¿Cómo funcionan Docker y Docker Compose?

Diferencia entre una imagen de Docker usada con Docker Compose y sin Docker Compose

Beneficio de Docker comparado con máquinas virtuales (VMs)

Pertinencia de la estructura de directorios requerida para este proyecto

Relevancia de las versiones del sistema operativo (Debian) en la VM

Relevancia de las versiones de Docker, Docker Compose y Makefile

Aspectos clave para que el proyecto funcione correctamente

Otros recursos

1. ¿Cómo funcionan Docker y Docker Compose?

Docker

Docker es una herramienta que te permite crear, ejecutar y gestionar contenedores. Un contenedor es como una caja ligera que contiene todo lo necesario para que una aplicación funcione: el código, las bibliotecas, las dependencias y la configuración, pero sin un sistema operativo completo como en una máquina virtual. Esto hace que sea muy rápido y eficiente.

Cómo funciona:

Primero, creas una imagen de Docker, que es como una plantilla con todas las instrucciones para instalar y configurar tu aplicación (por ejemplo, un servidor web como WordPress). Esto se define en un archivo llamado Dockerfile.

Luego, usas esa imagen para crear un contenedor, que es una instancia en ejecución de esa imagen. Puedes tener varios contenedores corriendo a la vez, cada uno con su propia copia de la aplicación.

Docker usa el sistema operativo del anfitrión (tu computadora o servidor) para compartir el kernel (el núcleo del sistema), lo que ahorra recursos.

Ejemplo práctico: Supongamos que quieres correr un sitio web de WordPress. Creas un Dockerfile que dice: "Instala un sistema basado en Debian, añade PHP y WordPress, y configura un servidor." Luego, con el comando docker build -t mi-wordpress ., generas la imagen. Después, con docker run -d -p 8080:80 mi-wordpress, inicias un contenedor que expone el sitio en el puerto 8080 de tu máquina.

Docker Compose

Docker Compose es una herramienta que te ayuda a gestionar múltiples contenedores al mismo tiempo, definiendo cómo trabajan juntos en un solo archivo (normalmente llamado docker-compose.yml). Es como un director de orquesta que coordina varios instrumentos (contenedores) para que toquen en armonía.

Cómo funciona:

Creas un archivo docker-compose.yml donde describes los servicios (contenedores) que necesitas, como un contenedor para WordPress, otro para la base de datos MariaDB, y otro para el servidor web Nginx.

Con un solo comando, docker-compose up, levantas todos los contenedores, y Docker Compose se encarga de las conexiones entre ellos (por ejemplo, que WordPress se conecte a MariaDB).

También puedes definir volúmenes (para guardar datos) y redes (para que los contenedores se comuniquen).

Ejemplo práctico: En tu proyecto Inception, tienes un docker-compose.yml que define tres servicios:

```
services:
 wordpress:
  build: ./requirements/wordpress
  volumes:
   - wordpress_data:/var/www/html
 mariadb:
  build: ./requirements/mariadb
  volumes:
   - mariadb_data:/var/lib/mysql
 nginx:
  build: ./requirements/nginx
  volumes:
   - wordpress_data:/var/www/html
volumes:
 wordpress_data:
 mariadb_data:
```

Con docker-compose up -d, se crean y conectan los tres contenedores, y los datos se guardan en volúmenes para persistir entre reinicios.

2. Diferencia entre una imagen de Docker usada con Docker Compose y sin Docker Compose

Sin Docker Compose:

Creas y gestionas imágenes y contenedores manualmente con comandos como docker build y docker run.

Cada contenedor se inicia por separado, y debes configurar las conexiones (redes, volúmenes) manualmente.

Ejemplo: Construyes una imagen con docker build -t mi-wordpress . y la ejecutas con docker run -d -p 8080:80 --name wordpress mi-wordpress. Si necesitas MariaDB, haces otro docker run para esa imagen y conectas manualmente las redes.

Con Docker Compose:

La imagen se define en el docker-compose.yml (o se construye desde un Dockerfile especificado), y Docker Compose automatiza la creación y conexión de los contenedores.

Todo se gestiona con un solo archivo y el comando docker-compose up, lo que simplifica la orquestación.

Ejemplo: En tu proyecto, las imágenes srcs-wordpress, srcs-mariadb, y srcs-nginx se construyen y se conectan automáticamente según el docker-compose.yml, sin necesidad de comandos individuales.

Diferencia clave:

Sin Docker Compose, tienes que hacer todo paso a paso. Con Docker Compose, defines todo en un archivo y lo ejecutas de una vez, ideal para aplicaciones con múltiples componentes como WordPress (que necesita un servidor web y una base de datos).

3. Beneficio de Docker comparado con máquinas virtuales (VMs)

Las máquinas virtuales (VMs) y Docker son herramientas diferentes para aislar aplicaciones, pero Docker tiene ventajas importantes:

Ligereza y velocidad:

Una VM incluye un sistema operativo completo (como Ubuntu o Windows), lo que la hace pesada (varios GB) y lenta de iniciar.

