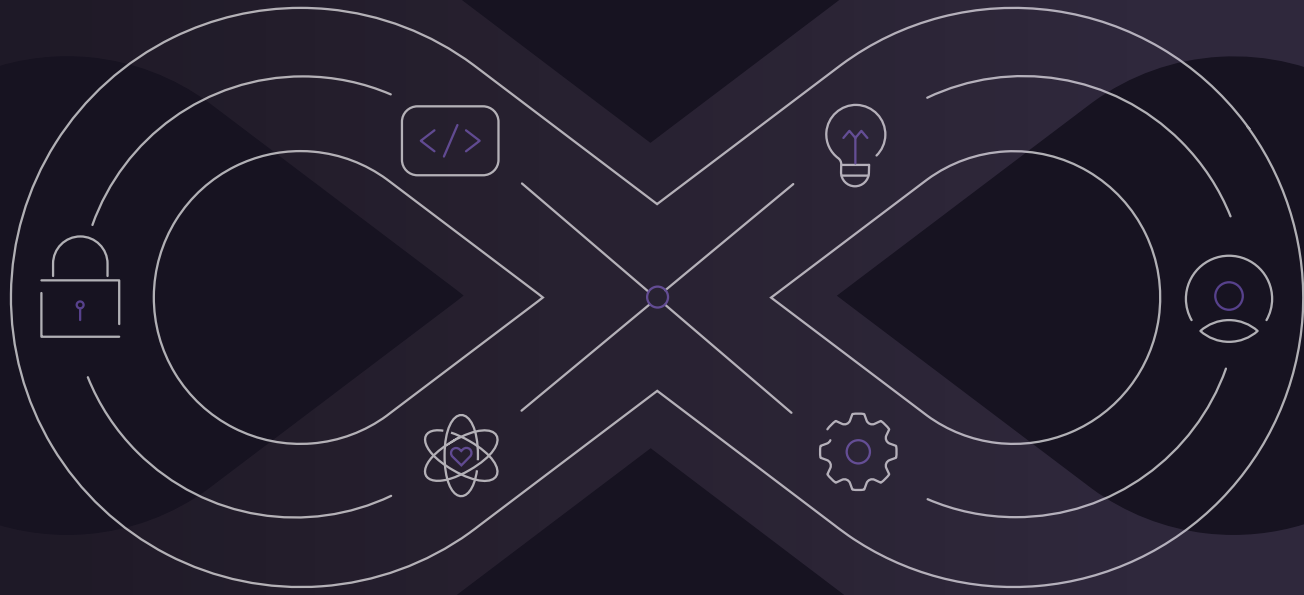
GitLab

Guide de démarrage DevOps



Sommaire

03 Introduction

04 Comment l'approche DevOps peut-elle aider votre entreprise ?

05 Technologies et processus : les fondamentaux

09 Autres technologies DevOps clés à maîtriser

12 Comment l'approche DevOps résout-elle des problèmes concrets ?

13 Ressources

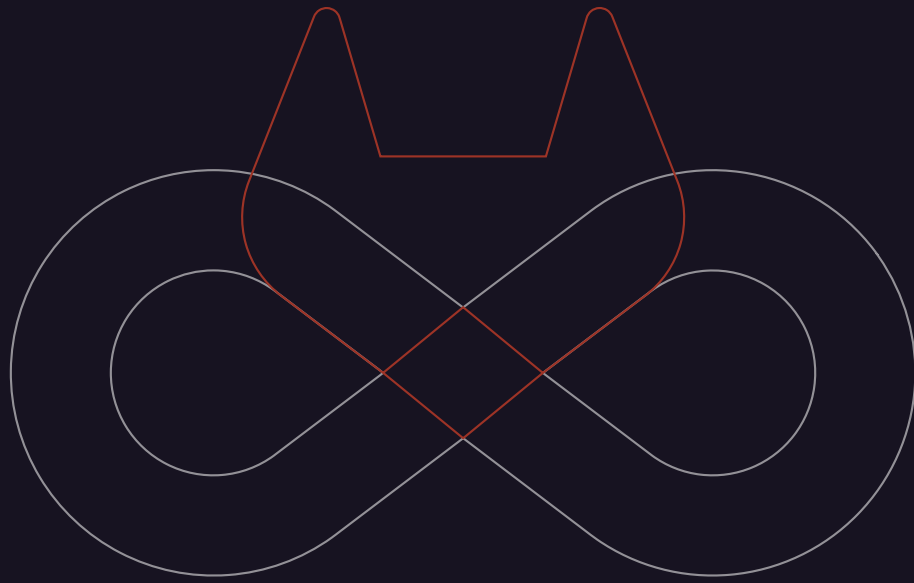
15 À propos de GitLab

Introduction

Au vu du nombre croissant d'entreprises qui se tournent aujourd'hui vers l'approche DevOps, il n'est pas surprenant que de nombreuses équipes DevOps soient encore novices dans ce domaine. Selon le **Rapport Global DevSecOps 2024** de GitLab, 40 % des répondants ont déclaré utiliser des méthodologies DevOps, contre 35 % en 2023.

Si votre entreprise a récemment adopté le modèle DevOps (ou s'apprête à le faire), vous ne connaissez peut-être pas encore tous les outils et les pratiques associés à cette approche. Ce guide contient tout ce que vous devez savoir pour être opérationnel. Nous vous expliquerons ce qu'est (et n'est pas) l'approche DevOps, les technologies et termes clés que vous devez connaître et pourquoi la collaboration est si importante. Nous vous présenterons ensuite un exemple pratique de processus DevOps, puis vous expliquerons en quoi cette approche peut booster votre carrière. Enfin, nous mettrons à votre disposition une liste exhaustive de ressources.

