Tél : +352 27 39 351
Fax : +352 27 39 52 68
Email : formation@oxiane.lu

Software Craftsmanship par José Paumard

Clean Code, TDD, BDD et principes SOLID

Un logiciel peut être parfaitement fonctionnel mais poser de nombreux problèmes de fiabilité et de maintenabilité. Le Software Craftsmanship ou « l'artisanat du logiciel » propose un ensemble de méthodes et d'approches de haute qualité pour concevoir et construire des bases de code de tous volumes maintenables à coût constant.
La formation s'adresse à des développeurs Java expérimentés ayant une bonne connaissance du langage et de ses API fondamentales. Une première confrontation aux problèmes de gestion de bases de code anciennes, volumineuses et non (ou mal) testées 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-SCS
- Durée : 3 jours (21 heures)

Public
- Développeurs

Pré-requis
- La formation s'adresse à des développeurs Java expérimentés ayant une bonne connaissance du langage et de ses API fondamentales. Une première confrontation aux problèmes de gestion de bases de code anciennes, volumineuses et non (ou mal) testées est un avantage.

Objectifs
- Comprendre la notion de développement dirigé par la valeur
- Maîtriser les différentes étapes du développement TDD
- Maîtriser les principes SOLID et les patterns du « clean code » en programmation objet
- Comprendre les pratiques de l'eXtreme Programming : pair programming, coding dojos
- Être capable d'appliquer ces principes et patterns au développement de nouveau code (greenfield development) et au code existant (brownfield development)

Programme

Remarque
- 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
- Exercice pratique : Elephant Carpaccio
- Mise en œuvre du cycle de développement
- Bilan

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

Pratiques de l'eXtreme Programming
- Pratiques des Katas
- Pratiques des Coding Dojo
- Pratiques du Pair Programming

Tél : +352 27 39 351
Fax : +352 27 39 52 68
Email : formation@oxiane.lu

Introduction des principes SOLID

Le principe Open / Close : application au pattern Strategy

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

Le Single Responsibility Principle : application au Kata Rental Movie

- Exemple du Kata Rental Movie (code legacy)
- Développement du Kata
- Bilan : détecter les manquements au SRP

Travail sur le code legacy

- Spécificités du travail sur code legacy
- Principes fondamentaux du travail sur code legacy
- Méthodes, approches

Katas code legacy

- Exemple du Kata Rental Movie
- Développement du Kata
- Bilan : étapes du développement

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
- Utilisation de fonctionnalités avancées : tests paramétrés, tables de données
- Utilisation d'annotations
- Intégration avec Maven, génération de rapports de tests
- Mise en œuvre sur un kata complexe : Mars Rover

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