Smartfuture
Certifications | Training | Technology solutions

Formation

CI/CD & Outils DevOps

Formation (présentiel ou distance) pour mettre en place une chaîne CI/CD moderne avec les outils DevOps clés (Git, GitHub/GitLab/Azure DevOps, Docker, pipelines, monitoring), afin d'industrialiser les livraisons et renforcer la collaboration entre développement et opérations.

Formation 100% pratique

Exercices & cas d'usage

Formateurs expérimenté

4 jours
24–28 h

Format
Ateliers & labs guidés

Modalité
Présentiel ou en ligne

Niveau
Initiation

INSCRIPTION / RÉSERVATION

Je m'inscris maintenant

OBJECTIFS PÉDAGOGIQUES

- Assimiler les principes essentiels de DevOps et du CI/CD.
- Mettre en place une intégration continue avec Git et un outil de CI.
- Automatiser build, tests et packaging applicatif.
- Déployer automatiquement sur les environnements de recette et de production.
- Utiliser Docker et intégrer les conteneurs dans la chaîne DevOps.

PUBLIC CIBLE

- Développeurs souhaitant automatiser la livraison de leurs applications.
- Ingénieurs systèmes / opérations souhaitant adopter les pratiques DevOps.
- Chefs de projet techniques, tech leads et architectes impliqués dans les chaînes CI/CD.
- Équipes souhaitant industrialiser leurs déploiements (on-premise ou cloud).

PRÉREQUIS

- Bases en développement logiciel (C#, Java, Python, JavaScript, etc.).
- Notions sur le fonctionnement d'une application web ou d'un service backend.
- Connaissances en administration système/réseau appréciées mais non obligatoires.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Introduction à DevOps & CI/CD

Vision globale DevOps

- Contexte : des méthodes traditionnelles aux approches agiles et DevOps.
- Objectifs de DevOps : collaboration, rapidité, qualité et fiabilité.
- Les grands principes : automatisation, feedback continu, mesure, partage.
- Cycle de vie logiciel moderne : planification, développement, tests, déploiement, exploitation.

Concepts d'intégration et de déploiement continus

- Définition de l'Intégration Continue (CI) et de ses bénéfices.
- Définition du Déploiement Continu (CD) et de la Livraison Continue.
- Chaîne outillée de bout en bout : du commit au déploiement.
- Notions d'environnement : développement, intégration, recette, préproduction, production.

Gestion de code source avec Git

Rappels Git

- Concepts de base : dépôt, commit, branche, merge.
- Commandes essentielles : clone, add, commit, push, pull, fetch.
- Gestion de branches : stratégie Git Flow / Trunk-based (vue d'ensemble).

Plateformes Git collaboratives

- Découverte de GitHub, GitLab et Azure DevOps Repos.
- Pull requests / Merge requests et revues de code.
- Gestion des permissions, des branches protégées et des règles de validation.

Bonnes pratiques Git pour CI/CD

- Commits fréquents, petits et significatifs.
- Messages de commit clairs et conventions d'équipe.
- Branches dédiées aux fonctionnalités, corrections et releases.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Mise en place d'une chaîne d'intégration continue (CI)

Principes d'un pipeline CI

- Déclencheurs (triggers) : push, pull request, planification.
- Étapes typiques : build, tests, packaging, analyse de qualité.
- Artefacts générés et archivage.

CI avec GitHub Actions / GitLab CI / Azure DevOps (selon l'outil retenu)

- Concepts de base : jobs, étapes (steps), runners/agents.
- Fichiers de configuration YAML : structure et bonnes pratiques.
- Variables d'environnement et secrets (mots de passe, clés API).

Automatisation du build & des tests

- Compilation ou packaging d'une application (ex. .NET, Java, Node.js ou Python).
- Exécution de tests unitaires et remontée des résultats dans l'outil de CI.
- Qualité de code (aperçu) : intégration simple avec SonarQube ou équivalent (présentation).