Plongeons maintenant dans le vif du sujet.



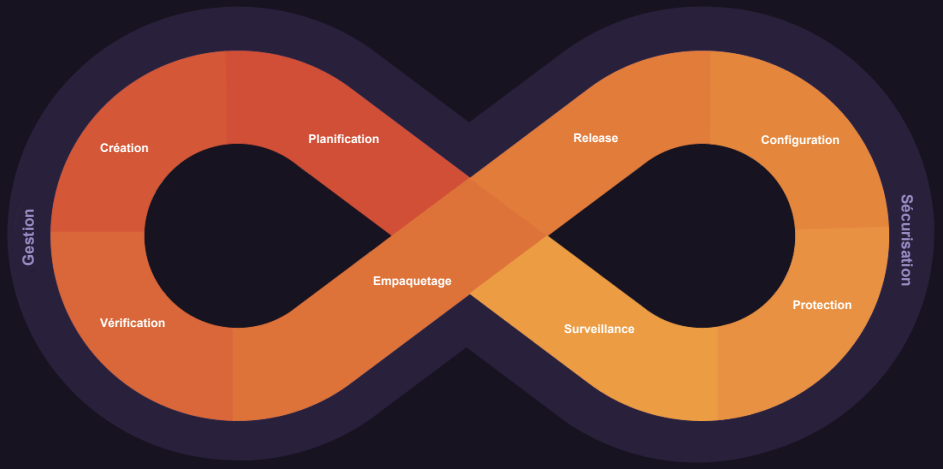
Comment l'approche DevOps peut-elle aider votre entreprise ?

La première chose que vous devez **savoir sur le processus DevOps** est qu'il implique de responsabiliser les équipes afin d'instaurer une culture de travail collaborative dans toute l'entreprise de manière à créer et fournir des logiciels sécurisés plus rapidement et plus efficacement. Pendant des années, le développement logiciel était tout sauf rationalisé et efficace. Les processus étaient cloisonnés et créaient des goulots d'étranglement ainsi que des retards coûteux, tandis que la sécurité intervenait bien trop tardivement. L'approche DevOps est née de la profonde frustration ressentie à l'égard des anciennes méthodes de travail et a apporté avec elle la promesse de processus plus simples et plus rapides.

L'approche DevOps se révèle particulièrement efficace sur une plateforme complète et unifiée. En effet, les équipes de développement peuvent abandonner ou éviter d'utiliser une multitude d'outils souvent complexes en faveur d'un écosystème de développement logiciel unique et complet. Les membres de l'équipe n'ont ainsi plus à passer d'un outil à l'autre, ce qui permet d'économiser du temps et de l'argent. Cet écosystème peut servir à concevoir, compiler et, à terme, livrer des logiciels de meilleure qualité, plus sécurisés et plus conformes, plus efficacement, plus rapidement et en continu. Il aide les équipes DevOps à être **plus agiles**, mais il offre également plus d'agilité à l'ensemble de l'entreprise, permettant ainsi de répondre plus rapidement aux besoins des clients, de rester conforme, de garder une longueur d'avance sur les concurrents et de tirer parti de l'évolution du climat des affaires. L'approche DevOps stimule donc l'agilité tant dans le cadre du développement et du déploiement de logiciels que dans celui des opérations des entreprises.

En recourant à l'automatisation, en intégrant la sécurité plus en amont et en rendant les processus reproductibles et mesurables, une plateforme DevOps permet d'améliorer la qualité des logiciels. Elle permet également de réduire les délais entre la conception de nouvelles fonctionnalités et leur déploiement dans l'environnement de production.

Il est essentiel de ne pas négliger **la culture** et l'état d'esprit qui accompagnent la pratique DevOps, car il ne s'agit pas d'un processus de développement habituel. La culture DevOps repose sur **la collaboration et la responsabilité conjointe**, ainsi que sur un cycle constant d'itération, de mesure, d'évaluation et de réévaluation rapides. Encore une fois, tout est question d'agilité et de capacité à apprendre et à déployer rapidement pour apporter des améliorations continues et itératives et procéder au déploiement de fonctionnalités.



Technologies et processus : les fondamentaux

Les étapes du processus DevOps

Votre apprentissage de l'approche DevOps reste incomplet tant que vous ne maîtrisez pas les étapes du cycle de développement, à savoir le processus allant de la planification au lancement de nouvelles fonctionnalités, en passant par l'analyse et la collecte des retours.

Lorsque vous travaillez sur une plateforme DevOps, vous devez impérativement connaître ces étapes, car chacune d'entre elles fait partie intégrante du processus. Dans l'ensemble, il comprend trois grandes étapes globales qui s'exécutent dans un ordre logique : la compilation, le test et le déploiement. Il s'agit là du cours naturel d'un workflow standard. Vous compilez le code, puis vous le testez et, si tout fonctionne, vous le déployez.

Cependant, il faut y regarder de plus près pour déceler les couches plus complexes de ces étapes. Chacune est déterminante pour générer de la valeur logicielle et commerciale. Une fois que vous comprendrez et utiliserez ce processus, vous gagnerez en efficacité, en fiabilité, en rapidité et en agilité. Voici un aperçu plus détaillé des neuf étapes clés :