Docker usa el sistema operativo del anfitrión, por lo que los contenedores son mucho más ligeros (MB) y arrancan en segundos.

Ejemplo: Una VM de WordPress con Ubuntu puede pesar 2-3 GB, mientras que un contenedor Docker con WordPress pesa menos de 500 MB.

Eficiencia de recursos:

Una VM usa RAM y CPU para todo el sistema operativo, incluso si no lo necesitas. Docker comparte el kernel del anfitrión, usando solo los recursos de la aplicación.

Ejemplo: Puedes correr 10 contenedores Docker en la misma máquina con 8 GB de RAM, mientras que con VMs solo cabrían 2-3.

Portabilidad:

Un contenedor Docker incluye todo lo necesario (dependencias, configuraciones) en una imagen, por lo que funciona igual en cualquier máquina con Docker instalado.

Una VM depende de la configuración del hipervisor (como VirtualBox) y puede fallar si el sistema anfitrión cambia.

Ejemplo: Subes tu imagen mi-wordpress a un servidor y funciona sin ajustes; una VM podría requerir reinstalar drivers.

Desventaja de VMs:

Las VMs son mejores para aislar sistemas operativos completos (por ejemplo, correr Windows en una máquina Linux), mientras que Docker es ideal para aplicaciones específicas.

Resumen con ejemplo: Imagina que quieres una pastelería. Con una VM, construyes una cocina completa (horno, nevera, etc.) para cada pastel. Con Docker, usas una sola cocina compartida y solo preparas los ingredientes de cada pastel en contenedores separados, ahorrando espacio y tiempo.

4. Pertinencia de la estructura de directorios requerida para este proyecto

La estructura de directorios para Inception (como se detalla en el PDF del sujeto) está diseñada para organizar los archivos de manera lógica, facilitar la gestión con Docker Compose, y cumplir con los requisitos del proyecto 42. Vamos a explicarlo con un ejemplo basado en tu proyecto.

Estructura típica (según el PDF):

```
Inception/
├── srcs/
│   ├── .env
│   ├── docker-compose.yml
│   ├── requirements/
│   │   ├── wordpress/
│   │   │   ├── Dockerfile
│   │   │   ├── conf/
│   │   │   │   └── www.conf
│   │   │   └── tools/
│   │   │       └── wp.sh
│   │   ├── mariadb/
│   │   │   ├── Dockerfile
│   │   │   └── tools/
│   │   │       └── mariadb.sh
│   │   ├── nginx/
│   │   │   ├── Dockerfile
│   │   │   └── conf/
│   │   │       └── nginx.conf
├── data/
│   ├── wordpress/
```

```
│   └── mysql/
└── Makefile
```

Explicación de cada parte:

/srcs/:

Contiene el archivo principal docker-compose.yml, que define los servicios (wordpress, mariadb, nginx).

El archivo .env guarda las variables de entorno (como contraseñas y URLs) para que sean reutilizables y seguras.

Pertinencia: Centraliza la configuración y permite gestionar todo el proyecto desde un solo lugar.

/srcs/requirements/:

Cada subdirectorio (wordpress, mariadb, nginx) contiene un Dockerfile que especifica cómo construir las imágenes para cada servicio.

Los directorios conf/ y tools/ incluyen archivos de configuración (como www.conf para PHP-FPM o nginx.conf) y scripts (como wp.sh para instalar WordPress).

Pertinencia: Separa las configuraciones específicas de cada servicio, haciendo el código modular y fácil de mantener. Por ejemplo, wp.sh automatiza la instalación de WordPress, lo que cumple con los requisitos del proyecto.

/data/:

Contiene los subdirectorios wordpress/ y mysql/ para almacenar datos persistentes (volúmenes) que sobreviven al reinicio de los contenedores.

Pertinencia: Garantiza que los datos de la base de datos y los archivos de WordPress no se pierdan, lo cual es esencial para un entorno funcional como el que exige Inception.

Makefile:

Incluye comandos como fclean para limpiar el entorno Docker, build para construir imágenes, y up para iniciar los contenedores.

Pertinencia: Facilita la automatización de tareas, como las requeridas por 42 para evaluar tu capacidad de gestionar el proyecto desde la línea de comandos.

Ejemplo de uso:

Imagina que estás cocinando una paella. /srcs/requirements/ es como las recetas (Dockerfiles) para el arroz, el pollo y el marisco. /data/ es el refrigerador donde guardas los ingredientes preparados. docker-compose.yml es el plan de cocina que coordina

todo, y el Makefile es tu lista de tareas (limpiar la cocina, cocinar, servir). Esta estructura te ayuda a mantener todo organizado y cumplir con los pasos del proyecto.

Por qué es pertinente:

Organización: Evita mezclar configuraciones y facilita la colaboración o la corrección.

