

# **L'utilisation des conteneurs pour gérer les mises à jour applicatives**

# Table des matières

<b>I. Conteneurisation applicative</b>	<b>3</b>
<b>II. Exercice : Quiz</b>	<b>6</b>
<b>III. Comment utiliser les conteneurs pour la mise à jour des applications</b>	<b>7</b>
<b>IV. Exercice : Quiz</b>	<b>11</b>
<b>V. Essentiel</b>	<b>11</b>
<b>VI. Auto-évaluation</b>	<b>12</b>
A. Exercice .....	12
B. Test .....	12
<b>Solutions des exercices</b>	<b>13</b>

# I. Conteneurisation applicative

**Durée :** 1 h

**Environnement de travail :** PC connecté à Internet

## Contexte

Le *cloud computing* fait partie des plus grandes révolutions technologiques de ces dernières décennies. Il est adopté par de nombreuses entreprises en quête de performance et d'une meilleure expérience utilisateur. Ces acteurs s'en servent pour déployer leur application sur des serveurs distants plus sécurisés, plus accessibles et surtout moins contraignants en maintenance.

En réalité, le besoin de migration des applications hébergées sur les clouds suscite un problème d'incompatibilité entre les environnements de développement. Une application conçue sur Windows devient inexploitable dans un environnement Linux et inversement.

C'est dans ce contexte qu'est apparue la technologie de conteneurisation des applications. Cette solution permet d'utiliser les conteneurs pour séparer et combiner des codes en des blocs exécutables autonomes. Avec une telle technologie, les processus de développement sont devenus plus efficaces et efficients. Adieu les difficultés de portabilité, de déploiement sous différents environnements et d'incohérences de codes. Pour le développeur, c'est surtout un gain de temps salvateur puisqu'il peut compter sur un répertoire d'images de conteneurs déjà prêts pour assurer ses projets.

Toutefois, il est indispensable de maîtriser les bonnes pratiques et les méthodes appropriées pour développer et surtout faire évoluer une application basée sur la conteneurisation. Cette expertise s'avère indispensable pour contourner les défis sécuritaires que présente l'architecture des plateformes de conteneurisation.

Nous allons nous atteler à cet exercice pour vous aider à réussir de la bonne manière la mise à jour des applications basées sur les conteneurs.

## Définition Qu'est-ce qu'un conteneur applicatif ?

En entendant ce mot, on pense en premier lieu à ces caissons métalliques qui sont utilisés pour le transport des marchandises par bateau. Mais ici, nous sommes dans l'univers informatique et le conteneur signifie autre chose - bien que l'élément physique puisse donner un aperçu de ce dont on parle.

Le conteneur applicatif est l'équivalent d'une boîte ou d'une enveloppe numérique. Il stocke un processus exécutable ou un fichier de code, de bibliothèque et de dépendance. Ce sont des instruments révolutionnaires de développement logiciel apparus aux alentours de 2013. Ils ont été portés par la société Docker, qui a été la première à œuvrer pour sa démocratisation.

Aujourd'hui, le grand public les connaît comme des composants qui facilitent la migration des codes, des librairies et la configuration des environnements de développement.

## Les différents types de conteneurs applicatifs

Il existe différents types de conteneurs applicatifs. Ils sont identifiés par la nature de l'élément embarqué en leur sein. Ainsi, on retrouve :

- Les codes,
- Les fichiers de configuration,
- Les dépendances d'application,
- Les bibliothèques,
- Les variables d'environnement.

### Définition Qu'est-ce que la conteneurisation ?

Au sens strict du terme, la conteneurisation est un processus qui consiste à **encapsuler des codes** dans un conteneur. Elle regroupe des composants de programmes informatiques en des noyaux isolés, indépendants, légers et exécutables sur différentes machines et dans des environnements variés. Ces noyaux sont appelés « *image de conteneur exécutable* ». Ces images fonctionnent comme des machines virtuelles composées des librairies, des dépendances et des fichiers de configuration dont une application a besoin pour fonctionner.

Au sens large du terme, la conteneurisation fait également référence à toute activité ayant rapport avec le déploiement, la surveillance, la maintenance, en quelque sorte l'orchestration des conteneurs.

### L'orchestration de conteneur

L'orchestration de conteneur est la technologie qui permet d'automatiser la création, le déploiement et la gestion évolutive des conteneurs. C'est une technologie qui devient nécessaire au fur et à mesure que le nombre de conteneurs dans le système augmente. En réalité, elle vient réduire la complexité des tonnes d'opérations que la gestion d'un tel réseau peut impliquer. Sur le marché, elle se décline en une large gamme d'outils, dont les plus connus sont :

