Docker Compose

Explicación de práctica 4

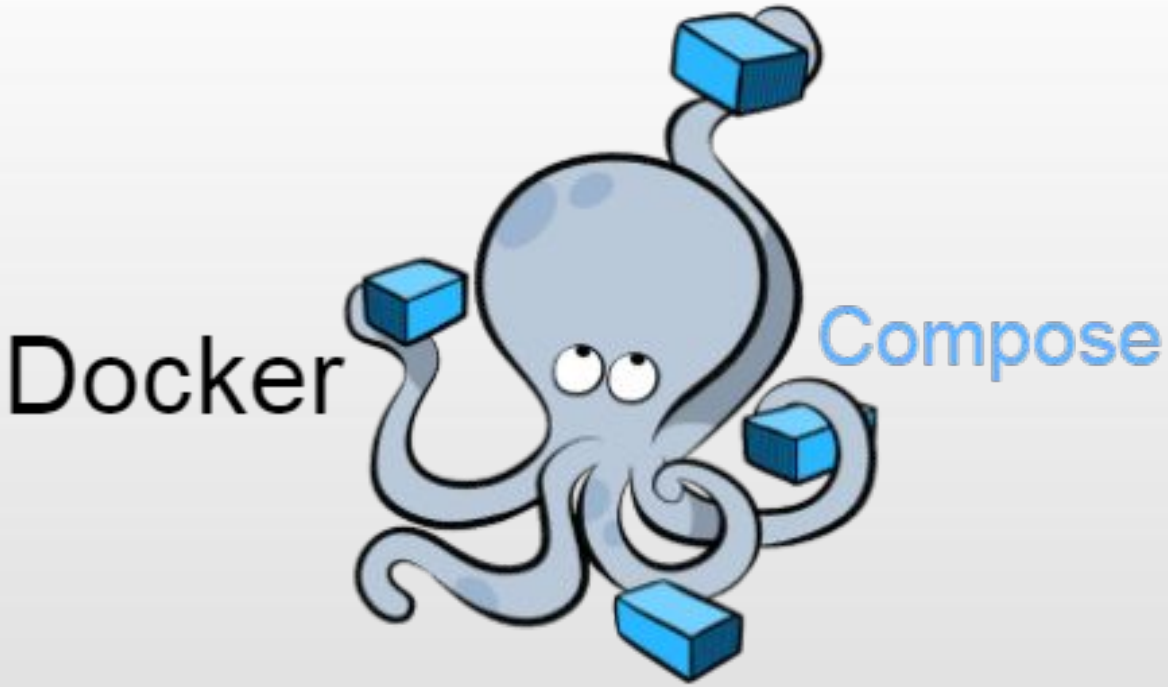
Sistemas Operativos

Facultad de Informática
Universidad Nacional de La Plata

2025

Agenda

1. Docker Compose



¿Qué es docker compose?

Como vimos, docker permite empaquetar y ejecutar una aplicación/proceso en contenedores.

Por ahora vimos cómo desplegar UN contenedor que puede correr UN proceso determinado, o UN servicio determinado.

¿Qué es docker compose?

Los sistemas suelen estar formados por varias partes que se intercomunican entre sí. Esto es así especialmente en sistemas diseñados como un conjunto de microservicios, lugar donde el uso de docker se da mayoritariamente.

Ejemplo básico:

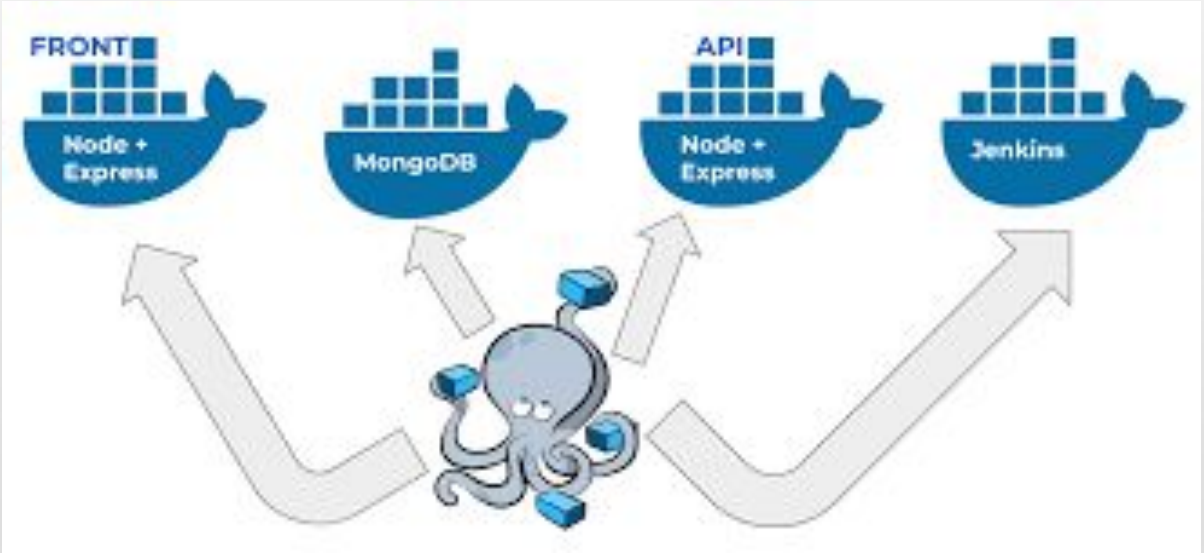
App + Base de datos.