Cumple requisitos: 42 exige esta estructura para evaluar tu comprensión de Docker y la gestión de proyectos.

Escalabilidad: Si añades más servicios (e.g., un servidor de correo), puedes expandir /srcs/requirements/ sin caos.

Resumen final

Docker crea contenedores ligeros a partir de imágenes, y Docker Compose los orquesta juntos con un archivo YAML.

Diferencia: Sin Compose, gestionas contenedores manualmente; con Compose, todo se define y ejecuta automáticamente.

Beneficio sobre VMs: Docker es más rápido, usa menos recursos y es más portátil.

Estructura de directorios: Organiza el proyecto, cumple con los requisitos de 42, y facilita la gestión y persistencia de datos.

Relevancia de las versiones del sistema operativo (Debian) en la VM

El sistema operativo de la máquina virtual (VM) donde se ejecuta el proyecto, como Debian, juega un papel importante porque afecta la compatibilidad y el rendimiento de los contenedores. En el código adjunto, se usa FROM debian:bullseye en los Dockerfile de los servicios (wordpress, mariadb, nginx), lo que significa que las imágenes se basan en la versión "bullseye" de Debian (lanzada en agosto de 2021).

Compatibilidad de paquetes: Las versiones de Debian determinan qué versiones de software (como PHP, MariaDB, o Nginx) están disponibles en los repositorios. Por ejemplo, "bullseye" incluye PHP 7.4, que es compatible con WordPress, pero versiones más nuevas como "bookworm" (2023) podrían ofrecer PHP 8.x, lo que podría requerir ajustes en el código adjunto (e.g., actualizar php7.4-fpm a php8.2-fpm en el Dockerfile de wordpress).

Seguridad y soporte: Versiones antiguas de Debian (como "buster", 2019) pueden tener paquetes obsoletos o sin soporte, lo que podría introducir vulnerabilidades. "Bullseye" está en soporte activo hasta 2026, lo que es adecuado para este proyecto.

Rendimiento: Una VM con una versión más ligera o optimizada de Debian (e.g., sin interfaz gráfica) puede mejorar el rendimiento al ejecutar múltiples contenedores.

Qué tener en cuenta:

Coherencia: Asegúrate de que la versión de Debian en la VM sea compatible con la versión usada en los Dockerfile (bullseye). Si la VM usa "bookworm", podrías necesitar ajustar los paquetes instalados o enfrentarte a errores de dependencia.

Actualizaciones: Mantén el sistema y los paquetes actualizados con apt update && apt upgrade en la VM para evitar conflictos con los contenedores.

Pruebas: Si cambias a otra versión (e.g., de bullseye a bookworm), prueba el código adjunto completo (con docker-compose up) para verificar que los servicios (wordpress, mariadb, nginx) sigan funcionando.

Ejemplo:

Si la VM usa Debian "buster" y el Dockerfile de wordpress dice RUN apt install -y php7.4-fpm, podrías obtener un error porque "buster" solo tiene PHP 7.3. Deberías cambiar a FROM debian:bullseye en todos los Dockerfile y reconstruir con docker-compose up --build.

Relevancia de las versiones de Docker, Docker Compose y Makefile

Las versiones de Docker, Docker Compose y el Makefile son críticas porque determinan cómo se construyen, ejecutan y gestionan los contenedores y el flujo de trabajo del proyecto.

Docker

Relevancia: La versión de Docker (e.g., 20.10.5 en el código adjunto) afecta la compatibilidad con las características usadas en los Dockerfile y el docker-compose.yml. Por ejemplo, versiones más antiguas podrían no soportar ciertas opciones de montaje de volúmenes o redes definidas en el archivo YAML.

Qué tener en cuenta:

Compatibilidad: Asegúrate de que la versión de Docker sea reciente (mínimo 20.10.x) para evitar problemas con comandos como docker build o docker network. Si usas una versión más nueva (e.g., 24.x), verifica que las opciones en el Dockerfile (como COPY o RUN) sean compatibles.

Actualización: Actualiza Docker con sudo apt update && sudo apt install docker.io en la VM para obtener las últimas correcciones de errores.

Pruebas: Si actualizas Docker, prueba el código adjunto con docker-compose up -d para asegurarte de que los contenedores (wordpress, mariadb, nginx) se inicien correctamente.

Docker Compose

Relevancia: La versión de Docker Compose (e.g., 2.29.2 en el código adjunto) define cómo se interpreta el docker-compose.yml. Versiones antiguas podrían no soportar la sintaxis moderna (como build: o volumes:) usada en el archivo adjunto.

Qué tener en cuenta:

Versión del archivo: El código adjunto no especifica version: en docker-compose.yml, lo que usa la versión predeterminada de Docker Compose. Si usas una versión muy antigua (e.g., 1.x), podrías necesitar agregar version: '3' y ajustar la sintaxis.