Docker & conteneurisation pour DevOps

Concepts Docker

- Rappels : machines virtuelles vs conteneurs.
- Images, conteneurs, registres (Docker Hub, registry privée).
- Dockerfile : structure, instructions de base (FROM, RUN, COPY, CMD, EXPOSE).

Construction et exécution d'images

- Build d'une image pour une application web / API.
- Lancement de conteneurs : ports, volumes, variables d'environnement.
- Introductions à docker-compose (multi-conteneurs simples).

Intégration de Docker dans les pipelines CI

- Build d'images Docker dans la CI.
- Push d'images vers un registre (Docker Hub, GitLab Registry, Azure Container Registry).
- Gestion des tags d'images (version, commit, branche).

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Déploiement continu (CD) vers les environnements

Principes de CD

- Différences entre Déploiement Continu et Livraison Continue.
- Stratégies de déploiement : manuel, automatique, approbations.
- Gestion des environnements et des variables de configuration.

Pipelines de déploiement

- Création de pipelines de release dans l'outil choisi (GitHub, GitLab, Azure DevOps…).
- Déploiement sur un serveur applicatif, une VM ou un service PaaS (ex. Azure Web App, conteneur).
- Utilisation de scripts (PowerShell, Bash) pour automatiser les actions de déploiement.

Stratégies de déploiement avancées (vue d'ensemble)

- Blue-Green, Rolling, Canary (concepts et cas d'usage).
- Back-out / rollback : retour rapide en cas de problème.
- Gestion des fenêtres de maintenance et communication avec les équipes.

Infrastructure as Code & configuration

Concepts d'Infrastructure as Code (IaC)

- Définition et avantages : reproductibilité, traçabilité, versionning.
- Présentation d'outils : Terraform, Ansible, ARM/Bicep, etc. (vue d'ensemble).

Exemple simple d'IaC

- Description d'une infrastructure simple en code (environnement de démo).
- Provisionnement automatisé d'un environnement de test.
- Intégration de l'IaC dans la chaîne CI/CD (concepts).

Gestion de la configuration

- Variables, secrets, stores de configuration (Key Vault, Secret Manager…).
- Gestion des paramètres selon les environnements.
- Bonnes pratiques de sécurité (jamais de secrets dans le code !)

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Supervision, logs & DevSecOps (aperçu)

Monitoring & observabilité

- Pourquoi superviser ? Indicateurs clés (disponibilité, temps de réponse, erreurs).
- Logs, métriques, traces : principes.
- Intégration avec des outils de monitoring / logging (ex. Prometheus, Grafana, ELK, Application Insights – selon contexte).

Introduction à DevSecOps

- Intégrer la sécurité au pipeline (Shift Left).
- Exemples d'outils : analyse de vulnérabilités, scan de dépendances.
- Bonnes pratiques de base : mots de passe, clés, gestion des accès.

Retour d'expérience & bonnes pratiques

- Organisation d'une équipe autour de DevOps.
- Petits pas d'industrialisation : par où commencer.
- Indicateurs de suivi : temps de cycle, fréquence de déploiement, taux d'échec.

Atelier de synthèse : mise en place d'une chaîne CI/CD de bout en bout

Mise en pratique guidée

- Création d'un dépôt Git pour une application exemple.
- Configuration du pipeline d'intégration continue (build + tests).
- Conteneurisation avec Docker et publication d'une image dans un registre.

Mise en place du déploiement continu

- Déploiement automatique sur un environnement de recette.
- Validation fonctionnelle rapide et promotion vers la production (scénario démonstratif).
- Simulation d'un rollback en cas de problème.

Synthèse & plan d'action

- Récapitulatif des concepts et outils vus durant la formation.
- Conseils pour adapter la chaîne CI/CD au contexte de l'entreprise.
- Échanges sur les prochaines étapes : pilotes, POC, déploiement à grande échelle