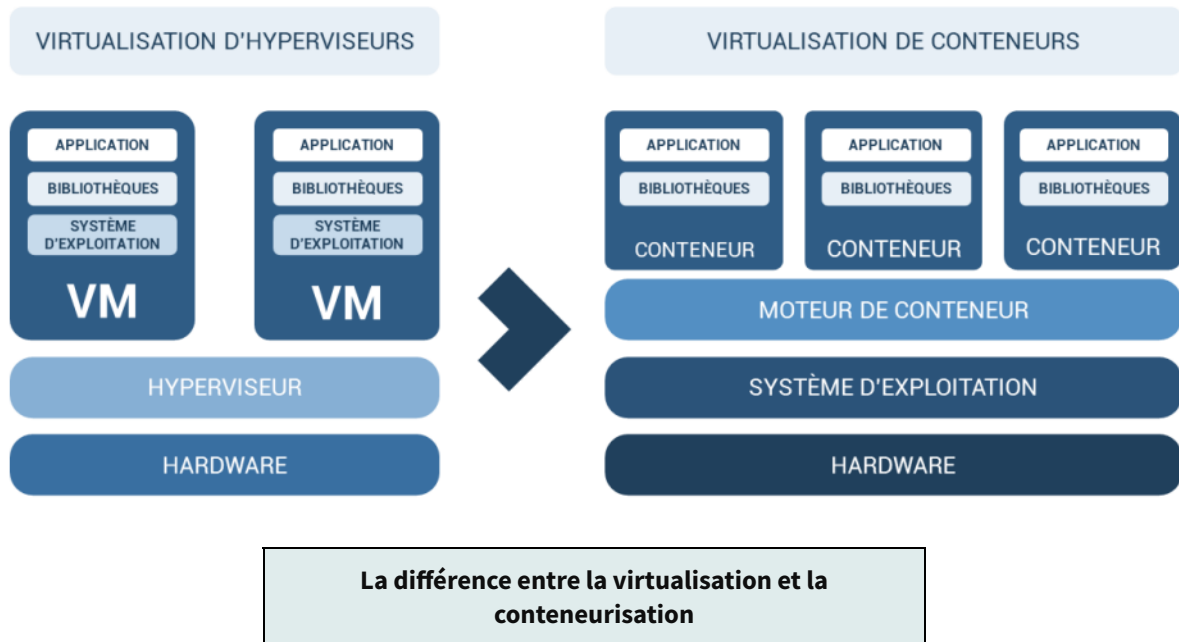
- Kubernetes,
- Docker Swarm,
- Apache Mesos.

### Différence entre conteneur et Machine Virtuelle (VM)

Il existe un amalgame autour de la relation entre le concept de conteneur et celui de machine virtuelle. Dans l'assertion populaire, ces notions sont confondues. En effet, d'une part les 2 technologies reproduisent un environnement virtuel dans lequel les logiciels peuvent être déployés. D'autre part, toutes deux permettent d'exploiter davantage les potentiels du processeur ou de la mémoire d'une machine. Hélas, en dehors de ces points communs, les 2 notions sont foncièrement divergentes.

Les machines virtuelles sont des programmes capables de reproduire sur une machine physique un système d'exploitation virtuel avec tous ses utilitaires. L'environnement qu'elles créent est indépendant et distinct de celui installé initialement sur la machine. Ainsi, avec les VM, on peut profiter de plusieurs systèmes d'exploitation sur une même machine. Elles utilisent des couches logiciel appelées **hyperviseurs** pour lire et s'allouer des ressources (mémoire et processeur) sur la machine hôte. De plus, chaque exécution d'application sollicite les ressources de l'environnement hôte. Il s'agit donc d'un fonctionnement qui engendre une forte consommation de ressources machine.

Avec les conteneurs, c'est un tout autre processus qui se met en place. Tout d'abord, les conteneurs sont directement déployés sur l'environnement initial d'une machine. Ils partagent donc le noyau du système d'exploitation principal de l'hôte. De même, les programmes conteneurisés s'exécutent directement sur ce dernier. Cela aurait pu submerger les ressources machine, mais heureusement les conteneurs sont légers. Ils n'exécutent que quelques tâches spécifiques contrairement aux machines virtuelles. En clair, plus que les VM, les conteneurs permettent d'exploiter le potentiel des machines sans que leur fonctionnement ne soit sujet à des pannes.



### Les avantages de la conteneurisation

C'est désormais une évidence, la conteneurisation applicative renvoie à la complexification des processus de création et de déploiement des applications. Avec cette solution, on peut correctement développer un logiciel sous un environnement quelconque puis le déployer sous un autre environnement sans risquer de faire face à des échecs d'exécution et à des pannes de non-compatibilité.

On résume souvent la conteneurisation à ses atouts majeurs. Mais derrière ces éléments se trouve d'autres points tout aussi importants. Il s'agit :

- Du **faible coût de déploiement** des solutions basées sur les conteneurs. Les conteneurs ne contiennent pas les fichiers des systèmes d'exploitation, ce qui se traduit par une faible taille mais surtout par des traitements en moins. Résultat, ils sollicitent moins de ressources en matière de mémoire et de puissance de calcul de la machine hôte. De plus, ils donnent la possibilité à différents projets d'exploiter les mêmes fichiers, codes et bibliothèques. C'est-à-dire qu'une image de conteneur spécifique peut être utilisée dans différentes applications. Tous ces points contribuent alors à alléger les coûts financiers d'un projet de déploiement applicative.
- De la **portabilité avancée** : le caractère isolé des conteneurs et leur indépendance par rapport au matériel ou à l'environnement leur garantit une transportabilité importante. Par exemple, un conteneur peut être déplacé d'un système macOS à Windows ou d'un ordinateur à serveur cloud et dans tous ces cas, il continue à fonctionner de la même manière.
- De l'**efficacité du processus de développement** : chaque conteneur est construit de façon objective pour atteindre un résultat spécifique. Ils améliorent donc la cohérence des codes, réduisent les erreurs dans le code et les risques de bug. Par exemple, si un conteneur venait à ne pas fonctionner, les autres continueraient leur opération normalement.
- De la **rapidité des processus de développement** : la création d'un conteneur est un exercice de quelques secondes. Tous les codes et les bibliothèques sont directement empaquetés, il suffit de faire les configurations appropriées pour le rendre exécutable. De plus, il peut être répliqué, déployé ou détruit en une minute. Les conteneurs permettent donc d'aller plus vite, que ce soit pour le déploiement, les tests ou les ajustements.
- De l'**évolutivité accrue** : la capacité à répliquer les conteneurs vient avec la possibilité de mettre rapidement une application à l'échelle. Il suffit de reproduire le conteneur et de lui apporter quelques modifications. Enfin, en composant le nouveau conteneur avec d'anciens, on obtient une nouvelle version de l'application.