Actualiza Docker Compose con:

sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose && sudo chmod +x /usr/local/bin/docker-compose.

Pruebas: Después de actualizar, ejecuta docker-compose --version y prueba con docker-compose up --build -d para verificar que los servicios se levanten sin errores.

Makefile

Relevancia: El Makefile (con un comando como fclean en la línea 20) automatiza tareas como limpiar el entorno Docker. La versión o estructura del Makefile no cambia directamente, pero su compatibilidad depende de los comandos de Docker y Docker Compose disponibles en la VM.

Qué tener en cuenta:

Comandos válidos: Asegúrate de que los comandos en el Makefile (e.g., docker stop, docker rm) sean compatibles con la versión de Docker instalada. Como vimos, el error en make fclean ocurrió porque no había contenedores que detener, lo que sugiere que el Makefile debe manejar casos vacíos.

Ajustes: Modifica el Makefile para agregar verificaciones condicionales, como:fclean:
@if [-n "$$(docker ps -qa)"]; then docker stop $$(docker ps -qa); fi
@if [-n "$$(docker ps -qa)"]; then docker rm $$(docker ps -qa); fi
@if [-n "$$(docker images -qa)"]; then docker rmi -f $$(docker images -qa); fi

Pruebas: Después de corregir, ejecuta make fclean y verifica que no haya errores, incluso si no hay recursos Docker.

Ejemplo con el código adjunto:

Si usas Docker 19.x (más antiguo que 20.10.5), el comando docker build en el Dockerfile de wordpress podría fallar si usa sintaxis moderna. Actualiza a 20.10.5 con sudo apt install docker.io=20.10.5-0ubuntu1~20.04.1 y prueba con docker-compose up.

Si Docker Compose es 1.29.2 (anterior a 2.29.2), el docker-compose.yml podría no reconocer build: ./requirements/wordpress. Actualiza a 2.29.2 y reconstruye.

Si el Makefile no verifica la existencia de contenedores antes de docker stop, fallará como en el código adjunto. Aplica las correcciones y usa make fclean sin problemas.

Aspectos clave para que el proyecto funcione correctamente

Coherencia de versiones:

Asegúrate de que la versión de Debian en los Dockerfile (bullseye) sea compatible con la VM. Usa cat /etc/debian_version en la VM para verificar y ajusta si es necesario.

Verifica las versiones de Docker (docker --version) y Docker Compose (docker-compose --version) con las mínimas requeridas por el proyecto (e.g., 20.10.5 y 2.29.2).

Pruebas exhaustivas:

Después de cambiar cualquier versión (Debian, Docker, Compose), ejecuta el flujo completo: make fclean, docker-compose up --build -d, y docker logs <servicio> para cada servicio (wordpress, mariadb, nginx).

Ejemplo: Si actualizas Docker a 24.x, prueba que el volumen wordpress_data:/var/www/html siga funcionando en el docker-compose.yml.

Gestión de dependencias:

En el Dockerfile de wordpress, RUN apt install -y php7.4-fpm depende de los repositorios de bullseye. Si cambias a bookworm, actualiza a php8.2-fpm y verifica con docker-compose up.

Asegúrate de que las variables en .env (e.g., WORDPRESS_DB_HOST=mariadb) sean consistentes con las versiones de los servicios.

Documentación y retrocompatibilidad:

Anota las versiones usadas (e.g., Debian bullseye, Docker 20.10.5) en un archivo README.md para que otros (o tú en el futuro) sepan qué esperar.

Si el proyecto se evalúa en una VM con versiones diferentes, ajusta el Makefile o los Dockerfile para adaptarse (e.g., agregar apt update antes de instalar paquetes).

Resolución de errores:

Si un cambio causa fallos (e.g., error de paquete en el Dockerfile), revisa los logs con docker logs wordpress y corrige la versión o el comando (e.g., instalar una versión específica de PHP con apt install -y php7.4-fpm=7.4.3-4ubuntu2).

Resumen final

Debian en la VM: Afecta la compatibilidad de paquetes y el soporte. Usa bullseye en el código adjunto y verifica la VM con cat /etc/debian_version.

Docker y Docker Compose: Versiones antiguas pueden romper la sintaxis del docker-compose.yml o Dockerfile. Mantén 20.10.5 y 2.29.2 como mínimo y actualiza con cuidado.

Makefile: Asegúrate de que maneje casos vacíos para evitar errores en make fclean. Ajusta con verificaciones condicionales.

Acciones clave: Prueba cada cambio, verifica logs, y documenta las versiones para garantizar que el proyecto funcione en cualquier entorno.