- **Planification** : cette étape englobe tout ce qui se passe avant que la première ligne de code ne soit écrite. Il s'agit de créer une roadmap produit qui guidera le développement à venir et aidera l'équipe à organiser les ressources et les priorités, à coordonner les efforts et à suivre les projets.
- **Création** : il s'agit de la première étape du pipeline CI/CD. C'est à ce moment-là que le code est conçu et développé. Les développeurs utilisent une pratique basée sur le contrôle de version pour coordonner toutes les modifications qu'ils apportent à un même code base. C'est l'une des clés pour améliorer la vélocité.
- **Vérification** : ce processus est axé sur la confirmation de la qualité du code. Il s'appuie sur les tests de sécurité, l'analyse de la qualité du code, l'exécution parallèle et l'automatisation pour permettre aux développeurs et aux testeurs d'obtenir rapidement des commentaires et des informations instantanées sur chaque validation. Il s'avère **plus rentable et plus efficace** de permettre aux développeurs de trouver et de corriger les failles de sécurité pendant le développement.
- **Empaquetage** : l'étape qui est lancée une fois que le code a été créé et testé. L'empaquetage des applications et des dépendances, la gestion des conteneurs et la création d'artefacts contribuent à créer une chaîne d'approvisionnement logicielle fiable.
- **L'étape de release** ou de déploiement : elle consiste à effectuer un push des mises à jour du code dans l'environnement de production. Avec l'approche DevOps, les versions de release peuvent être déployées dès que les itérations sont créées, testées et prêtes, et non selon une date de release groupée statique et pré-planifiée.
- **Configuration** : elle comprend la mise en place, la gestion et la maintenance des environnements d'application. La gestion automatisée de la configuration est conçue pour gérer ces environnements complexes sur les serveurs, les réseaux et les systèmes de stockage.
- **Surveillance** : cette partie proactive et automatisée du processus se concentre sur le suivi des logiciels, de l'infrastructure et du réseau afin de garder un œil sur leur statut et d'émettre des alertes en cas de problèmes. Elle augmente la sécurité, la fiabilité et l'agilité.
- **Protection** : cette étape vise à sécuriser les applications et leur environnement d'exécution contre les intrusions et les nouvelles vulnérabilités.
- **Gestion** : elle implique d'avoir une visibilité et un contrôle sur l'ensemble du cycle de développement logiciel par le biais de la gestion des autorisations, de la standardisation des processus de compilation et de déploiement DevOps, et de l'automatisation des garde-fous afin de garantir que les stratégies de sécurité et de conformité sont respectées.

Qu'en est-il de la sécurité ? C'est une bonne question. Et c'est la beauté de l'approche DevOps : la sécurité n'est pas une considération secondaire. Elle intervient à CHAQUE étape du processus, de la documentation des exigences aux tests automatisés en passant par la validation de ces exigences. Cela garantit que le nouveau code et les nouvelles fonctionnalités fonctionnent exactement comme prévu et qu'aucun bogue, qu'aucune menace pour la sécurité ou qu'aucun problème de conformité ne puisse apparaître.

Ces étapes font toutes partie d'un cycle continu. Toutes les informations créées au cours du processus sont instantanément disponibles via la plateforme pour toutes les parties prenantes. Elles constituent ainsi une source unique de vérité qui favorise la visibilité et la collaboration. L'autre avantage intéressant d'une plateforme unifiée est la capacité de gérer et de contrôler l'ensemble du cycle de développement logiciel à partir d'une seule interface.

Comment optimiser les étapes du cycle DevOps ?

Gestion du code source (SCM)

C'est ainsi qu'un dépôt de code est partagé entre de nombreux développeurs sans que les modifications effectuées par l'un d'entre eux n'aillent à l'encontre de celles d'un autre. Le code est divisé et géré en projets et groupes de projets. Un développeur examine le code existant ou ajoute de nouvelles lignes, tandis que l'outil SCM détecte les modifications conflictuelles apportées et indique qu'elles doivent être résolues. Ce processus permet à plusieurs développeurs de travailler sur un projet en même temps, ce qui est essentiel pour augmenter la vélocité des mises à jour logicielles. L'approche DevOps repose sur les dépôts Git, **une distinction importante par rapport aux systèmes de contrôle de version traditionnels**, en raison des puissantes capacités que leur architecture moderne permet.

Intégration continue (CI)

L'étape de l'intégration continue rend l'itération possible en validant les modifications à temps et fréquemment (plusieurs fois par jour) dans un dépôt de code source partagé, en testant automatiquement chaque modification et en lançant une compilation.

L'intégration continue est une question d'efficacité. En automatisant le travail manuel et en testant le code plus fréquemment, les équipes peuvent itérer plus rapidement et déployer de nouvelles fonctionnalités avec moins de bogues plus souvent. Parmi **ses autres avantages, la CI** permet également d'identifier et de résoudre les problèmes plus facilement, de réduire le nombre de changements de contexte pour l'équipe et d'améliorer la satisfaction des utilisateurs et des clients.

Si vous souhaitez tirer le meilleur parti de l'intégration continue, vous devez vous assurer que votre configuration comprend les éléments clés suivants :

- Un dépôt de code source avec tous les fichiers et scripts nécessaires pour créer des compilations.
- Des compilations automatisées avec des scripts qui incluent tout le nécessaire pour compiler à partir d'une seule commande.
- Des compilations dotées de capacités d'auto-test qui automatisent vos stratégies (par exemple, mise en échec si un test échoue).
- Des itérations et des validations fréquentes afin de réduire le nombre de conflits.
- Des environnements de test stables qui reflètent fidèlement l'environnement de production.
- Une plus grande visibilité pour que chaque développeur puisse accéder aux derniers exécutables et visualiser les modifications apportées au dépôt.