### Complément Les cas d'utilisation des conteneurs

La conteneurisation peut être utilisée au même titre que la virtualisation. Cependant, il faut reconnaître qu'elle est plus efficace dans certains contextes que dans d'autres. C'est par exemple le cas de son utilisation pour :

- **Les microservices** : le principe de l'architecture *microserver* est le regroupement de composants natifs et autonomes pour en faire une solution unique. Cette notion est également partagée avec les conteneurs, ce qui les adapte pour l'occasion. Pour les entreprises, c'est surtout un cas d'utilisation qui permet de réduire les coûts de développement.
- **Le multi-cloud** : la portabilité avancée des conteneurs fait d'eux des solutions adaptées dans les environnements multi-cloud. À l'opposé d'une machine virtuelle, ils peuvent être déplacés d'un système à l'autre sans risque que des bugs surviennent.
- **L'automatisation** : la simplicité et la rapidité des processus de conteneurisation rendent le composant adapté à une mise à l'échelle horizontale et à une intégration continue.
- **La modernisation d'applications non conteneurisées** : les conteneurs peuvent être utilisés pour encapsuler différentes portions de code existantes et les rendre portables et exécutables dans différents environnements ou machines.
- **Le processus par lot** : il s'agit d'un modèle de traitement visant à permettre à plusieurs solutions isolées et indépendantes d'exploiter les mêmes ressources. C'est un modèle qui est parfaitement compatible avec le principe de partage de bibliothèque et des dépendances que proposent la conteneurisation.

### Exemple L'utilisation de la conteneurisation applicative en entreprise

Les conteneurs applicatifs sont largement adoptés dans les rangs des entreprises. C'est le cas par exemple de **PayPal**. En 2017, l'entreprise a présenté son plan de modernisation et de renforcement de la performance de ses services. La particularité d'un tel plan est d'être fortement centré sur l'utilisation des conteneurs applicatifs. 2 années après cette déclaration faite à l'occasion du DockerCon, l'architecture des services de PayPal pouvait compter plus de 150 000 composants applicatifs autonomes. C'est un pari gagné pour l'entreprise, qui est parvenue à adapter la performance de son infrastructure à une croissance de la base d'utilisateurs.

Le cas de PayPal n'est pas une exception. En 2019, c'est au tour de **Pipedrive** de se pencher sur l'idée d'une conteneurisation de sa plateforme. La rapidité d'exécution qu'a connu le projet permet de révéler au grand jour le potentiel de la conteneurisation à accélérer les processus de maintenance et de mise à l'échelle des solutions informatiques.

Nous pouvons également mentionner les expériences de la startup **Babylon Health** ou de la plateforme française **BlaBlaCar**.

Le premier à mis le processus de conteneurisation à profit pour transformer son moteur d'intelligence artificielle en une application portable. Elle a aussi utilisé la technologie des conteneurs pour réduire et optimiser la consommation en ressources mémoire ou processeurs des composants applicatifs de son infrastructure.

La plateforme de covoiturage, de son côté, s'en est servie pour accroître le cycle de vie de sa plateforme.

## Exercice : Quiz

[solution n°1 p.15]

### Question 1

Un conteneur est un système d'exploitation utilisé par une application pour reproduire avec exactitude son fonctionnement.

- ☐ Vrai
- ☐ Faux

### Question 2

Un conteneur est composé de :

- ☐ Codes
- ☐ Dépendances
- ☐ Librairies
- ☐ Les 3

Question 3

Les conteneurs applicatifs ont été inventés par Docker.

- ☐ Vrai
- ☐ Faux

Question 4

Qu'est-ce qu'un hyperviseur ?

- ☐ Un composant de caméra
- ☐ Un programme qui permet de faire une virtualisation
- ☐ Un gestionnaire de base de données distant

Question 5

Qu'est-ce que la virtualisation ?

- ☐ Une technique qui permet de créer des conteneurs applicatifs
- ☐ Une technique qui permet de produire une copie visuelle et fonctionnelle d'un système d'exploitation sur une machine hôte

### III. Comment utiliser les conteneurs pour la mise à jour des applications

Les conteneurs sont légers, agiles, moins complexes. Leur mise en place est donc un jeu d'enfant. Pourtant, beaucoup d'entreprises échouent à mettre en place des solutions fiables et efficaces en s'appuyant sur la technologie des conteneurs. La raison principale de cette situation regrettable est une méconnaissance des défis que présente la conteneurisation ainsi que des bonnes pratiques pour contourner ses problèmes.

#### Les défis de la conteneurisation

Au-delà des nombreux avantages précédents, l'utilisation des conteneurs présente aussi des défis sur plusieurs points. Le premier se réfère à la **complexité des plateformes d'orchestration** de conteneur. Ces solutions, en particulier celle *open source*, requiert des formations spécifiques pour réussir les configurations. À titre d'exemple, il est crucial de cerner les notions de *cluster* ainsi que les règles d'intégration des conteneurs dans un même réseau. Ne pas le faire, c'est prendre le risque de laisser des erreurs s'infiltrer dans l'architecture de l'application.

