Tél : (+33)1 55 60 76 72
Email : formation@oxiane.com

TDD, BDD et principes SOLID

Développement dirigé par les tests : TDD, BDD et principes SOLID

La réussite d'un projet logiciel repose sur la qualité de sa base de code et l'adéquation entre ce qui est développé et les besoins des utilisateurs finaux. La mise en application des principes SOLID permet de garantir la qualité du code produit et sa maintenabilité à long terme.
Cette formation s'appuie sur cette première présentation et présente ensuite la pratique du développement dirigé par les tests, déclinée en deux volets. Le premier, le TDD, dans lequel les tests sont écrits de façon classique et le second, le BDD qui repose sur des tests écrits en langage naturel.
La formation s'adresse à des développeurs Java expérimentés ayant une bonne connaissance du langage et de ses API fondamentales. Une première dans l'écriture de tests sur du code legacy est un avantage.

A propos de José Paumard :
José est universitaire et indépendant depuis une vingtaine d'années, Java Champion, Java Rockstar et auteur pour Pluralsight . Coach en Software Craftsmanship depuis 4 ans, entre autres dans une grande banque Parisienne, il publie un catalogue raisonné de katas craft sur GitHub.io. Il enseigne les technologies Java / Java EE à l'université Paris 13 en école d'ingénieur et intervient auprès de sociétés en formation, architecture et expertise. Il est speaker invité à JavaOne (San Francisco), et intervient régulièrement à Devoxx (Belgique, France, Angleterre) et dans de nombreuses autres conférences européennes (JPrime, JFokus, JTres, etc…) . Il publie des articles pour Java Magazine, Oracle Technology Network et anime le blog technique "Java le soir". Il est cofondateur de Devoxx France qu'il a coorganisé les 3 premières années. Il est actuellement trésorier de l'association BJPC, organisatrice des soirées du Paris JUG. Il est enfin membre du groupe d'experts pour CDI 2.0 (Context dans Dependency Injection, JSR 365).

Détails

- Code : MP-TDD
- Durée : 2 jours (14 heures)

Public
- Développeurs expérimentés

Pré-requis
- Bonne maîtrise du langage Java et de ses API fondamentales
- Une première expérience dans l'écriture de tests sur du code legacy est un avantage

Objectifs
- Comprendre le rôle des tests dans le processus de développement des logiciels, les enjeux de l'écriture de tests automatisés et le coût de la mise en production de code non testé automatiquement
- Comprendre ce qu'est un test d'acceptance et comment découper une fonctionnalité en tests d'acceptance
- Maîtriser l'écriture de tests automatisés, unitaires et tests d'intégration ainsi que les différentes étapes du développement TDD : découpage d'un problème en tâches élémentaires, écriture de tests ciblés et recouvrant, cycle TDD ;
- Maîtriser les principes et patterns du « clean code » en programmation objet, les outils de frameworks de test, la syntaxe Gherkin pour l'écriture de tests d'acceptance, le framework Cucumber pour exécuter des tests d'acceptance
- Etre capable d'appliquer ces principes et patterns au développement de nouveau code (greenfield development) et au code existant (brownfield development)

Programme

Le programme se compose de 30% présentation sur slides, 70% de codage. La partie codage porte principalement sur le codage de « Katas », un exercice essentiel pour s'exercer à la pratique du TDD / BDD.

Introduction à la pratique du développement dirigé par les tests, le cycle de développement TDD

- Exemple du Kata FizzBuzz
- Développement du Kata
- Bilan : les étapes de la pratique du TDD

Introduction des principes du Clean Code et des principes SOLID

Pratiques de l'eXtreme Programming

- Pratiques des Katas
- Pratiques des Coding Dojo
- Pratiques du Pair Programming

Le principe Open / Close : application au pattern Strategy

- Exemple du Kata RPN Calculator
- Développement du Kata
- Bilan : le pattern Strategy et son implémentation

Utilisation de Gherkin / Cucumber pour l'écriture de tests

- Écriture de tests en Gherkin : méthodes, organisation, syntaxe
- Écriture de classes Cucumber pour l'exécution de ces tests
- Fonctionnalités avancées : tests paramétrés, tables de données, tags
- Intégration avec Maven, génération de rapports de tests

Mise en œuvre sur un kata complexe : Mars Rover

- Écriture des tests avec Cucumber
- Résolution du kata
- Conclusion

Modalités

- Type d'action : Acquisition des connaissances
- Moyens de la formation : Formation présentielle – 1 poste par stagiaire – 1 vidéo projecteur – Support de cours fourni à chaque stagiaire
- Modalités pédagogiques : Exposés – Cas pratiques – Synthèse
- Validation : Exercices de validation – Attestation de stages
- Profils des intervenants : Toutes nos formations sont animées par des consultants-formateurs expérimentés et reconnus par leurs pairs.