Livraison continue (CD)

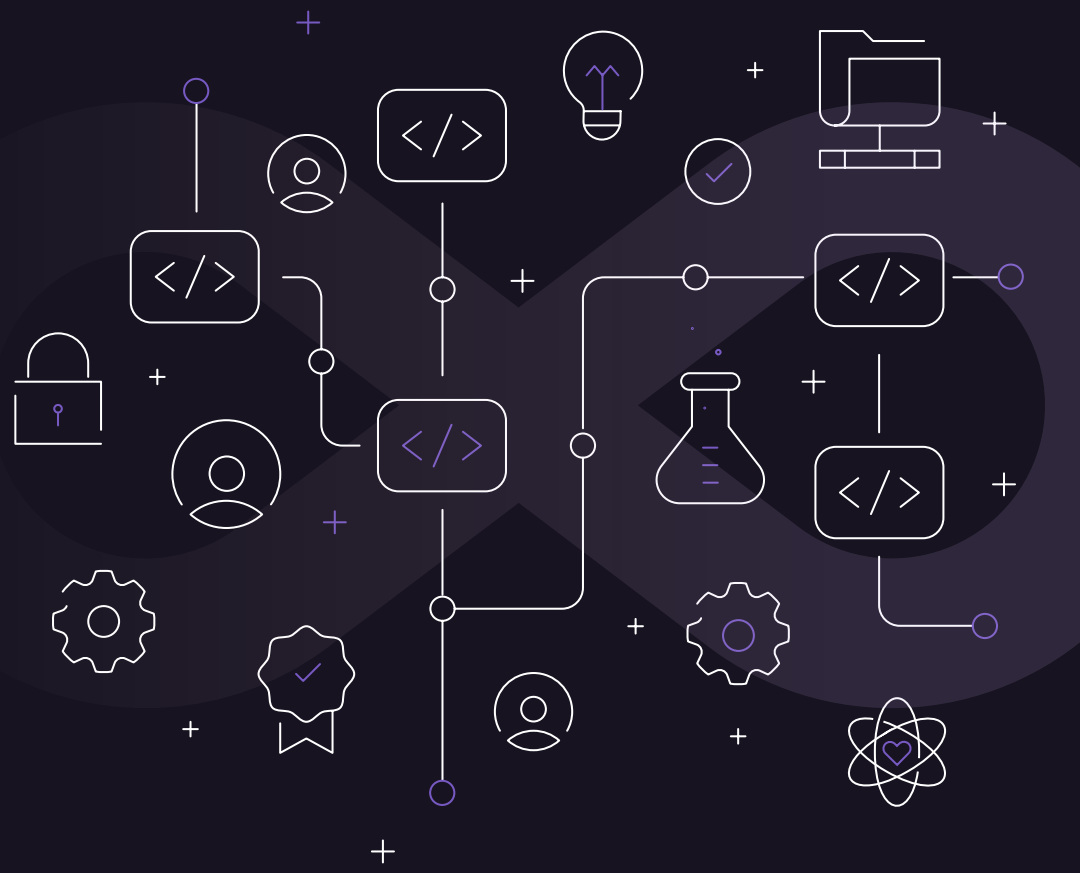
La livraison continue est un processus de développement logiciel qui fonctionne conjointement avec l'intégration continue pour automatiser le processus de sortie d'une application. Une fois que le code a été testé et compilé dans le cadre du processus d'intégration continue, les étapes finales sont prises en charge par la livraison continue pour garantir que les paquets contiennent tout le nécessaire pour un déploiement dans n'importe quel environnement, à tout moment. La livraison continue peut englober une multitude d'opérations, du provisionnement de l'environnement de l'infrastructure au déploiement de l'application testée en environnement de test/préproduction ou de production.

Grâce à la livraison continue, le logiciel est compilé de manière à pouvoir être déployé à tout moment. Vous pouvez ensuite déclencher les déploiements manuellement ou automatiser le processus.

Lorsque la livraison continue est bien configurée, les **processus de livraison de logiciels deviennent prévisibles**, car ils présentent peu de risque, sont cohérents et reproductibles. Vous pouvez alors planifier en toute confiance les processus et les calendriers de sortie des nouvelles versions, automatiser l'infrastructure et les déploiements, et gérer plus efficacement vos ressources cloud.

Tests automatisés

Les tests automatisés sont essentiels pour adopter pleinement l'approche DevOps et l'intégration continue, ainsi que pour des releases plus fréquentes de code de meilleure qualité. Lorsque les tests sont intégrés au pipeline CI, chaque modification de code validée déclenche la création d'une compilation. La compilation exécute ensuite des tests pour s'assurer que les modifications réussissent tous les tests et respectent les stratégies et les normes de conformité de code définies pour votre application. De cette façon, les bogues sont identifiés plus tôt et avec plus de contexte pour simplifier la correction, vos équipes peuvent effectuer des déploiements plus fréquemment et en toute confiance, et vous réduisez au strict minimum les tests manuels et les retouches en fin de processus.



Sécurité en amont

Une opération clé pour garantir la réussite d'une approche DevOps consiste à intégrer la sécurité dans l'automatisation de bout en bout. Cette approche est plus connue sous le nom de DevSecOps. Si vous intégrez les tests et le processus de vérification de la sécurité au début du cycle du développement logiciel, il est plus facile de traiter correctement les problèmes liés à la sécurité. D'autre part, si les tests de sécurité passent au second plan ou sont uniquement effectués lorsque le code est prêt pour la production, il peut être difficile de revenir en arrière et de résoudre les problèmes, et il est même parfois trop tard pour les corriger rapidement et efficacement. Cela peut ensuite entraîner des retards dans les déploiements, des vulnérabilités dans l'environnement de production, une dette technique plus importante et des cloisonnements inefficaces entre l'équipe de sécurité et les équipes DevOps.