Le deuxième défi renvoie à la dualité entre le besoin d'implémenter des opérations persistantes et le **caractère éphémère des conteneurs**. En réalité, dans le logiciel conteneurisé, lorsque les opérations sont terminées, les conteneurs sont détruits avec leurs informations. Dès lors, il devient difficile de réaliser des tâches qui nécessitent l'utilisation des fichiers « *États* ».

Le dernier point est certainement le plus important. Il se rapporte au **niveau de sécurité** plus ou moins élevé des systèmes d'orchestration de conteneur. En effet, ces dispositifs sont composés d'éléments vulnérables comme l'API et d'autres outils de gestion. Comme un système d'exploitation, ces éléments peuvent faire l'objet d'intrusions au travers des connexions réseau ou des accès et privilèges.

Mais il faut aussi reconnaître qu'il est possible de pallier ce risque en adoptant certaines bonnes pratiques d'orchestration de conteneurs.

### Les bonnes actions pour réussir la conteneurisation

De plus en plus d'entreprises ont recours à la conteneurisation pour réduire les coûts de développement et profiter d'un développement plus productif. Mais face aux défis importants que présente la technologie, elles échouent à optimiser leur application sur le plan sécuritaire. Pourtant, il existe certaines pratiques qui permettent de garantir la fiabilité du processus. Entre autres, on retient qu'il faut :

- **Encapsuler une application par conteneur** : le conteneur n'est pas une machine virtuelle. Même s'il peut exécuter plusieurs opérations comme la VM, cette utilisation lui est peu appropriée. De plus, en raison du cycle de vie éphémère des conteneurs, il serait contre-productif de les garnir de processus dont ils n'ont pas forcément besoin à chaque exécution.
- **Maintenir les images des conteneurs petites et simples** : plus une image de conteneur est légère, plus rapide sera son chargement et meilleure s'en trouvera sa performance. Il est donc important d'éviter de surcharger son image en couches, en bibliothèque ou en fichiers de dépendances.
- **Rester pertinent dans le choix des balises d'images de conteneur** : le tag des images de conteneur a pour rôle d'identifier la version de l'exécutable. Mieux, il permet aux machines de reconnaître le mode d'exécution à implémenter pour chaque paquet. Par exemple, avec le tag « *stable* », on invite la machine à exploiter le paquet de façon à conserver l'image initiale. Par contre, le tag « *unique* » invite à un processus de déploiement.
- **Exploiter des données persistantes non encapsulées dans les conteneurs** : le stockage des données est crucial pour pallier les limites que peuvent engendrer la destruction d'un conteneur et de son contenu. Cependant, la diffusion des données dans les couches de stockage des conteneurs est inappropriée. Le faire reviendrait à alourdir les conteurs, car ces stockages croissent en volume au fil de l'exécution des processus. En conséquence, ce choix peut réduire le gain de performance et même déclencher des pannes. La seule solution qui reste est alors l'utilisation de stockage persistant.
- **Profiter de la puissance d'autonomisation des orchestrateurs** : il est évident que plus le nombre de conteneurs dans un système augmente, plus sa gestion se complexifie. C'est là que les solutions d'orchestration interviennent. Elles permettent de préprogrammer l'implémentation des différentes tâches de création, de déploiement, de destruction et de mise à jour des conteneurs ou des clusters.
- **Sécuriser l'exécution des conteneurs** : la seule manière d'assurer la sécurité des phases d'exécution des conteneurs est la mise en œuvre d'une politique de surveillance. Cela peut être difficile car la destruction des conteneurs se conjugue avec la perte de leurs données. Heureusement, cette situation ne s'applique pas à tous les conteneurs. La stratégie ici est donc de repérer et de surveiller les traces que laissent ces paquets exécutables.
- **Adopter un décalage des processus vers la gauche** : c'est une règle qui recommande d'effectuer des tests de sécurité sur les applications déjà durant la phase de développement.
- **Développer et déployer une politique globale de protection des conteneurs** : il est crucial de mettre en place des actions pour assurer la sécurité de l'ensemble du cycle de vie des codes embarqués. Pour rendre la démarche efficace, il faut tenir compte du niveau de sensibilité et de risque des conteneurs.

#### Méthode La mise à jour d'un conteneur d'applications

Avec les conteneurs, les applications sont formées de plusieurs composantes indépendantes. Dès lors, mettre à jour l'application revient à installer la dernière image d'un ou de certains conteneurs. Ce processus simple peut être implémenté en exploitant les fonctionnalités intuitives des plateformes de conteneurisation ou en utilisant des lignes de commande. Quelle que soit l'approche de résolution choisie, le cheminement peut être décliné en 4 étapes.



**Étape 1 :**

Pour commencer, il faut rechercher la dernière image du conteneur. Un grand nombre d'images de conteneurs sont proposées dans les bibliothèques des plateformes de conteneurisation comme Docker ou Kubernetes. En l'absence de quelque chose qui convient à ces besoins, on peut choisir d'apporter des modifications à la dernière image en question.

**Étape 2 :**

Une fois la dernière image trouvée, il faut à présent passer à son téléchargement.

**Étape 3 :**

Il faut arrêter et nettoyer le conteneur avant d'intégrer la dernière image à l'architecture de l'application.

**Étape 4 :**

Enfin, vient la dernière étape du processus. Elle consiste à nettoyer le conteneur pour rendre opérationnelles les mises à jour apportées par la dernière image.

