Test Driven Development en Java/JEE

★★★★★

MR-179 **3 Jours (21 Heures)**

Description

Ce cours vous apportera toutes les connaissances nécessaires pour développer vos applications Java/Java EE autour des meilleurs outils et pratiques de tests unitaires et de recettes. Vous apprendrez à intégrer le test dans votre cycle de développement, le Refactoring, la conception émergente et les pratiques agiles.

À qui s'adresse cette formation ?

Pour qui

Ingénieurs ou chefs de projets en développement logiciel.

Prérequis

Aucune

Les objectifs de la formation

- Maîtriser la démarche et la mise en oeuvre du Test Driven Development Intégrer les tests dans le cycle de développement d'une application Java/Java EE Prendre en main les principaux outils de tests et d'intégration continue

Programme de la formation

Définition et principes du TDD

- Le test dans le processus de développement.
- Processus, qualité, tests.
- Typologie des tests.
- Origine du TDD.
- L'agilité et les tests.
- Cycle de développement.
- Les 3A.
- Gestion des exceptions.
- Refactoring et conception émergente.
- Gestion des scénarios.
- Gains du TDD ? Travaux pratiques Conception et intégration de tests dans le cycle de développement d'?un projet.

Tests automatisés avec le framework JUnit

- Le besoin d'un framework de test.
- JUnit.
- Alternatives (TestNG) et outillage complémentaire.
- Bonnes pratiques associées à JUnit.
- Travaux pratiques Mise en oeuvre de JUnit.

Les bonnes pratiques de développement Agiles

- TDD et gestion des données SGBDR, des interfaces graphiques, des interfaces Web.
- Travaux pratiques Mise en oeuvre de pratiques.

Techniques avancées avec le TDD

- Corriger des anomalies.
- Gérer la montée en charge, la sécurité des produits.
- Gestion de la sécurité logicielle.
- Gestion de la performance.
- Stress tests.
- Travaux pratiques Gestion des anomalies.
- Tests de performance.

Les objets Mock et Stub

- La théorie.
- Application de la théorie sans utiliser de bibliothèque.
- Découverte des bibliothèques du marché.
- Etude en détail de JMOCK ou MockObject.
- Travaux pratiques Utilisation des objets Mock.

Techniques d'écriture de tests

- Fixtures.
- Qualités d'un code de test.
- Tests basés sur la responsabilité, l'implémentation.
- Styles de TDD.
- Travaux pratiques Améliorer la qualité des tests écrits.

Test de code hérité

- Qu'est-ce que du code hérité ? Cycle d'évolution du code hérité.
- Tests fonctionnels avec Fit et FitNesse.
- Tests fonctionnels et TDD.
- Exécution de tests fonctionnels avec FitNesse.

Les outils

- Les outils Open Source et commerciaux.
- Architecture matérielle de tests.
- Etude d'un outil d'intégration continue.
- Etude et choix d'un intégrateur continu.
- Etude d'un outil de couverture de test.
- Etude d'un outil de gestion des tests et de communication entre MOA et MOE : Fitnesse.
- Travaux pratiques Mise en oeuvre de plusieurs outils.