Pour intégrer la sécurité en amont, vous devez ajouter des tests de sécurité à vos pipelines CI. De cette façon, le code est continuellement testé dans une optique de sécurité globale, et non pas uniquement pour les autres validations du dépôt partagé. Voici quelques types de tests de sécurité que vous devriez inclure au début de votre cycle de développement :

- Test statique de sécurité des applications (SAST)
- Test dynamique de sécurité des applications (DAST)
- Analyse des images de conteneurs et de clusters
- Analyse des dépendances
- Détection des secrets
- Analyse de l'Infrastructure as Code (IaC)
- Tests API

Documentation

Bien que parfois négligée, la documentation est inestimable pour la mise en œuvre des pratiques DevOps. La création et la maintenance d'une documentation interne pour les services et les applications sur lesquels votre équipe travaille et pour votre processus DevOps peuvent contribuer à améliorer les performances de votre équipe et des logiciels que vous créez. Le **rapport Accelerate State of DevOps 2024** a révélé que les équipes qui améliorent la qualité de la documentation produit pour leurs utilisateurs constatent une augmentation des performances du produit.

Retours

Les retours constituent une pièce essentielle du puzzle, car les entreprises doivent toujours rechercher des moyens d'améliorer l'expérience utilisateur ainsi que le processus DevOps global. Dans le cadre du développement traditionnel de logiciels, il peut s'avérer difficile d'obtenir des retours d'information. Avec l'approche DevOps, une collaboration renforcée et des itérations rapides permettent aux équipes d'avoir un accès continu à des données exploitables, leur offrant ainsi la possibilité d'intégrer les retours, d'ajuster leurs efforts et d'apporter des améliorations de manière efficace et rapide. L'automatisation du processus est donc une étape primordiale pour s'assurer que les informations sont collectées et distribuées aux bonnes personnes dans l'équipe, qui pourront ensuite procéder rapidement aux modifications du code.

Autres technologies DevOps clés à maîtriser

À mesure que vous vous initiez à l'approche DevOps, il est essentiel de vous informer sur plusieurs domaines et de veiller à rester à jour. Voici quelques-uns des domaines clés que vous devez comprendre et suivre de près.

Le cloud

Si vous travaillez sur une plateforme DevOps, il est utile de bien comprendre le fonctionnement **du cloud**. La plupart des logiciels modernes s'appuient sur une infrastructure cloud et cloud-native, y compris les conteneurs et les outils d'orchestration, qui aident à automatiser le processus de développement et de livraison des logiciels. Les applications ainsi développées permettent aux professionnels DevOps d'effectuer des déploiements n'importe où et de tirer le meilleur parti des **plateformes multi-cloud**, voire des clouds privés au sein de leur centre de données.

Une approche cloud-native est plus évolutive, car elle éloigne le code du matériel utilisé. Pour soutenir leur entreprise et faire progresser leur carrière, les professionnels DevOps doivent maîtriser les fournisseurs, les services et les plateformes cloud. Et c'est un point central pour de nombreux professionnels DevOps. Selon le **Rapport Global DevSecOps 2024 de GitLab**, 55 % des répondants ont déclaré exécuter au moins la moitié de leurs applications dans le cloud, contre 32 % des répondants en 2023.

Culture de collaboration

La collaboration est essentielle, c'est même l'un des principes fondamentaux de l'approche DevOps. En invitant les membres de l'équipe, qu'ils soient expérimentés ou non, à discuter, à apporter leur contribution et à offrir leur aide, il est possible de créer une culture d'apprentissage et de confiance basée sur l'expertise des autres. Pour faire partie d'une véritable culture DevOps, vous devez également être capable d'écouter, de garder votre calme, d'instaurer une relation de confiance entre les membres de l'équipe et de prendre la responsabilité d'une situation ou d'un problème rencontré.

Cependant, la collaboration ne concerne pas seulement les équipes DevOps qui travaillent ensemble. Elle implique également d'autres parties prenantes de l'entreprise comme les équipes de sécurité, du marketing, de la finance, du service clientèle et même la direction. La collaboration entre les équipes DevOps et de sécurité, par exemple, est un moyen d'intégrer la sécurité dans l'ensemble du processus de développement. Ne commettez pas l'erreur de considérer cet aspect comme une compétence non technique qui est moins importante que les compétences techniques. **Développez des compétences clés** comme la communication pour savoir parler des besoins de l'entreprise et collaborer de manière à résoudre des problèmes.

Langages de programmation essentiels

Les ingénieurs DevOps doivent savoir coder, mais ils doivent surtout tenir compte des processus, des outils et des méthodologies qui sont utilisés tout au long des étapes du cycle DevOps décrites ci-dessus. Certains langages se prêtent mieux à ce processus que d'autres, mais il existe de nombreux langages de programmation et il peut être très difficile de savoir par où commencer. Vous devez d'abord déterminer ce dont vos équipes DevOps ont besoin. Sur quels projets travaillez-vous ? De quels langages avez-vous actuellement besoin ? Quels langages seront nécessaires pour vos projets futurs ?