**Exemple** **Cas de mise à jour applicative avec le conteneur Bitwarden via Docker**

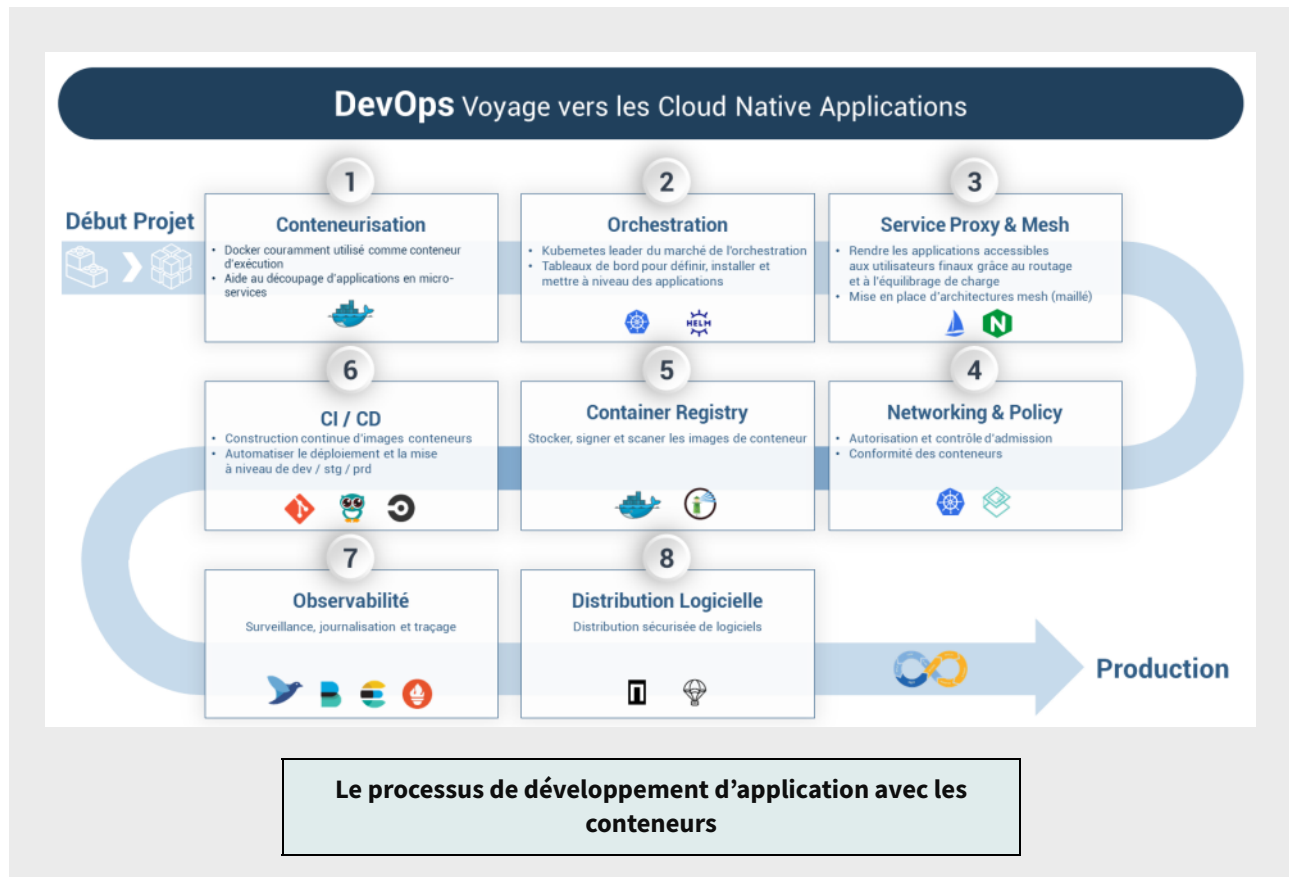
Pour notre exemple, nous avons choisi de mettre à jour le conteneur docker Bitwarden de notre NAS Synology. Avant l'entrée en matière, nous allons nous connecter à notre espace administrateur. Puis il nous faut vérifier que notre Docker est à jour. Pour cela, nous nous rendons dans l'espace « *centre de paquets* » de la plateforme. Si l'interface affiche un bouton « *mettre à jour* », alors nous allons cliquer dessus pour actualiser la version du NAS.

Après les précédents prérequis, nous ouvrirons la fenêtre « *registre* » de l'interface de Docker. Là, nous allons entrer le nom de l'application dans la barre de recherche. La plateforme fournira une liste d'images intégrées à son référentiel et dont les noms sont similaires au texte saisi. Nous ferons un double clic sur le nom de l'application à mettre à jour. Dans notre cas, ce sera bien évidemment « *Bitwarden* ».

Avec ce double clic, nous allons lancer le téléchargement de l'image du conteneur. Mais avant qu'il ne démarre, nous devons préciser la version qui nous intéresse. Pour cela, dans la liste déroulante de la fenêtre qui s'ouvre par-dessus l'interface de la plateforme, nous allons choisir l'expression « *latest* ». Elle signifie que nous souhaitons télécharger la dernière image du conteneur Bitwarden disponible dans la bibliothèque de Docker. Une fois la requête validée, la plateforme entame le téléchargement du paquet exécutable.

Après le téléchargement, nous allons arrêter le conteneur afin d'effectuer la mise à jour. Alors nous plaçons la souris sur l'image puis nous réalisons un clic droit. À l'issue de cette manœuvre, nous choisissons l'option « *action* » puis par la suite l'option « *Arrêt* ». Voilà, nous venons de suspendre le fonctionnement du conteneur Bitwarden. Maintenant, nous pouvons passer à la mise à jour effective.

