Smartfuture
Certifications | Training | Technology solutions

Formation

Git & GitHub Professionnel

Maîtrisez les fondamentaux et les pratiques avancées de Git et GitHub pour structurer vos dépôts, collaborer efficacement en équipe et industrialiser vos workflows de développement. Formation en distanciel (présentiel sur demande).

Formation pratique orientée projet

Formateurs experts DevOps & Git

Présentiel ou classe virtuelle à la demande

2 à 3 jours
14-21 h

Format
Atelier pratique

Modalité
Distanciel
Présentiel

Niveau
Intermédiaire

INSCRIPTION / RÉSERVATION

Je m'inscris maintenant

OBJECTIFS PÉDAGOGIQUES

- Comprendre les concepts fondamentaux du contrôle de version distribué avec Git.
- Savoir créer, gérer et exploiter des dépôts Git locaux et distants.
- Mettre en œuvre des workflows de branches professionnels (GitFlow, trunk-based, feature branches).
- Utiliser GitHub au quotidien : gestion de projets, issues, Pull Requests et revues de code.
- Intégrer Git et GitHub dans une démarche DevOps (CI/CD, automatisation, sécurité de code).

PUBLIC CIBLE

- Développeurs et ingénieurs logiciels.
- Architectes techniques, ingénieurs DevOps.
- Chefs de projets techniques, Team leaders.
- Étudiants en informatique ou profils en reconversion vers les métiers du développement.

PRÉREQUIS

- Connaissances de base en développement logiciel (quelques notions d'un langage comme Java, C#, Python, JS…).
- Être à l'aise avec l'utilisation d'un système d'exploitation (Windows, macOS ou Linux).
- Une première expérience de travail en équipe sur un projet est un plus.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Introduction à Git & au contrôle de version distribué

Concepts fondamentaux

- Pourquoi utiliser un système de contrôle de version ? Limites des approches manuelles.
- Centralisé vs distribué : positionnement de Git par rapport à SVN, TFS, etc.
- Vocabulaire Git : dépôt, commit, index (staging area), HEAD, branche, tag, remote.

Installation & configuration

- Installer Git sur Windows, macOS et Linux.
- Configurer l'identité (user.name, user.email) et les options globales.
- Configurer un éditeur de texte et les outils de fusion (merge tools).
- Notion de fichiers .gitconfig et .gitignore.

Premiers pas avec Git en local

Cycle de vie des fichiers

- Créer un dépôt local (git init).
- Suivi des fichiers : untracked, staged, tracked, modified.
- Commandes de base : git status, git add, git commit, git log.
- Rédiger des messages de commit de qualité.

Exploration de l'historique

- Visualiser l'historique : git log, options avancées (--oneline, --graph…).
- Inspecter un commit : git show, git diff.
- Comprendre les identifiants de commit (hash) et les références symboliques (HEAD, HEAD~1…).

Branches et fusions : organiser le travail

Gestion des branches

- Créer, lister et supprimer des branches : git branch, git checkout, git switch.
- Nommage des branches (feature/, hotfix/, release/…).
- Navigation entre les branches, bonnes pratiques de découpage.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Fusion & résolution de conflits

- Fusionner des branches : git merge (fast-forward vs merge commit).
- Gérer les conflits de fusion dans le code.
- Outils graphiques de résolution de conflits (VS Code, autres IDE).
- Stratégies pour limiter les conflits (taille des branches, synchronisation régulière).

Rebase et réécriture d'historique (introduction)

- Principe de git rebase et différences avec git merge.
- Utiliser git rebase --interactive pour nettoyer l'historique local.
- Bonnes pratiques et précautions avec le rebase.

GitHub : collaborer à l'échelle de l'équipe

Découverte de GitHub

- Créer un compte GitHub et configurer le profil.
- Organisation des dépôts : public, privé, organisations, équipes.
- Authentification HTTPS, SSH et GitHub CLI.

Dépôts distants & collaboration

- Créer un dépôt distant et le lier à un dépôt local (git remote).
- Synchronisation : git push, git pull, git fetch.
- Cloner un projet existant : git clone.
- Gestion des accès : collaborateurs, équipes, droits (read/write/admin).

Issues, Pull Requests & revues de code

- Créer et gérer des issues : templates, labels, milestones, assignment.
- Créer une Pull Request (PR), la commenter et la suivre.
- Processus de revue de code : commentaires, suggestions, approbations.
- Stratégies de fusion (merge commit, squash & merge, rebase & merge).

Workflows Git professionnels & bonnes pratiques

Workflows de branches

- Présentation des principaux workflows : GitFlow, GitHub Flow, trunk-based development.
- Choisir un workflow adapté à son contexte (TPE, grande entreprise, produit, projet interne…).
- Implémenter un workflow choisi avec Git & GitHub.

Bonnes pratiques au quotidien

- Structurer ses commits : petits, fréquents, cohérents.
- Normes de messages de commit (Conventional Commits, préfixes).
- Utiliser les tags pour marquer les versions.
- Politique de revue de code et validation par les pairs.

Intégration avec DevOps, CI/CD & sécurité

Automatisation avec GitHub Actions (introduction)

- Principes de base de GitHub Actions : workflows, jobs, runners.
- Créer un workflow simple pour lancer des tests à chaque push.
- Mettre en place un pipeline de build et de déploiement simple.

Sécurité & conformité

- Gestion des secrets et variables d'environnement.
- Éviter les fuites d'informations sensibles (mots de passe, clés API) dans Git.
- Présentation de fonctionnalités GitHub avancées : Dependabot, code scanning (aperçu).

Atelier de synthèse

- Mise en place complète d'un dépôt d'équipe avec branches, PR, revue de code et pipeline simple.
- Conseils pour l'intégration dans vos projets réels et accompagnement sur cas concrets.