Por lo tanto, para desplegar una app "dockerizada" generalmente voy a necesitar desplegar un conjunto de contenedores.

¿Qué es docker compose?

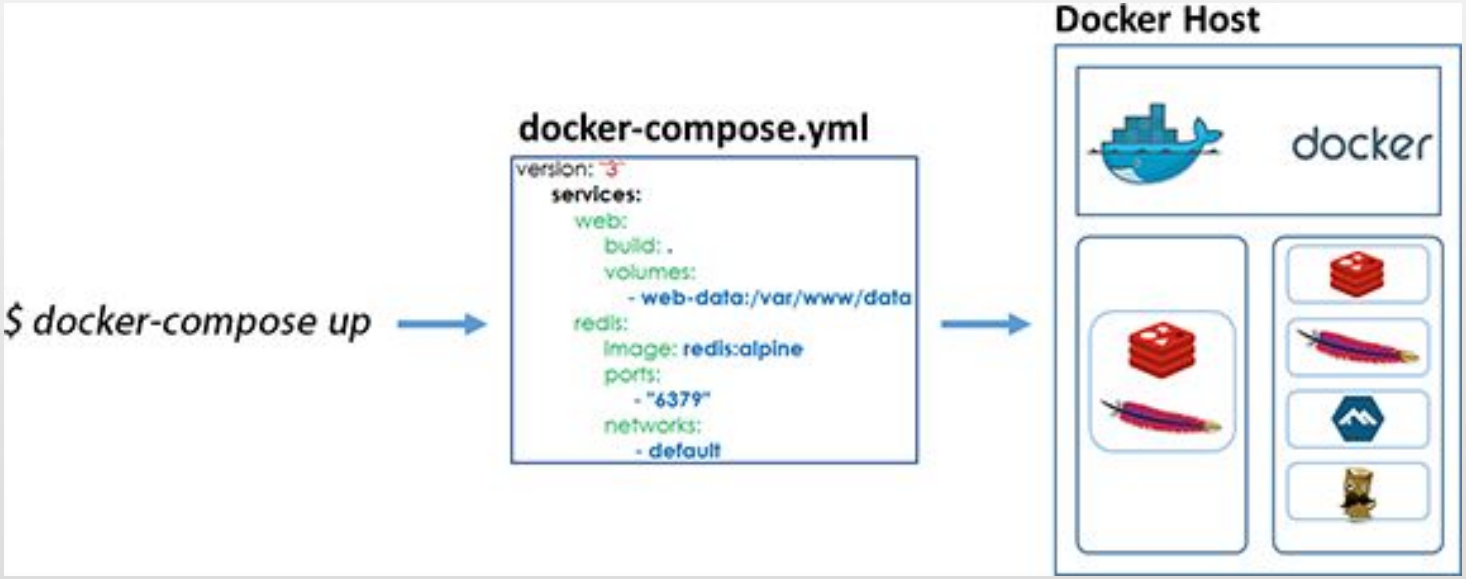
Docker Compose es una herramienta que facilita el despliegue de aplicaciones compuestas por múltiples contenedores.

Este conjunto de contenedores se comunican entre sí para brindar la funcionalidad objetivo.



¿Qué es docker compose?

Consideramos entonces a docker compose como el conjunto de la herramienta (el binario) y los archivos de configuración llamados "archivo compose" o "compose file" que definen los recursos deseados.



En la actualidad existen 2 versiones del binario docker compose.

Important

Compose V1 no longer receives updates and will not be available in new releases of Docker Desktop after June 2023.

Compose V2 is included with all currently supported versions of Docker Desktop. For more information, see Migrate to Compose V2.

Más información en
https://docs.docker.com/compose/install/

- Docker Compose es un binario. Es la herramienta que se invoca para interpretar los “compose file” y desplegar los contenedores allí definidos.

- El “compose file” es un archivo escrito en el lenguaje de marcado YAML que describe las características de los contenedores a desplegar. Usualmente llamado docker compose.yml.

Toda la configuración se define en el archivo compose!