OTROS RECURSOS

Contenedores de Docker | ¿Qué es Docker? | AWS
Los contenedores de Docker facilitan la ejecución de más código en cada servidor, mejorando su uso y ahorrándole dinero. Puede utilizar los contenedores de Docker como bloque de construcción principal a la hora de crear aplicaciones y plataformas modernas. Docker facilita la creación y ejecución de arquitecturas de microservicios distribuidos, la implementación de código con canalizaciones de integración y entrega continuas estandarizadas, la creación de sistemas de procesamiento de datos altamente escalables y la creación de plataformas totalmente administradas para sus desarrolladores. La colaboración reciente entre AWS y Docker facilita la implementación de artefactos de Docker Compose en Amazon ECS y AWS Fargate. Cree y escale arquitecturas de aplicaciones distribuidas al utilizar las implementaciones de código estandarizadas que los contenedores de Docker proporcionan.

aws.amazon.com
Youtube

youtube.com

YouTube

m.youtube.com

Tutorial de Docker: introducción a la popular plataforma de contenedores

En nuestro tutorial de Docker, explicamos la plataforma de virtualización Docker y te brindamos instrucciones fáciles de seguir sobre cómo utilizarla en tu sistema Ubuntu 22.04. VPS gratis Prueba un servidor virtual de forma gratuita durante 30 días · ¡Prueba tu servidor virtual durante 30 días! Si lo solicitas, te reembolsaremos todos los gastos incurridos. ... Con el eslogan “Build, Ship and Run Any App, Anywhere” la plataforma de contenedores de código abierto Docker promociona una alternativa flexible y eficiente a la emulación de componentes de hardware basada en máquinas virtuales (VM). A diferencia de la virtualización tradicional de hardware, que implica iniciar diferentes sistemas huésped en un mismo sistema anfitrión (host), Docker permite que las aplicaciones se ejecuten como procesos aislados dentro del mismo sistema gracias a los contenedores.

ionos.es

Qué es Docker: Una Guía Completa

Antes de Docker, las empresas solían utilizar máquinas virtuales (VM) para ejecutar aplicaciones. Éstas pueden emular ordenadores físicos, permitiendo a los desarrolladores convertir un servidor en varios. Sin embargo, este enfoque puede tener algunas desventajas. Cada máquina virtual contiene una copia completa del sistema operativo y de la aplicación, así como los binarios y bibliotecas necesarios. Estos archivos pueden ocupar decenas de GB en un ordenador. Además, la virtualización del hardware para un SO invitado puede requerir una sobrecarga considerable. En lugar de virtualizar el hardware, los contenedores virtualizan el SO. En Docker, los contenedores son abstracciones en la capa de aplicaciones que pueden contener tanto código como dependencias. En la misma máquina, varios contenedores pueden ejecutarse como procesos aislados: Comparación de Docker y las máquinas virtuales (Fuente: ResearchGate) Como resultado, los contenedores Docker suelen ocupar menos espacio.

kinsta.com

r/docker on Reddit: La Serie de Vídeos Definitiva sobre Docker — Dockerfile en 203 Segundos

¡Espero que estéis todos bien! La semana pasada, comenzamos nuestra serie educativa centrada en Docker. Tras la positiva respuesta que recibimos a nuestra entrada de blog “Hoja de trucos definitiva de Docker”, hemos decidido ampliar nuestro contenido a formato vídeo. Nuestra última incorporación, “Dockerfile en 203 Segundos”, pretende ofrecer información sobre: ... Nos encantaría conocer vuestras opiniones sobre este nuevo vídeo. Vuestros comentarios influirán en nuestros futuros proyectos y nos ayudarán a desarrollar materiales más relevantes y útiles para todos aquí. ... Absolutamente brillante. Se puede reducir la música de fondo y hacerlo un poco más rápido, como el canal de YouTube de Fireship.

reddit.com

¿Qué es Docker y cómo funciona? Ventajas de los contenedores Docker

Estas herramientas están diseñadas a partir de los contenedores de Linux, por eso la tecnología Docker es sencilla y única. Además, ofrecen a los usuarios acceso sin precedentes a las aplicaciones, la posibilidad de realizar implementaciones en poco tiempo y el control sobre las versiones y su distribución. ... Si bien ambos conceptos suelen confundirse, no son iguales. Originalmente, Docker se diseñó a partir de la tecnología LXC, la cual por lo general se asocia con los contenedores "tradicionales" de Linux, pero desde aquel entonces se ha ido alejando de esa dependencia. LXC funcionaba como una virtualización ligera, pero no ofrecía una buena experiencia al desarrollador ni al usuario. La tecnología Docker no solo ofrece la capacidad para ejecutar los contenedores, sino que también facilita su creación y diseño, así como el envío y el control de versiones de las imágenes, entre otras funciones.

