Formation Git : Gestion de dépôts

Durée :	2 jours
Public :	Développeurs
Pré-requis :	Notions d'administration système
Objectifs :	Mettre en place une solution de configuration logicielle basée sur Git - Gérer les versions des projets du dépôt de données
Sanction :	Attestation de fin de stage mentionnant le résultat des acquis
Taux de retour à l'emploi:	Aucune donnée disponible
Référence:	DEV705-F
Demandeurs d'emploi:	Des entreprises recrutent des demandeurs d'emploi qui ont suivi ce cours dans le cadre d'une POEI, contactez-nous au 09.72.37.73.73 pour plus d'informations.
Note de satisfaction des participants:	4,80 / 5

Découvrir Git

Principes de gestion de contrôle de source (SCM)
Historique, contrôle local, centralisé et distribué
Fonctionnement des instantanés, comparaison avec les différences
Installation (Linux, MacOS, Windows)
Accès au manuel : man / help
Configuration initiale de Git : préférences, profil utilisateur
Initialisation d'un dépôt local

Atelier : Installation de Git - Création d'un projet

Comprendre le cycle de vie du répertoire de travail

Concepts de répertoire de travail, index et dépôt
Vérifier l'état de la copie de travail : status
Indexer ses modifications : add
Ignorer des fichier : .gitignore
Valider ses modifications : commit
Supprimer et déplacer des fichiers

Atelier : contributions et validations

Visualiser l'historique

Visualiser les modifications : log
Personnaliser le format : stat, pretty, …

Filtrer par date, auteur, message de commit, contenu modifié, ...
Visualiser et exporter une différence (format natif, outil externe)
Étiqueter ses validations : étiquettes légères et annotées
Rechercher avec git-grep

Annuler des actions

Réécrire la dernière validation
Désindexer un fichier
Réinitialiser un fichier

Travailler avec les branches

Principe de branche, le pointeur HEAD
Créer une branche
Basculer entre les branches, le mode détaché
Fusionner les branches : avance-rapide, trois sources
Gérer les conflits de fusion
Outil de fusion externe : mergetool (emerge, vimdiff, meld, …)
Visualiser les branches existantes, celles qui ont été fusionnées
Supprimer une branche
Stratégies de gestion de branches : branche longue, thématique, …

Travailler avec un dépôt distant

Dépôt distant, branches distantes, suivi de branche
Afficher et inspecter les dépôts distants
Ajouter, renommer, retirer ses dépôts distants
Tirer, pousser et supprimer une branche distante

Réécrire l’histoire, rebaser

Mise en garde : les dangers de la réécriture
Rebaser une portion de branche
Quand rebaser et quand fusionner

Remiser et nettoyer

Remiser son travail en cours
Créer une branche depuis une remise
Nettoyer son répertoire de travail

Personnaliser Git

Configurer l'éditeur par défaut, exclusions automatiques, ...
Création et utilisation d’alias
Outils graphiques : Git-Gui, GitKraken, SmartGit, ...
Créer des filtres : smudge et clean
Crochets côté client : pre-commit, pre-rebase, post-rewrite…
Crochets côté serveur : pre-receive, update, post-receive

Faire référence à un projet externe

Principe des sous-modules
Déclarer, tirer et mettre à jour un sous-module
Modifier et gérer les conflits sur une bibliothèque externe
Problèmes des sous-modules

Git sur un serveur

Les protocoles : local, HTTP, SSH, Git
Création d’un dépôt nu, comptes utilisateurs
Utilisateur git unique, clés SSH et git-shell
Démon Git

Atelier : Mise en place d’un serveur Git

Gestion de dépôt web

Un serveur simple et léger : GitWeb
Une plate-forme plus complète : GitLab
GitLab : configuration des utilisateurs
GitLab : exploration de projet, suivi des activités, wiki
GitLab : issue manager, web hooks, revue de code
Un service hébergé clé-en-main : GitHub
GitHub : création de compte et configuration
GitHub : règles de contribution
GitHub : maintenance d’un projet

Atelier : Récupération et exploration d’un GitLab

Conclusion

Git et le cycle de vie du projet
Git et l’intégration continue : exemple de GitLab