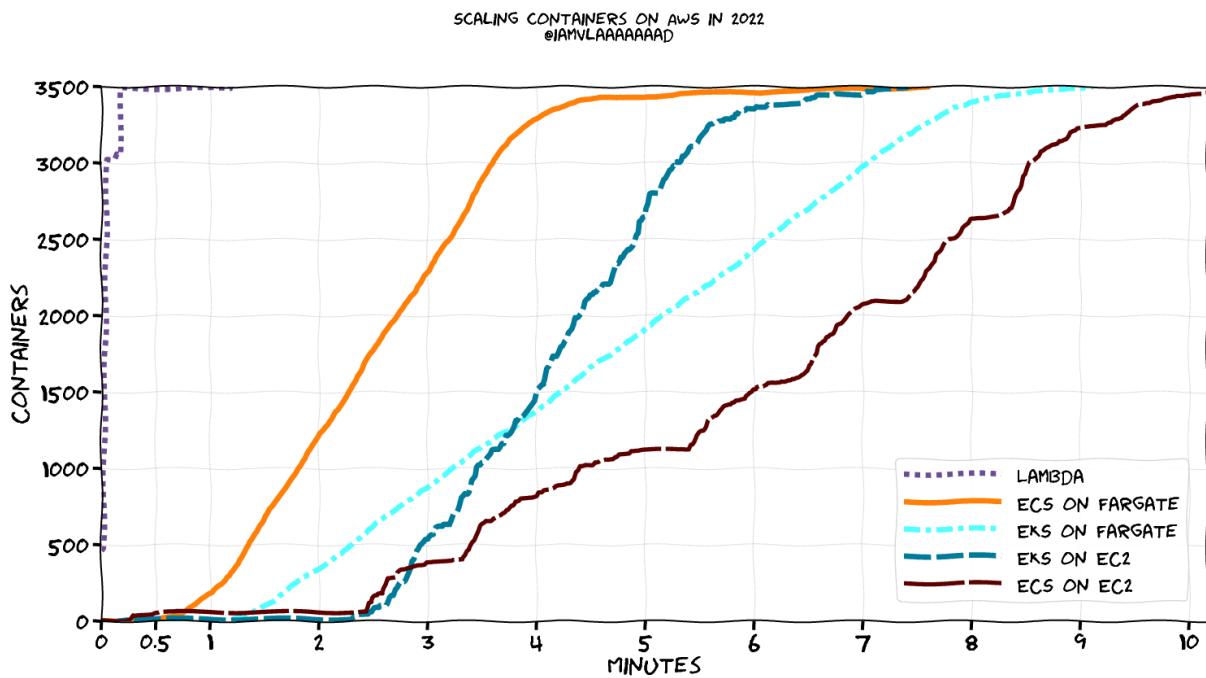
<https://dockermlops.com/docker/cheatsheet/>