Pour cette dernière étape, il suffit de choisir le conteneur puis de cliquer sur le menu contextuel du clavier pour accéder aux options « *action* » puis « *effacer* ». Après cette opération, l'ancienne version du conteneur laisse place à la dernière image qu'on vient juste de télécharger.



### Comment choisir son outil de conteneurisation applicative

Docker, AWS, LXC, Microsoft Azure, etc. Il y a tellement d'outils de conteneurisation des applications sur le marché du Dev ! Pour une simple mise à jour d'application conteneurisée, on peut se retrouver confus. Or, il suffit de maîtriser certains éléments clés pour réussir à faire un choix efficace et pertinent. À cet effet, il faut :

- **Maîtriser les préférences des développeurs de votre équipe** : les développeurs peuvent être plus à l'aise sur une plateforme que sur une autre. Ils peuvent être mieux aguerris à tester leurs travaux ou maintenir l'application avec certains outils spécifiques. Ne pas tenir compte de leurs antécédents avec ses solutions risque de ralentir le projet. Puis il faut prévoir un temps d'adaptation. De plus, une mauvaise relation à l'outil comporte un risque d'incohérence des opérations, ce qui peut se révéler très coûteux pour l'entreprise.
- **Analyser les fonctionnalités offertes par les plateformes** : en dépit d'une base fonctionnelle commune, les outils de conteneurisation admettent de nombreuses différences. En fonction du niveau d'exigence du projet, vous pouvez avoir besoin d'une fonctionnalité telle que celle offerte par JFrog Artifactory. Elle pourrait aider à améliorer la qualité du livrable en veillant, tout au long du processus, à valider le niveau de qualité de chaque image de conteneur intégrée à l'application.
- **Appliquer la règle de vérification des 4 C** : c'est un principe simple qui préconise de faire son choix en tenant compte du code, de la clientèle de la solution, de sa capacité cloud et du caractère complet de l'offre. En quelque sorte, la bonne plateforme est celle dont le code est fiable. C'est une plateforme largement utilisée, qui offre des possibilités sécurisées d'exécution sur le cloud. Enfin, c'est aussi cette plateforme qui dispose d'une large variété de produits complémentaires pouvant aider à la mise en œuvre du projet du début jusqu'à la fin.
- **Assurez-vous de la durabilité de la solution** : c'est toujours plus efficace d'adopter des solutions que vous pouvez utiliser d'une année sur l'autre. Mais, pour repérer de tels outils, il faut déjà étudier la courbe d'évolution de la solution. Assurez-vous qu'elle est en constante évolution et s'adapte aux progrès qui apparaissent dans le secteur du DevOps.

**Exercice : Quiz**

[solution n°2 p.16]

## Question 1

Docker est l'unique outil de conteneurisation sur le marché.

- ☐ Vrai
- ☐ Faux

## Question 2

Pour créer une application conteneurisée, on a besoin de construire le code à partir de zéro.

- ☐ Vrai
- ☐ Faux

## Question 3

Les images de conteneurs sont stockées dans des bases de données.

- ☐ Vrai
- ☐ Faux

## Question 4

La conteneurisation est un processus complexe.

- ☐ Vrai
- ☐ Faux

## Question 5

Plus un conteneur est volumineux, moins il est performant.

- ☐ Vrai
- ☐ Faux

**V. Essentiel**

En définitive, la mise à jour applicative par les conteneurs est un processus qui consiste à utiliser des conteneurs pour implémenter les nouvelles versions des applications. Au cœur du processus, il y a donc les conteneurs et ce qu'on appelle la conteneurisation.

Un conteneur, ici, est une composante numérique utilisée pour emballer un programme spécifique. La conteneurisation quant à elle est l'ensemble des actions de création, de déploiement et de gestion des applications conçues avec des conteneurs informatiques. C'est une technique révolutionnaire qui a permis d'une part aux entreprises de réduire leurs coûts de développement de solution informatique, et d'autre part qui a permis au développeur d'être plus efficace et plus productif.

Désormais, il suffit de rassembler les conteneurs de codes, de bibliothèques et de dépendances puis de les orchestrer pour avoir une application prête à l'usage. Le reste consistera à répliquer certains conteneurs puis à les modifier pour obtenir des fonctionnalités nouvelles.

Au-delà de la réduction de coût et de la productivité, la conteneurisation est une technologie qui garantit la portabilité des programmes. Les codes sont regroupés par blocs exécutables et opérationnels, d'un environnement ou d'une machine à l'autre.

Tout ceci justifie l'engouement des entreprises vis-à-vis de cette technologie. Mais, ce que beaucoup oublient, c'est qu'elle n'est en aucun cas une baguette magique. En effet, elle comporte aussi des limites, en particulier au niveau sécuritaire, ce qui nécessite d'adopter de bonnes pratiques lors de son implémentation.

Et l'un des cas récurrents où l'on fait appel à cette technologie est la mise à jour des applications basées sur les conteneurs. Si les principes de fonctionnement et de protection du processus sont maîtrisés, alors le reste est un véritable jeu d'enfant. En effet, il suffit de trouver la nouvelle version des conteneurs à actualiser, de les télécharger puis de supprimer les anciennes versions.

## VI. Auto-évaluation

### A. Exercice

Vous êtes membre de l'équipe de l'agence d'ingénierie informatique StarLab. Dans le cadre du déploiement de plusieurs progiciels bancaires, l'entreprise hésite entre utiliser la technologie de la virtualisation ou celle de la conteneurisation. Vous êtes chargé d'aider les décideurs à trouver la bonne solution.

