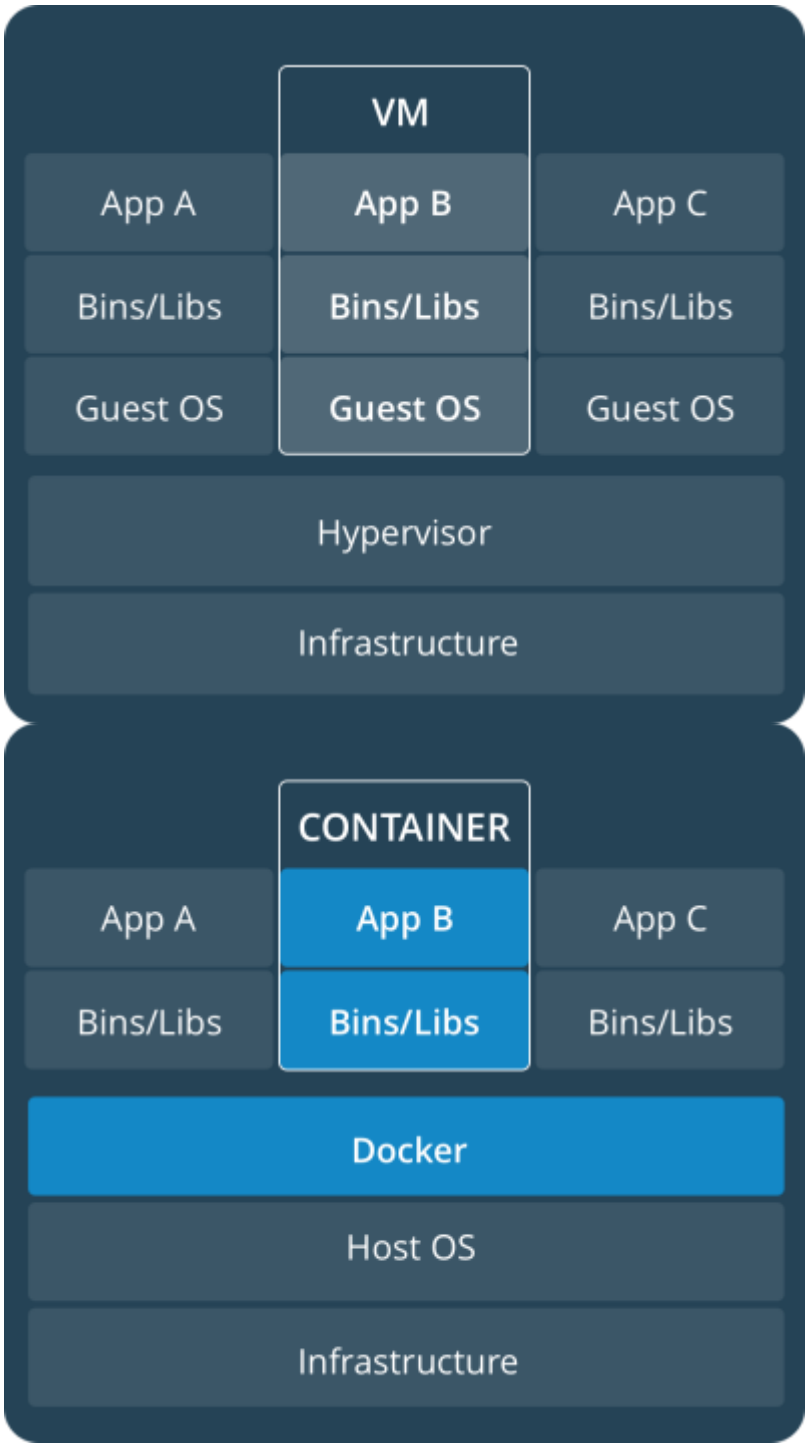
Docker

Differences entre virtualisation et container docker



Quelle est son histoire ?

En 2010, Solomon Hykes qui travaillait à ce moment-là chez dotCloud, une entreprise française proposant des solutions de PaaS (Platform as a Service), et deux de ces collègues, Andréa Luzzardi et François-Xavier Bourlet, ont décidé de fonder Docker, l'entreprise, et ont lancé le projet Docker, la

plateforme.

Le projet est devenu open source à partir de 2013, ce qui a contribué à la popularisation du système de conteneurisation pour devenir de nos jours la plateforme la plus utilisée dans ce domaine.

En 2019, l'entreprise Mirantis rachète la version Enterprise de Docker et reprend donc les clients et les employés de ce dernier.

Nous sommes actuellement à la version 4.48.0 (10/2025) de Docker Desktop pour Windows et MacOS, à la version 28.5.1 de Docker Engine et 2.40.1 en ce qui concerne Docker Compose.