redhat.com

Cómo aprender Docker desde cero: Guía para profesionales de los datos | DataCamp

Para los que prefieren el aprendizaje visual, los tutoriales de YouTube del canal oficial de Docker y de otros educadores ofrecen contenidos útiles y accesibles para reforzar tu comprensión. Participar en la comunidad Docker es otra forma poderosa de avanzar en tu aprendizaje. Participa en foros como Docker Community Forums y Reddit para intercambiar ideas, hacer preguntas y compartir experiencias con otros estudiantes y profesionales. · Asistir a reuniones, seminarios web o conferencias sobre Docker también es una forma estupenda de estar al día de las tendencias del sector y ampliar tu red de contactos. Contribuir a proyectos de código abierto que utilizan Docker puede ser una forma fantástica de adquirir experiencia práctica en el mundo real. · Sitios web como GitHub ofrecen oportunidades para colaborar en proyectos relacionados con Docker, lo que te permite aprender de otros y aplicar tus habilidades en diversos contextos.

datacamp.com

5 canales de YouTube sobre “unboxing” en español para niños | Common Sense Media

¡Llegaron las reseñas en español de películas y series de televisión de Common Sense Media!

commonsensemedia.org

Los 73 mejores canales de YouTube de programación en español

Descubre los 73 mejores canales de YouTube en castellano para aprender programación y desarrollo de software.

webreactiva.com

DOCKER De NOVATO a PRO! (CURSO COMPLETO EN ESPAÑOL) - YouTube

youtube.com

GitHub - brunocascio/docker-espanol: Un tutorial Docker en español. Basado en el libro Docker Cookbook de O'reilly

La idea de Docker, es la de crear aplicaciones/servicios que sean independientes y portables. Esto es, no importa que sistema operativo utilices o con que hardware cuentas, si puedes instalar docker,

entonces podras correr tus contenedores en él. Entre las ventajas de usar docker, se encuentra la de olvidarte de instalar dependencias (ejemplo nodejs, java, python, ruby, etc) dentro de tu host o servidor y sin utilizar máquinas virtuales. A lo largo de este documento, iremos viendo el ecosistema de Docker y como todo se relaciona. Docker posibilita ir de tu maquina local a producción, con tu aplicació lista. Ya no es necesario instalar en cada servidor dependencias o depender del sistema operativo. Solo basta con tener Docker instalado. La instalación puede seguirse desde la Documentación Oficial. Muestra las imagenes locales disponibles. ... Podemos eliminarlas con docker rmi <image-name>, siempre y cuando no tenga contenedores asociados (corriendo o no).

github.com

Guía de Docker para principiantes: cómo crear tu primera aplicación Docker

Docker solo compartirá los recursos de la máquina anfitriona para ejecutar sus entornos. Docker VS Virtual machines (Copyright to Docker blog) Esta herramienta realmente puede cambiar la vida diaria del desarrollador. Para responder esta pregunta de la mejor manera, he redactado una lista no exhaustiva de los beneficios que encontrará: Docker es rápido. A diferencia de una máquina virtual, tu aplicación puede iniciarse en pocos segundos y detenerse igual de rápido. Docker es multi-plataforma. Puedes lanzar tu contenedor en cualquier sistema. Los contenedores pueden construirse y destruirse más rápido que en una máquina virtual. Se acabaron las dificultades para configurar tu entorno de trabajo. Una vez configurado tu Docker, no tendrás que volver a instalar tus dependencias manualmente. Si cambias de ordenador o si un empleado se incorpora a tu empresa, solo tendrás que darle tu configuración.

freecodecamp.org

Aprende Docker ahora! curso completo gratis desde cero! - YouTube

Accede a todos mis cursos aquí y obtén un 10% de descuento para siempre en la suscripción con el cupón 'off10": https://academia.holamundo.io/bundles/acceso-...

youtube.com

Qué es Docker Compose y Cómo Usarlo | Tutorial Completo

Tutoriales/Tutoriales de Cloud para Administradores/Qué es Docker Compose y Cómo Usarlo | Tutorial Completo ... En el universo del desarrollo de software, la eficiencia y la simplicidad son pilares fundamentales para el éxito de cualquier proyecto. En este contexto, Docker Compose emerge como una herramienta revolucionaria, capaz de simplificar la gestión de · aplicaciones multi-contenedor. Pero, ¿qué hace exactamente Docker Compose y por qué se ha convertido en un aliado indispensable para desarrolladores y administradores de sistemas por igual? Docker Compose es una herramienta versátil que te permite definir y gestionar aplicaciones multi-contenedor de forma sencilla. Con Docker Compose, puedes describir la configuración de tu entorno de desarrollo en un archivo YAML, especificando los servicios, volúmenes y redes necesarios para tu aplicación.

imaginaformacion.com

Uso de Docker Compose para implementar varios contenedores - Azure AI services | Microsoft Learn

Este artículo muestra cómo implementar varios contenedores Azure AI. En concreto, aprenderá a usar Docker Compose para orquestar varias imágenes de contenedor de Docker en un único equipo host. El ejemplo de este artículo consiste en implementar un contenedor de Inteligencia de documentos y un contenedor de AI Vision lectura juntos. ... Docker Compose es una herramienta para definir y ejecutar aplicaciones de Docker de varios contenedores. En Compose, se usa un archivo YAML para configurar los servicios de la aplicación. Después, con un solo comando, se crean y se inician todos los servicios de la configuración. Este procedimiento requiere varias herramientas que se deben instalar y ejecutar localmente: Suscripción a Azure. Si no tiene una, cree una cuenta gratuita antes de empezar. Motor de Docker. Confirme que la CLI de Docker funciona en una ventana de consola.