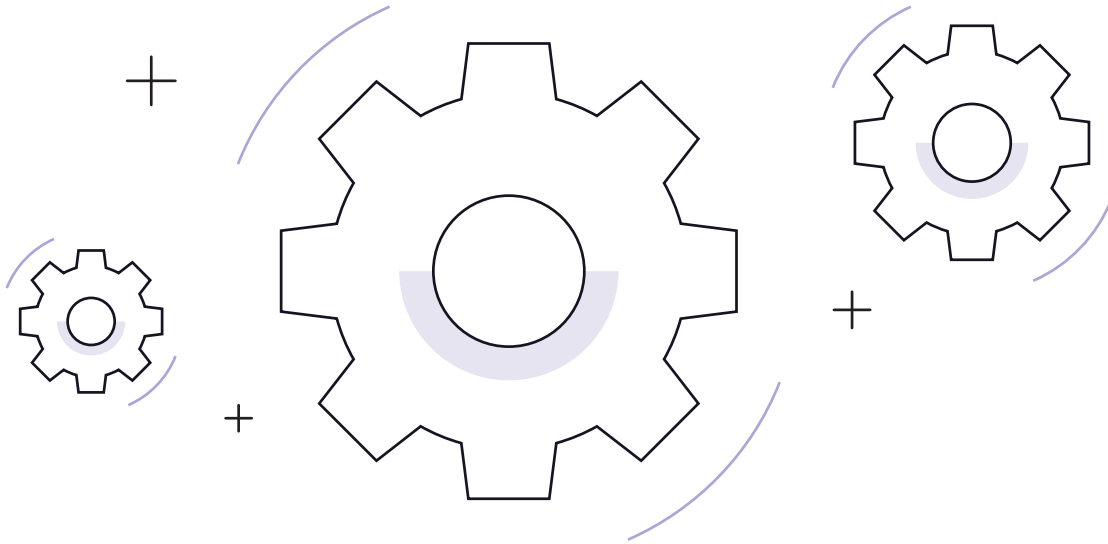
Certains des langages de programmation les plus populaires sont Python, Golang, Ruby, JavaScript, Perl, Java, Bash et PHP. Selon le **Stack Overflow Developer Survey 2024**, JavaScript est le langage de programmation le plus utilisé globalement, tandis que Python est le langage de prédilection pour les personnes apprenant à coder.

Automatisation

Les équipes DevOps ont de plus en plus tendance à automatiser les processus tout au long du cycle de développement et de déploiement. Par conséquent, il va sans dire que vous devez savoir comment fonctionne l'automatisation et comment l'utiliser. L'automatisation permet de réduire le temps et l'argent consacrés aux tâches répétitives et d'éliminer les erreurs humaines en rationalisant l'ensemble du processus DevOps. Grâce à l'automatisation, chaque tâche est effectuée de manière identique et avec cohérence, fiabilité et précision. Cela accélère le processus et augmente le nombre de livraisons (et donc de déploiements). L'automatisation n'exclut pas les humains, elle réduit seulement la nécessité de faire appel à eux pour la gestion des tâches récurrentes, comme la surveillance de la disponibilité, des performances ou des problèmes de sécurité, la configuration homogène des environnements logiciels, le test des nouvelles versions d'applications selon des normes de qualité prédéfinies, l'intégration de code, l'accélération des déploiements, la prise en charge des logiciels de test CI/CD pendant le processus de développement et la gestion des journaux et de la documentation. Selon le Rapport Global DevSecOps 2024 de GitLab, 67 % des répondants ont déclaré que leur cycle de développement logiciel est « presque entièrement » ou « entièrement » automatisé. 29 % ont également déclaré qu'il est « partiellement » automatisé. Cela signifie que si vous travaillez au sein d'une équipe DevOps, vous devriez normalement déjà posséder quelques connaissances en automatisation.

Surveillance

À mesure que la pile d'applications de votre entreprise et le nombre d'équipes DevOps qui y travaillent augmentent, le nombre d'éléments mobiles ne fera que se multiplier, et les suivre à la trace peut être une tâche compliquée. Par conséquent, une surveillance continue est une condition sine qua non pour pouvoir garder un œil sur la situation de votre écosystème de bout en bout et en temps réel. Grâce à l'approche



DevOps, les applications les plus complexes peuvent être mises à jour et déployées quotidiennement, voire plusieurs fois par jour. Une surveillance détaillée et automatisée est un moyen proactif de réduire les bogues, d'améliorer la rapidité et l'efficacité des déploiements, d'identifier les menaces de sécurité et les problèmes de conformité, d'éliminer les changements cassants et de maintenir la documentation. Elle doit être appliquée de la planification au développement, y compris lors de l'intégration, des tests, du déploiement et même des opérations.

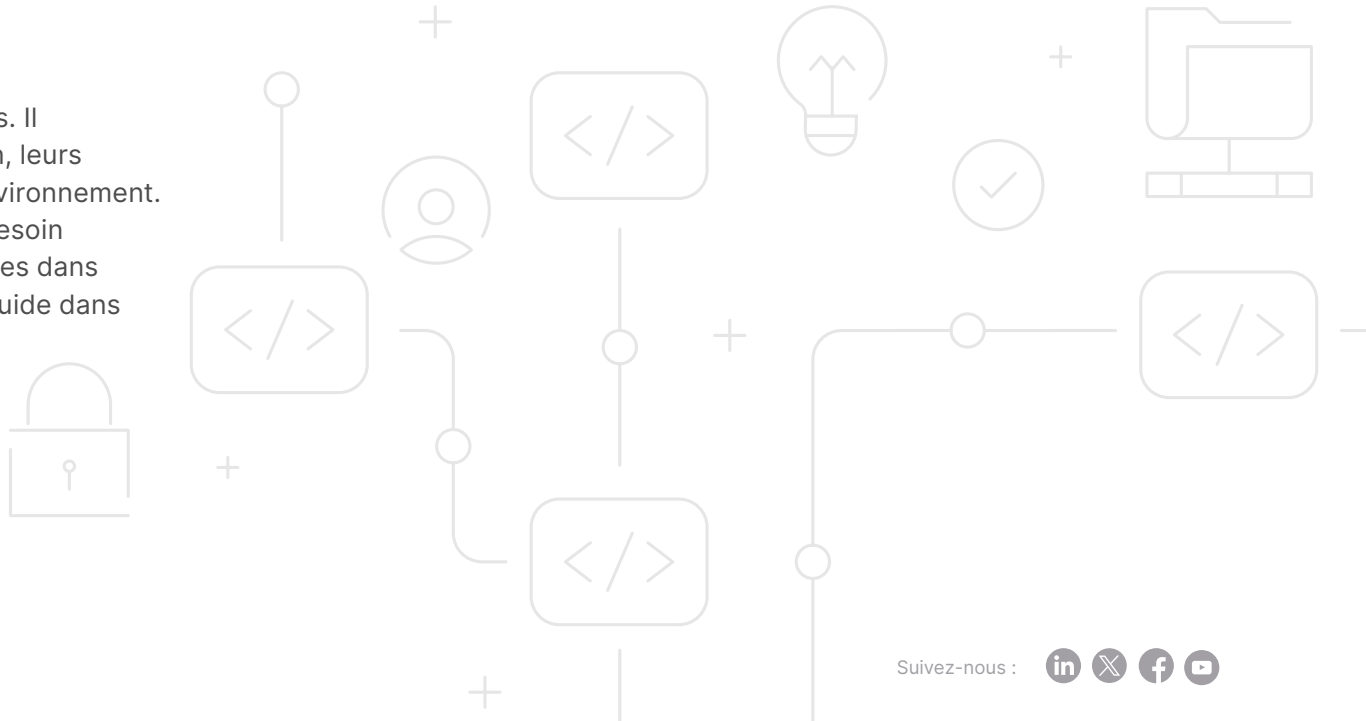
En d'autres termes, la surveillance est un processus indispensable non seulement pour les développeurs, mais aussi pour les chefs de projets et les équipes de sécurité. Elle ne se résume pas à suivre les processus, elle sert également à générer des alertes en cas de problèmes de performance et de menaces dans le pipeline.