Qu'est-ce qu'un conteneur Docker ?

Un conteneur Docker est un package logiciel autonome contenant toutes les dépendances nécessaires pour exécuter une application spécifique sur différents systèmes d'exploitation. L'image Docker dicte toutes les instructions de configuration pour démarrer ou arrêter des conteneurs. Un nouveau conteneur est créé chaque fois qu'un utilisateur exécute une image.

Pourquoi utiliser des conteneurs Docker ?

Les conteneurs Docker sont une révolution pour les développeurs d'aujourd'hui. Avec l'isolation des conteneurs, vos applications peuvent fonctionner sur le même système d'exploitation tout en restant séparées des autres systèmes d'exploitation et conteneurs. Cette fonctionnalité assure des performances cohérentes dans les environnements de développement et de préproduction.

Pour les entreprises, les conteneurs Docker améliorent la vitesse de déploiement et maximisent l'utilisation des ressources système. En termes de ressources, le déploiement de conteneurs Docker nécessite nettement moins de mémoire qu'une machine virtuelle. De plus, leur nature portable rend possible la migration et la mise à l'échelle d'applications héritées.

Pour résumer, les conteneurs Docker optimisent le processus de développement, économisant un temps précieux et contribuant au succès économique de vos projets de développement d'applications.

Les principaux avantages à utiliser Docker

1 - Utilisation plus efficiente des ressources système

Docker permet une utilisation plus efficiente des ressources d'un système tout en permettant les mêmes avantages qu'une machine virtuelle : c'est-à-dire principalement l'isolation et la reproductibilité.

Les applications conteneurisées utilisent beaucoup moins de ressources que des machines virtuelles car elles utilisent le même noyau Linux.

Elles démarrent et s'arrêtent en quelques millisecondes, alors qu'il faut quelques secondes ou minutes pour une machine virtuelle.

2 - Reproductibilité

Un conteneur Docker est garanti d'être identique quel que soit le système.

Il garantit que la bonne version de chaque dépendance soit installée. Ainsi, chaque membre d'une équipe est certain d'avoir le ou les applications qui fonctionnent identiquement sur des environnements différents (pas les mêmes OS, pas les même configurations locales, pas les mêmes dépendances globales etc).

Avec Docker, plus jamais vous entendrez le : "Moi, cela fonctionne sur ma machine".

Cela permet aux développeurs de se soucier uniquement du code et pas de l'environnement où celui-ci sera finalement exécuté.

3 - Isolation

Les dépendances ou les configurations d'un conteneur n'affecteront aucune dépendance ou configuration des autres conteneurs et de la machine hôte.

Vous pouvez ainsi avoir 5 versions de Node.js ou de MongoDB différentes localement sans aucune peine ! C'est un énorme plus lorsque l'on travaille sur plusieurs projets avec des versions de dépendances qui ne sont pas les mêmes. En effet, elles présentent souvent des incompatibilités ou des breaking changes se qui se révèlent être un véritable casse-tête !

4 - Mises à jour et tests

Pour ceux qui ont à gérer plusieurs serveurs, vous savez combien il est difficile de maintenir et de mettre à jour correctement des environnements complexes.

Il faut bien sûr mettre à jour l'OS sur chaque serveur, puis les environnements serveurs et les bases de données : le tout dans le bon ordre et sans oublier aucun serveur. Les risques sont élevés (downtimes, pertes de données etc).

Avec Docker, la mise à jour d'un environnement dans le cloud, même sur de nombreux serveurs est très simple. Vous gagnerez un temps précieux.

Même chose si vous devez gérer plusieurs environnements : par exemple le classique development / tests, staging et production. Il faut être sûr que tous les environnements aient les mêmes dépendances, les mêmes versions d'OS etc pour que les tests soient fiables. Avec Docker toute cette gestion d'environnements est grandement simplifiée !

5 - Bonnes pratiques grâce aux images open-source

Docker est si populaire que tous les mainteneurs de librairies ou de logiciels divers (environnements

serveurs, serveurs Web, bases de données etc) maintiennent des images Docker.

Ces images Docker officielles, disponibles sur une plateforme que nous étudierons, Docker Hub, permettent de lancer des conteneurs fiables et sécurisés très rapidement et sans avoir à les configurer soit même.

Vision globale du processus Docker

Une image vaut mille mots, voici donc une illustration qui montre les différentes étapes de création d'un conteneur :

Le Dockerfile permet de créer une image. Cette image contient la liste des instructions qu'un conteneur devra exécuter lorsqu'il sera créé à partir de cette même image.

Liens web

Mise en place et utilisation de Docker FR

Apprendre Docker FR

Presentation de Docker FR

utilisation-docker.pdf

Cours Tp les dockerfiles FR

docker-fr.pdf