EduGroupe
Accompagner pour réussir

Gestion de versions avec GIT

Référence : **LUUX151** Durée : **2 jours (14 heures)** Certification : **Aucune**

Connaissances préalables

- Connaissance des processus de développement et d'un langage de programmation

Profil des stagiaires

- Tout développeur, chef de projet, architecte, souhaitant utiliser git comme gestionnaire de versions

Objectifs

- Comprendre les principes d'un gestionnaire de version distribué, les apports de git, savoir le mettre en oeuvre pour gérer les codes sources d'un projet, les versions, corrections de bugs, etc

Certification préparée

- Aucune

Méthodes pédagogiques

- Mise à disposition d'un poste de travail par participant
- Remise d'une documentation pédagogique papier ou numérique pendant le stage
- La formation est constituée d'apports théoriques, d'exercices pratiques et de réflexions

Formateur

- Consultant-Formateur expert Développement Internet

Méthodes d'évaluation des acquis

- Auto-évaluation des acquis par le stagiaire via un questionnaire
- Attestation des compétences acquises envoyée au stagiaire
- Attestation de fin de stage adressée avec la facture

Contenu du cours

1. Présentation de Git

- La notion de gestionnaire de versions distribué
- Historique de git, licence
- Présentation des principes techniques de git : architecture, les objets stockés
- Les différentes utilisations de git : utilisation d'applicatifs stockés sous git, développement, partage de codes, gestions de modifications, de correctifs etc.
- Aperçu des types de workflows possibles

2. Prise en main

- La commande git, options principales
- Installation et configuration de git.Présentation des notions de base : référentiel, index, répertoire de travail, clônage
- Travaux pratiques : Création d'un premier dépôt.Utilisation de la ligne de commande pour les opérations de base

3. Gestion des développement

- Etude des commandes principales de manipulation des fichiers :add, status, diff, commit, ...
- Gestion des branches : branch, checkout, merge, log, stash, etc.
- Travaux pratiques : Mise en oeuvre sur un projet exemple représentatif des principaux cas d'utilisation
- Résolution des conflits
- Intérêt des branches temporaires

4. Travail collaboratif

- Objectif : partage et mise à jour de projets
- Fonctionnalités requises : mise à disposition des objets, analyse des modifications, intégration,etc.
- Définition des rôles (développeurs, intégrateurs)
- Notion de dépôt local et dépôt centralisé
- Etude des commandes : fetch, pull, push, remote, ...
- Pour le contrôle de fichiers : show, log, diff, ...
- Gestion des patchs : apply, rebase, revert, ...
- Travaux pratiques : Connexion à un réferentiel

5. Administration

- Tâches d'administration : nettoyage des arborescences, vérification de la cohérence de la base de données, état du service git
- Travaux pratiques : Installation d'un dépôt privé centralisé pour une gestion de sources collaborative, import de développements externes avec fast-import

6. Compléments

- Interagir avec des référentiels partagés via GitHub
- Exemples de projets sur GitHub, GitLab
- Présentation d'outils complémentaires : gerrit, un système de revue de code.Gitweb, l'interface web
- GitKraken, client graphique

7. Bonnes Pratiques

- Echanges par rapport aux contextes projets et à l'organistion des équipes pour savoir définir l'utilisation de git la plus adaptée à chaque contexte projet

Notre référent handicap se tient à votre disposition au 01.71.19.70.30 ou par mail à referent.handicap@edugroupe.com pour recueillir vos éventuels besoins d'aménagements, afin de vous offrir la meilleure expérience possible.