A cada contenedor definido en el “compose file” se lo denomina “service”.

Estructura básica de un archivo docker compose.yml:

```yaml
version: "3.9"
services:
  web:
    build: .
    ports:
      - "8000:5000"
    volumes:
      - .:/code
    environment:
      FLASK_ENV: development
  redis:
    image: "redis:alpine"
```

Actualmente existen tres versiones para el formato de archivo de docker compose.yml:

- Versión 1: Está obsoleta.
- Versión 2.x
- Versión 3.x: Es la última versión y la recomendada por Docker.
    - Desde Junio del 2020 se utiliza la versión 3.9.

Las versiones 2.x y 3.x comparten estructura pero no algunas opciones. Veremos la sintaxis de la V3.

Cuando se invoca el binario docker compose, este busca en la carpeta desde donde es invocado algún archivo con nombre docker compose.yml.

También se le puede pasar como parámetro cuál es el "compose file" a utilizar.

En base al contenido del "compose file", se iniciarán los distintos contenedores con las características definidas para cada uno.

Comandos básicos

- Verificar la versión instalada: **docker compose -v**
- Construir imágenes de servicios (desde Dockerfile): **docker compose build**
- Crear todos los contenedores: **docker compose create**
- Iniciar todos los contenedores: **docker compose up**
- Frenar todos los contenedores: **docker compose down**
- Iniciar contenedores en background: **docker compose up -d**
- Listar los contenedores iniciados: **docker compose ps**
- Eliminar todos los contenedores: **docker compose rm**
- Ver los logs de los contenedores: **docker compose logs**

Simplicidad: Todo definido en un archivo con una estructura determinada y documentada.

Replicabilidad: Si necesito volver a desplegar los servicios, simplemente corro el mismo archivo.

"Shareability": Facilidad de compartir, solo necesito proveer el "compose file" (NOTA: las imágenes docker deben estar accesibles)

Versionable: Facilidad de usar control de versiones (por ej git).

Vamos a comparar cómo iniciar un contenedor `mysql` en la versión 5.7, utilizando variables de entorno para los nombres de usuario, nombre de base de datos y contraseñas, publicando el puerto "`3306`" en el host, que use un volumen nombrado "`data`" para los datos de la bbdd ("`/var/lib/mysql`") y que tenga como política de reinicio "`always`" y nombre "`database`".

Comparemos cómo hacerlo usando directamente docker y docker compose.

Con docker:

Primero tengo que crear el volumen ”data”:

```
~$ docker volume create data
```

Luego puedo iniciar el contenedor:

```
docker run -d \
  --name database \
  -e MYSQL_ROOT_PASSWORD=myrootpass \
  -e MYSQL_DATABASE=mydb \
  -e MYSQL_USER=myuser \
  -e MYSQL_PASSWORD=mypass \
  -p 3306:3306 \
  -v data:/var/lib/mysql \
  --restart always \
   mysql:5.7
```

Para “entrar” al contenedor:

```
docker exec -it database /bin/bash
```

En cambio, con docker compose:

docker compose.yml:

```yaml
version: "3.9"
services:
  database:
    image: mysql:5.7
    container_name: database
    volumes:
      - data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: myrootpass
      MYSQL_DATABASE: mydb
      MYSQL_USER: myuser
      MYSQL_PASSWORD: mypass
    ports:
      - 3306:3306
volumes:
  data:
```

y en el directorio ejecutar:

docker compose up.

Pensemos,

- ¿Que es más claro?
- ¿Que es más replicable?
- ¿Que es más fácil de compartir?
- ¿Que es "más" versionable?
- ¿Y si en lugar de solo 1 contenedor, tuviera que iniciar 5 definiendo todo para cada uno?

Ventajas docker compose

Característica	Docker CLI	Docker Compose
Sintaxis	Comando único con múltiples flags	Archivo YAML estructurado
Variables de entorno	`-e` por cada variable	Bloque `environment:`
Puertos	`-p 3306:3306`	Bloque `ports:`
Volúmenes	`-v data:/var/lib/mysql`	Bloque `volumes:` + declaración
Política de reinicio	`--restart always`	`restart: always`
Gestión de configuración	Parámetros en línea	Configuración centralizada
Reusabilidad	Dificil de versionar/replicar	Fácil de compartir y versionar
Escalabilidad	Manual para múltiples servicios	Gestión integrada de multi-servicios

Referencias

- https://docs.docker.com/compose/install/
- https://github.com/docker/compose
- https://docs.docker.com/compose/compose-file/
- https://docs.docker.com/compose/reference/
- https://docs.docker.com/compose/gettingstarted/

¿Preguntas?


200
DOSCIENTOS
PESOS
BANCO CENTRAL DE LA
REPÚBLICA ARGENTINA
200
200
200
CC
200