Les équipes DevOps qui souhaitent exceller dans leur domaine devront de plus en plus s'appuyer sur la surveillance. Il est donc essentiel de maîtriser ce sujet.

Conteneurs

Les équipes DevOps ont largement adopté les conteneurs. Il s'agit de paquets de code logiciel, avec leur configuration, leurs bibliothèques système, leur runtime et le reste de leur environnement. Les conteneurs contiennent tout ce dont l'application a besoin pour s'exécuter. Ils garantissent que les applications créées dans un environnement s'exécutent de manière cohérente et fluide dans d'autres environnements. Ils permettent ainsi d'exécuter un logiciel de manière fiable lorsqu'il est déplacé d'un environnement à un autre. Ces unités modulaires (ou éléments de base) sont conçues pour permettre aux équipes DevOps de compiler, de tester, de déployer, et de maintenir efficacement des applications, avec moins de ressources, extrêmement rapidement et en toute sécurité.

Puisque les conteneurs peuvent être facilement partagés entre plusieurs équipes, non seulement ils accélèrent le développement et le déploiement, mais ils favorisent également une culture de collaboration, qui est le fondement même de l'approche DevOps. Si vous savez comment fonctionnent les conteneurs, vous devez également connaître Docker, une technologie populaire de conteneurisation, et Kubernetes, un système d'orchestration de conteneurs open source qui régit comment et où les conteneurs s'exécutent.



Comment l'approche DevOps résout-elle des problèmes concrets ?

Vous vous demandez quel impact réel une approche DevOps peut avoir sur votre entreprise ? Découvrez ce qu'une plateforme DevOps a apporté à **HackerOne**. En tant que plateforme de sécurité la plus fiable au monde, HackerOne permet aux entreprises d'accéder à la plus grande communauté de pirates informatiques de la planète. Avec une présence dans plus de 70 sites dans le monde, la société a été confrontée à plusieurs défis en matière de collaboration interfonctionnelle. Par exemple, lorsque les développeurs de différents continents ont dû reprendre là où d'autres s'étaient arrêtés sur un projet de code, de longs délais de pipeline ont interrompu les transferts. Lorsque son équipe d'ingénieurs a triplé, HackerOne a dû accélérer le développement et le déploiement, réduire la complexité de la chaîne d'outils et permettre aux équipes de gérer efficacement plusieurs projets. **La solution : une plateforme DevOps**. Avec une plateforme DevOps unique et unifiée, les équipes de HackerOne ont pu détecter les problèmes de code en amont dans le pipeline, travailler de manière itérative pour résoudre les failles de sécurité et simplifier les audits. Elles ont également augmenté la fréquence des déploiements d'une ou deux fois par jour à jusqu'à cinq fois par jour, et économisé quatre heures de temps de développement par développeur et par semaine.

Ressources

Voici quelques-unes des ressources à votre disposition (en anglais) :

Podcasts pour découvrir les pratiques DevOps

- **The Humans of DevOps Podcast Series** aborde des sujets tels que l'amélioration des compétences, l'art du DevOps et les femmes qui travaillent dans ce domaine.
- **Arrested DevOps** regroupe des entretiens avec des technophiles qualifiés sur l'état actuel des pratiques DevOps.
- **Real World DevOps** propose des entretiens avec des spécialistes DevOps, des auteurs de livres dédiés à ce sujet et des développeurs de solutions technologiques.
- **The Cloudcast** porte bien son nom, il se consacre à tout ce qui concerne le cloud.
- **Greater Than Code** se concentre à la fois sur les problématiques humaines et technologiques dans l'approche DevOps, mais aborde également le domaine des technologies en général.
- **Code Newbie Podcast** est destiné aux personnes qui débutent dans le développement logiciel.
- **DevOps Paradox** fait appel à des sommités du secteur pour vous expliquer ce qu'est l'approche DevOps.