ECS, Fargate, Compute Engine

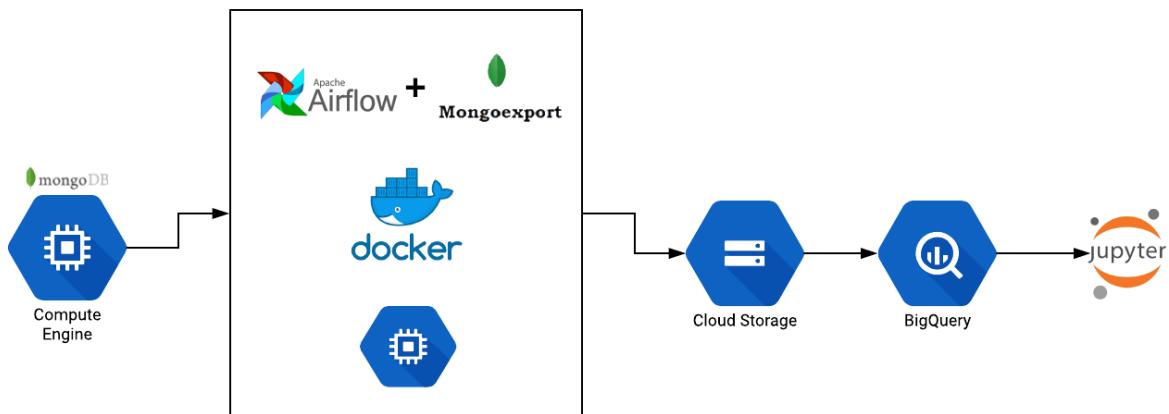
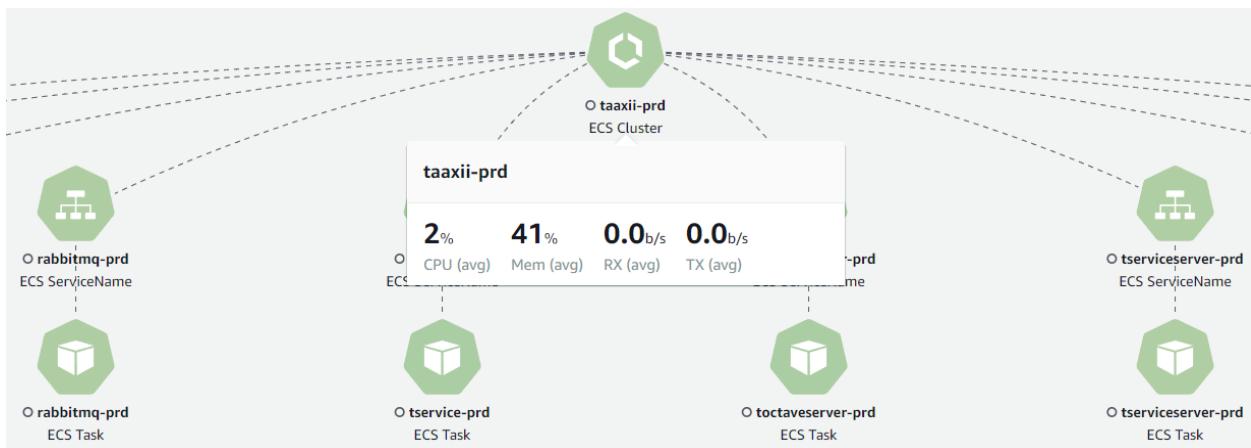
- el 80% de todas las aplicaciones en contenedores que se ejecutan en la nube se ejecutan en AWS
- 150% de crecimiento interanual de los servicios de contenedores de AWS
- más de 8 billones de imágenes descargadas semanalmente con ECR



<https://ecsworkshop.com/>



<https://www.vladionescu.me/posts/scaling-containers-on-aws-in-2022/>



<https://junjiejiang94.medium.com/get-started-with-airflow-google-cloud-platform-docker-a21c46e0f797>

Referencias y links útiles

Docker Documentation

Home page for Docker's documentation

<https://docs.docker.com/>

Docker Labs - The #1 Docker Tutorials and Free Resources for all Levels

A \$0 Learning Platform for all Levels - from the ground up Over 500+ Highly Interactive Docker Tutorials and Guides Well tested on Docker Desktop and can be run on Browser (no Infrastructure required) DevOps Application Developers Solution

<https://dockermlops.collabnix.com/>

Creating a container image for use on Amazon ECS

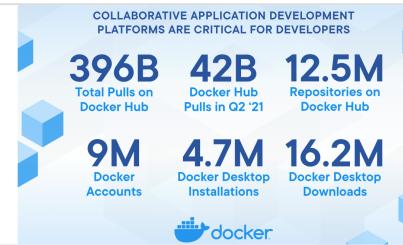
Amazon ECS uses Docker images in task definitions to launch containers. Docker is a technology that provides the tools for you to build, run, test, and deploy distributed applications in containers. Docker provides a walkthrough on deploying containers on Amazon ECS. For more information, see

 <https://docs.aws.amazon.com/AmazonECS/latest/developerguide/create-container-image.html>

Docker Index Shows Momentum in Developer Community Activity - Docker

The latest edition of the Docker Index is in, and it shows a continued growth in activity across the Docker community. The momentum we are seeing since the last Docker Index in February 2021 edition continues to grow. You'll recall that we started

 <https://www.docker.com/blog/docker-index-shows-surging-momentum-in-developer-community-activity-again/>



Curso de Docker

Éste es un curso sobre los fundamentos de Docker, tanto para quienes no conocen esta tecnología y quieren aprender a usarla como para quienes la usan frecuentemente pero quieren profundizar en los detalles de cómo funciona en

 <https://platzi.com/cursos/docker/>