#### Question 1

[solution n°3 p.17]

Quelle solution conseilleriez-vous ? Quelles sont les raisons qui peuvent justifier un tel choix ?

Vous êtes développeur dans l'agence WebMeter et spécialiste de Docker. L'agence vient de vous confier le déploiement d'une page web. Parmi les exigences du projet, il est spécifié de procéder à la conteneurisation de l'application web Mywebapp.

#### Question 2

[solution n°4 p.17]

Quelles sont les étapes et les opérations à réaliser ? Quelles sont les commandes que vous allez utiliser ?

### B. Test

#### Exercice 1 : Quiz

[solution n°5 p.18]

##### Question 1

Les conteneurs sont-ils des machines virtuelles ?

- ☐ Oui
- ☐ Non

##### Question 2

Quel est le défaut de l'utilisation des conteneurs ?

- ☐ Ils sont trop légers
- ☐ Ils ne sont pas transportables
- ☐ Ils sont éphémères
- ☐ Ils sont trop flexibles

##### Question 3

Les conteneurs sont moins sécurisés que les VM.

- ☐ Vrai
- ☐ Faux

##### Question 4

Un conteneur peut passer d'un environnement à l'autre.

- ☐ Vrai
- ☐ Faux

Question 5

Une application conteneurisée est un outil physique de gestion des conteneurs.

- ☐ Vrai
- ☐ Faux

## Solutions des exercices




**Exercice p. 6 Solution n°1****Question 1**

Un conteneur est un système d'exploitation utilisé par une application pour reproduire avec exactitude son fonctionnement.

☐ Vrai

☒ Faux

 Un conteneur est un progiciel qui contient des applications à déployer dans des environnements.

**Question 2**


Un conteneur est composé de :

☐ Codes

☐ Dépendances

☐ Librairies

☒ Les 3

 Un conteneur est un ensemble de fichiers, de variables, de codes et de bien d'autres éléments essentiels à l'exécution d'une application.

**Question 3**

Les conteneurs applicatifs ont été inventés par Docker.

☐ Vrai

☒ Faux

 Le concept de conteneur applicatif existait bien avant le projet Docker.


**Question 4**

Qu'est-ce qu'un hyperviseur ?

☐ Un composant de caméra

☒ Un programme qui permet de faire une virtualisation

☐ Un gestionnaire de base de données distant


 L'hyperviseur est un moniteur intégré dans les machines virtuelles. C'est lui qui alloue à la VM une partie des ressources mémoire et des processeurs de la machine hôte.

**Question 5**

Qu'est-ce que la virtualisation ?

☐ Une technique qui permet de créer des conteneurs applicatifs

☒ Une technique qui permet de produire une copie visuelle et fonctionnelle d'un système d'exploitation sur une machine hôte

-  La virtualisation est le processus mis en œuvre par les machines virtuelles. Elle permet, grâce à un hyperviseur, de cloner un environnement de travail et de le faire fonctionner sur une machine physique à côté d'autres environnements.


### Exercice p. 11 Solution n°2

#### Question 1

Docker est l'unique outil de conteneurisation sur le marché.

☐ Vrai

☒ Faux


-  Il existe beaucoup d'autres outils de création et de déploiement de conteneur d'application. Cependant, Docker demeure le premier à s'être lancé dans ce domaine.

#### Question 2

Pour créer une application conteneurisée, on a besoin de construire le code à partir de zéro.

☐ Vrai

☒ Faux


-  La raison même d'existence des conteneurs est de favoriser la réplication des codes exécutables sans risquer de faire face à des erreurs.

#### Question 3

Les images de conteneurs sont stockées dans des bases de données.

☐ Vrai

☒ Faux


-  Les images de conteneurs sont disponibles dans des référentiels proposés par des communautés de développeurs.

#### Question 4

La conteneurisation est un processus complexe.

☐ Vrai

☒ Faux


-  C'est plutôt un processus simplifié qui consiste à trouver les images de conteneur dont on a besoin et à les exécuter.

#### Question 5

Plus un conteneur est volumineux, moins il est performant.

☒ Vrai

☐ Faux

-  L'un des principes de la conteneurisation est de garder les exécutables des applications légères pour faciliter leur portabilité et optimiser l'utilisation des ressources.



**p. 12 Solution n°3**

Dans le cadre de ce projet, nous conseillons la virtualisation plutôt que la conteneurisation des applications.

Il est vrai que les conteneurs permettent d'embarquer plusieurs applications sur une même machine. Ils peuvent être exécutés dans tous les environnements et sur toutes sortes d'infrastructures matérielles (serveur local ou cloud). Ils permettent d'optimiser l'exploitation des ressources machine.

Mais au-delà de tous ces atouts supplémentaires vis-à-vis d'une machine virtuelle, les conteneurs constituent la technologie la moins sécurisée des 2. Tout d'abord, on connaît souvent le conteneur qu'on installe, mais pas toujours de quoi il dépend. Alors, comme il s'installe directement sur l'environnement de la machine, la présence de fichiers exécutables dangereux en son sein peut compromettre la machine voire le réseau de nœuds auquel elle appartient. Inversement, une attaque sur l'environnement déployé sur la machine peut mettre en danger les applications conteneurisées.