Livres et e-books utiles

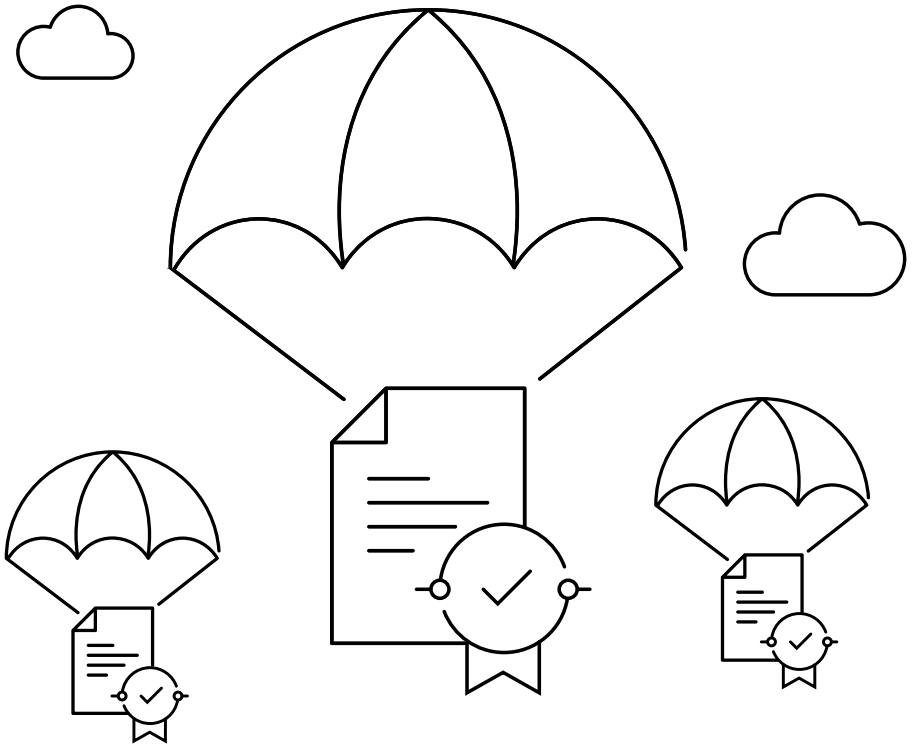
- **Sept astuces pour exploiter au maximum votre plateforme DevOps** est un e-book de GitLab qui a pour but de permettre aux équipes de tirer le meilleur parti d'une plateforme DevOps.
- **Continuous Delivery** a été qualifié d'ouvrage « incontournable » pour quiconque souhaite relier l'ensemble du processus de développement et de livraison.
- **Practical DevOps** explique comment fonctionne l'approche DevOps, puis aborde le stockage du code, les tests et le déploiement.
- **The DevOps Handbook** est considéré comme un ouvrage de référence pour toutes les personnes qui travaillent dans ce domaine. Il évoque non seulement les avantages de l'approche DevOps, mais explique également comment celle-ci peut offrir aux entreprises un avantage concurrentiel.
- **GitLab Quick Start Guide** est un e-book et un excellent guide pour passer à la plateforme GitLab.
- **Big Little Book on Git** est un e-book qui s'adresse aussi bien aux professionnels DevOps expérimentés qu'aux débutants.

Certifications

- **DevOps Institute propose des certifications** dans des domaines tels que le développement, l'ingénierie DevOps, les tests DevOps et l'ingénierie de la sécurité.

- **GitLab propose ses propres certifications** dans des domaines tels que la CI/CD, la gestion de projet et la sécurité DevOps.

- Allez toujours à la source. Par exemple, si vous souhaitez apprendre à utiliser Google Cloud, recherchez les **certifications possibles sur le site de l'entreprise**.



Ateliers et formations

- La plateforme de formation en ligne **A Cloud Guru** propose des certifications cloud moyennant des frais mensuels. Elle est structurée pour fournir aux utilisateurs du contenu vidéo, des exercices pratiques, des outils d'apprentissage, des questionnaires et des examens.

- **LinkedIn Learning's DevOps Foundations** offre une base de connaissances solide pour les personnes qui abordent l'approche DevOps pour la première fois, ou presque. Disponibles gratuitement pour toute personne disposant d'un abonnement LinkedIn, les vidéos offrent un aperçu de l'industrie ainsi que des principes et technologies clés comme l'automatisation, la collaboration, la surveillance et la culture.

- **The DevOps Implementation Boot Camp**, organisé par la société de conseil Cprime, est un cours de trois jours, disponible à partir de 1 695 $. La formation est proposée en présentiel ou en direct en ligne. Une formation en équipe privée est également disponible.

- **DevOps Culture and Mindset** est proposé par l'Université de Californie à Davis via Coursera, une plateforme de cours en ligne. Le cours d'environ 15 heures est entièrement en ligne et porte sur les principes fondamentaux de l'approche DevOps. Il est gratuit pour les personnes qui disposent d'un abonnement à Coursera.

- **Continuous Delivery & DevOps** est un cours en ligne de 8 heures pour débutants proposé par l'Université de Virginie via Coursera. Il se concentre sur la livraison continue, les tests et l'Infrastructure as Code. Il est gratuit pour les personnes qui disposent d'un abonnement à Coursera.

À propos de GitLab

GitLab est la plateforme DevSecOps alimentée par l'IA la plus complète pour l'innovation logicielle. Elle offre une interface centralisée avec un magasin de données unifié, un modèle d'autorisation unique, une chaîne de valeur cohérente, et des rapports centralisés, permettant de sécuriser, déployer et collaborer autour de votre code en un seul endroit. Il s'agit de la seule véritable plateforme DevSecOps complète et compatible avec tous les clouds qui regroupe et centralise toutes les fonctionnalités DevSecOps.

Avec GitLab, les entreprises peuvent créer, livrer et gérer du code rapidement et en continu et ainsi donner vie à leurs ambitions. GitLab permet aux clients et aux utilisateurs d'innover plus rapidement, de s'adapter plus facilement, de servir et de fidéliser les clients plus efficacement. GitLab est une plateforme open source qui s'appuie sur sa communauté grandissante, constituée de milliers de développeurs et de millions d'utilisateurs, pour proposer en permanence des innovations.


GitLab