learn.microsoft.com

Compose: tutorial aplicaciones Docker multicontenedor - IONOS España

Compose automatiza la gestión de contenedores, por lo que facilita el escalado e implementación de aplicaciones en Docker. En este tutorial, examinamos en profundidad la configuración y el uso de Docker Compose para que puedas aplicarlo en tu propio entorno de producción. Docker Compose se utiliza para gestionar aplicaciones y aumentar la eficiencia en el desarrollo de contenedores. La configuración se guarda en un único archivo YAML, lo que permite crear y escalar aplicaciones fácilmente. Docker Compose se utiliza a menudo para configurar un entorno local, pero también puede formar parte de un flujo de trabajo de continuous integration / continuous delivery (CI/CD). Los desarrolladores pueden definir una versión específica de contenedores para pruebas o para una fase específica del pipeline. Esto facilita la identificación de problemas y la corrección de errores antes de que pasen a producción.

ionos.es

Tutorial Docker Compose - Ander Fernández

Docker es una de las herramientas más útiles que debe aprender un Data Scientist. Pero hay veces que necesitas no uno, sino muchos contenedores trabajando conjuntamente: una base de datos y una aplicación, por ejemplo. En estos casos, se hace imprescindible saber qué es y cómo funciona Docker Compose. ¡Empezemos! Docker Compose es una extensión de Docker que permite trabajar con varios contenedores de forma simultánea haciendo que estos se conecten y relacionen entre sí de la forma en la que tú decidas. Docker Compose surge porque muchas aplicaciones requieren de más de un microservicio. Pero claro, la idea de Docker es que únicamente ejecute un único microservicio por contenedor y no varios a la vez. Por tanto en esta situación podemos hacer dos cosas: Ejecutar dos microservicios en un mismo Docker. Lo bueno en esta situación es que no necesitamos aprender nada para ello, simplemente con Docker podríamos lograrlo.

anderfernandez.com

Cómo instalar y usar Docker Compose en Ubuntu 20.04 | DigitalOcean

Para las aplicaciones que dependen de varios servicios, organizar todos los contenedores para que se inicien, comuniquen y se apaguen juntos puede convertirse rápidamente en algo difícil de manejar. Docker Compose es una herramienta que le permite ejecutar entornos de aplicación multi contenedor

según las definiciones establecidas en un archivo YAML. Se utilizan definiciones de servicio para crear entornos totalmente personalizables con varios contenedores que pueden compartir redes y volúmenes de datos. En esta guía, mostraremos cómo instalar Docker Compose en un servidor Ubuntu 20.04 y cómo empezar a usar esta herramienta. Para seguir los pasos de este artículo, necesitará lo siguiente: Acceso a un equipo local o servidor de desarrollo de Ubuntu 20.04 como usuario non root con privilegios sudo. Si utiliza un servidor remoto, se recomienda tener instalado un firewall activo.

digitalocean.com

¿Qué demonios es Docker y Docker-Compose? y cómo Dockerizar Dotnet Core WebApi y SQL Server en un ambiente de desarrollo ideal - DEV Community

A menudo que avanzamos en esta extensa curva de aprendizaje en el mundo de la ingeniería y desarrollo... Tagged with docker, dotnet, containers, csharp.

dev.to

El fichero docker-compose.yml - Docker

Por ejemplo para la ejecución de la aplicación Let’s Chat podríamos tener un fichero docker-compose.yml, dentro de una carpeta, con el siguiente contenido: version: '3.1' services: app: container_name: letschat image: sdelements/lets-chat restart: always environment: LCB_DATABASE_URI: mongodb://mongo/letschat ports: - 80:8080 depends_on: - db db: container_name: mongo image: mongo restart: always volumes: - /opt/mongo:/data/db · Puedes encontrar todos los parámetros que podemos definir en la documentación oficial. ... Es escenario está formado por services. Cada uno ello va a crear un contenedor. restart: always: Indicamos la política de reinicio del contenedor si por cualquier condición se para. Más información. depend on: Indica la dependencia entre contenedores. No se va a iniciar un contenedor hasta que otro este funcionando. Más información.