C'est là que la virtualisation devient une solution plus appropriée dans notre cas. Les machines virtuelles ont un fonctionnement indépendant du système d'exploitation déployé sur une machine. Elles utilisent un hyperviseur pour s'allouer les ressources machines dont elles ont besoin pour fonctionner. De plus, à leur installation, elles arrivent avec toutes leurs applications. Il serait alors plus simple et sûr de les utiliser dans le cadre de notre projet de déploiement d'application de *banking*.

**p. 12 Solution n°4**

Pour commencer, nous allons rédiger notre document Dockerfile dans le répertoire de notre application Mywebapp. C'est dans ce fichier que nous allons mettre nos instructions de conteneurisation ainsi que les détails sur l'image de conteneur et sur ses dépendances. Toujours dans ce fichier, nous mentionnerons également toutes les données de l'application web.

Maintenant, nous allons ouvrir notre terminal puis nous allons lancer les 2 commandes suivantes :

```
1 mkdir mywebapp
2 touch mywebapp/ Dockerfile
```

La première commande permet de créer un dossier pour isoler notre processus. La deuxième commande, quant à elle, permet de créer le fichier Dockerfile.

Nous allons placer dans le fichier les lignes de code suivantes :

```
1 # Utilisation d'une image Ubuntu (par défaut la dernière en date) pour construire notre image
  docker file
2
3 FROM ubuntu
4
5 # Mise à jour des repository distant du container, avant d'installer les paquets requis pour
  le projet
6
7 RUN apt update && apt upgrade -y
8
9 # Permet d'éviter d'avoir le bug concernant le choix de la timezone
10
11 RUN DEBIAN_FRONTEND="noninteractive" apt-get -y install tzdata
12
13 # Installation des paquets requis pour le projet à savoir git et le service web apache2
14
15 RUN apt-get install -y -q git apache2
16
17 # Le conteneur s'exécutera en se basant sur le service apache2
18
19 ENTRYPOINT /usr/sbin/apache2ctl -D FOREGROUND
20
21 # Renommage du fichier de base d'apache2 index.html vers index.html.old
```

```

22
23 RUN mv /var/www/html/index.html /var/www/html/index.html.old
24
25 # Récupération de mon repository Git avec le mini projet
26
27 RUN git clone https://github.com/archidote/get-ready-simple-countdown-html-css-js
28
29 # Copie des fichiers du mini projet web vers la racine de mon serveur web
30
31 RUN cd get-ready-simple-countdown-html-css-js && cp * /var/www/html/

```

Une fois cela fait, nous allons sauvegarder le fichier puis le fermer. Il ne nous reste plus qu'à construire l'image de notre application web. Pour cela, nous utiliserons le code suivant :

```
1 docker build -t mywebapp.
```

Cette commande ne fera pas qu'implémenter une image de notre site. Elle attribuera également à l'image créée, qui est ici « *mywebapp* », le nom du site.

Pour vérifier l'état des modifications que nous venons d'apporter, nous allons lancer la commande :

```
1 docker images
```

Cette commande doit fournir en sortie la liste des images, y compris la nôtre (déjà construite sur la plateforme *via* Docker).

Dès à présent, nous devons passer à l'instanciation du conteneur. C'est-à-dire que nous devons créer une instance de notre site à partir de notre conteneur et l'exposer. Pour cela, nous utiliserons une commande unique :

```
1 docker run -d -p 8080 : 80 'id'
```

Avec la directive « *-d* », nous venons de démarrer l'instance du conteneur en arrière-plan. Par contre, la directive « *-p* » permet à notre port local 8080 de communiquer avec le port 80 du conteneur.

Voilà, pour accéder à la page du site il suffit d'ouvrir le navigateur puis d'entrer dans la barre d'adresse <http://localhost> ou l'ip 127.0.0.1 accompagné du port 8080.

## Exercice p. 12 Solution n°5

### Question 1

Les conteneurs sont-ils des machines virtuelles ?


- ☐ Oui
- ☒ Non

**Q** Non, les machines virtuelles sont des systèmes d'exploitation virtuels qui tendent à imiter l'environnement matériel. Elles sont installées sur une machine et cohabitent avec l'environnement initial duquel elles sont indépendantes. Par contre, les conteneurs sont installés sur l'environnement initial d'une machine. Ils tendent à optimiser la gestion des ressources qu'ils partagent avec l'environnement initial.

### Question 2

Quel est le défaut de l'utilisation des conteneurs ?

- ☐ Ils sont trop légers
- ☐ Ils ne sont pas transportables
- ☒ Ils sont éphémères
- ☐ Ils sont trop flexibles

-  Le cycle de vie des conteneurs va de leur création à leur destruction. À la fin de leur vie, leur contenu est perdu, ce qui peut être une déconvenue lorsqu'on doit exécuter des tâches persistantes.


### Question 3

---

Les conteneurs sont moins sécurisés que les VM.

☒ Vrai

☐ Faux

-  L'indépendance des VM par rapport au système d'exploitation et au réseau installés sur une machine leur offrent une couche de protection supplémentaire. Les conteneurs quant à eux exploitent l'environnement installé sur la machine et sont assujettis aux mêmes risques que cette dernière.


### Question 4

---

Un conteneur peut passer d'un environnement à l'autre.

☒ Vrai

☐ Faux

-  Les conteneurs sont légers et partagent le même noyau de système d'exploitation. Cela facilite donc leur portabilité d'un écosystème à l'autre.


### Question 5

---

Une application conteneurisée est un outil physique de gestion des conteneurs.

☐ Vrai

☒ Faux

-  Ce sont des outils logiciels dont les codes sont répartis dans des conteneurs applicatifs pour faciliter leur transport et leur exécution dans différents environnements.