iesgn.github.io

Aprendiendo a utilizar Docker Compose

Docker Compose es una herramienta que permite simplificar el uso de Docker. A partir de archivos YAML es mas sencillo crear contendores, conectarlos, habilitar puertos, volumenes, etc. Aquí resumimos algunos tips. Con Compose puedes crear diferentes contenedores y al mismo tiempo, en cada contenedor, diferentes servicios, unirlos a un volúmen común, iniciarlos y apagarlos, etc. Es un componente fundamental para poder construir aplicaciones y microservicios. En vez de utilizar Docker via una serie inmemorizable de comandos bash y scripts, Docker Compose te permite mediante archivos YAML para poder instruir al Docker Engine a realizar tareas, programaticamente. Y esta es la clave, la facilidad para dar una serie de instrucciones, y luego repetirlas en diferentes ambientes. ... Mac OS: Ya viene instalado dentro de Docker for Mac. Con solo instalar Docker for Mac es suficiente.

dockertips.com

Despliegue de aplicaciones con Docker-Compose - Adictos al trabajo

En este tutorial vamos a aprender a desplegar nuestros proyectos en contenedores Docker de manera sencilla y rápida, utilizando docker-compose. ... Docker Compose es una herramienta de Docker que

orquesta contenedores en un mismo cliente. Consiste en un archivo de texto en formato YAML, que define de forma declarativa los contenedores que se van a desplegar, así como las dependencias entre ellos. Al utilizar el paradigma declarativo, en este documento sólo se especifican las características de los contenedores deseados, y no cómo se despliegan. Para el despliegue, se basa en la definición de servicios, que referencian imágenes Docker de un registro y las características de los contenedores que se desean desplegar. Este archivo normalmente tiene el nombre docker-compose.yml pero puede tener cualquier otro nombre, siempre que se especifique al ejecutar el comando docker-compose con el argumento -f.

adictosaltrabajo.com
GitHub - manufosela/introduccion-docker: tutorial introducción a docker y docker-compose creando un api node con mongo
Tutorial de introducción a docker y docker-compose creando una aplicación fullstack, con un contenedor node-express para el api, otro con mongodb y otro con nginx para servir los ficheros estáticos. ... Para este ejemplo se ha utilizado un sistema operativo Ubuntu, por lo que todos los ejemplos de instalación de programas se harán para este entorno. Puedes cambiarlo por los de tu sistema operativo. ... Cuidado con el copy-paste desde el README que puede hacer que no funcione, es preferible bajar el fichero del repo. 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 Fuente Wikipedia · Docker es un "emulador" de entornos aislado para poder ejecutar programas sin que afecte a mi sistema operativo (SO) y pudiendose llevar y replicar en otros SS.OO.

github.com
José Miguel Moreno – Medium

medium.com
Conoce los comandos básicos de Docker Compose | Coffee bytes
Mis ideas, opiniones, tutoriales y críticas relacionadas con el mundo del desarrollo web y la ingeniería de Software ... Docker compose nos permite crear aplicaciones con múltiples contenedores, estos contenedores interaccionarán y podrán verse entre sí. Para configurar cada uno de estos servicios usaremos un archivo en formato YAML (también le dicen YML). En este tutorial de docker compose te muestro algunos de los comandos más usados y lo que hace cada uno. Si quieres refrescar tu memoria visita mi tutorial de comandos básicos de Docker. Si quieres hostear una aplicación usando Docker o Kubernetes de manera económica checa Digital Ocean, puedes tener un VPS desde $4 usd el mes. Docker compose es una herramienta que te permite manejar aplicaciones que consisten en multiples contenedores de Docker . En lugar de tener múltiples Dockerfiles y estar ejecutando y vinculando uno por uno con Docker, definimos un archivo docker-compose.yml con la configuración que deseemos y lo ejecutamos, esto creará todos los servicios necesarios de nuestra aplicación.

coffeebytes.dev

Orquestar contenedores con Docker Compose - atareao con Linux
Con lo que has visto en los diferentes capítulos de este tutorial de Docker, ya estarás levantando contenedores a diestro y siniestro. Eso si, hasta el momento, solo puedes trabajar con contenedores aislados, contenedores que no se relacionan unos con otros. Ahora, ¿como levantar un WordPress? al fin y al cabo un WordPress está compuesto de diferentes piezas. Por un lado el servidor web, por otro lado la base de datos y por último, el lenguaje de programación. Podrías plantearte levantar un contenedor que contuviera todo esto... Sin embargo, ¿porque no levantar un contenedor para cada uno de estos servicios? En este nuevo capítulo del tutorial sobre docker, verás como puedes levantar varios contenedores y que estos se relacionen entre si. Es decir, vas a orquestar contenedores con docker-compose. Es posible que te plantees meter todos los servicios que necesitas en un único contenedor.

atareao.es

- Aprende Docker en Español playlist on YouTube
- DOCKER De NOVATO a PRO! complete course in Spanish on YouTube
- Docker tutorial introduction to popular container platform in Spanish on YouTube
- Docker Compose in 10 minutes tutorial in Spanish on YouTube
- Virtual machines vs Docker containers comparison in Spanish on YouTube
- Docker project directory structure tutorial in Spanish on YouTube
- Docker for web developers tutorial in Spanish on YouTube