Développez vos talents
Certifiez vos compétences

Tests unitaires

Pratique des tests unitaires avec JUnit

2 jours (14 heures) | ★★★★½ 4,6/5 | TEST-JUNIT | Évaluation qualitative de fin de stage | Formation délivrée en présentiel ou distanciel [1]

Formations Informatique › Tests › Tests unitaires

À l'issue de ce stage vous serez capable de :

- Comprendre les principes de développement des tests
- Maîtriser JUnit.

Niveau requis

Avoir une pratique du langage Java.

Public concerné

Architectes, chefs de projets, consultants ou ingénieurs.

Cette formation :

- Est animée par un consultant-formateur dont les compétences techniques, professionnelles et pédagogiques ont été validées par des diplômes et/ou testées et approuvées par l'éditeur et/ou par M2i Formation
- Bénéficie d'un suivi de son exécution par une feuille de présence émargée par demi-journée par les stagiaires et le formateur.

(1) Modalité et moyens pédagogique :

Formation délivrée en présentiel ou distanciel * (e-learning, classe virtuelle, présentiel à distance). Le formateur alterne entre méthodes ** démonstrative, interrogative et active (via des travaux pratiques et/ou des mises en situation). La validation des acquis peut se faire via des études de cas, des quiz et/ou une certification.

Les moyens pédagogiques mis en oeuvre (variables suivant les formations) sont : ordinateurs Mac ou PC (sauf pour les cours de l'offre Management), connexion internet fibre, tableau blanc ou paperboard, vidéoprojecteur ou écran tactile interactif (pour le distanciel). Environnements de formation installés sur les postes de travail ou en ligne. Supports de cours et exercices.

* Nous consulter pour la faisabilité en distanciel. ** Ratio variable selon le cours suivi.

Programme

Introduction

- Pourquoi tester ?
- Différents types de tests
- Coût des tests
- Qu'est-ce qu'un test unitaire ?
- Quels tests réaliser ?
- Automatiser les tests

JUnit 3 : les concepts de base

- Cas de test (TestCase)
- Contraintes et conventions
- Structure d'un test
- Assertions, échec et erreur
- Exécution d'un TestCase
- Exécution automatique
- Résultat des tests
- Gestion du cycle de vie
- Factoriser les fixtures
- Test de levée d'exception
- Organiser les tests (TestSuite)
- Bonnes pratiques : comment tester ?
- Notion de couverture de tests
- Autres aspects avancés de JUnit3
- Limitations de JUnit3

JUnit 4

- Ecriture des TestCase avec les annotations
- Cycle de vie avec les annotations
- Exécution des tests avec JUnit 4
- Assertions avec
  - JUnit 4
  - Hamcrest
  - AssertJ
- Test de levée d'exception
- Test de durée
- Désactivation d'un test
- Suppositions (assumptions)
- Suite de tests avec les annotations
- Organiser ses tests avec les catégories
- Tests paramétrés
- Concept de règle (Rule)
- Rules proposées

Mocks

- Motivation et principe
- Différents types de simulacre
- Concevoir "testable"
- Mock statique / dynamique
- Frameworks de mocking
- EasyMock : les bases
- Principe du record - replay - verify
- Vérifications plus complexes
- JMockit : pour le code non testable
- Mockito : un des plus utilisés
- PowerMock : pour compléter EasyMock / Mockito
- Bonnes pratiques

JUnit 5

- Nouvelle architecture
- Correspondance des concepts / annotations
- Nouvelles annotations
- Etiquettes (Tag)
- Nouvelles assertions
- Suites de tests avec JUnit 5
- Tests paramétrés avec sources
- Tests répétés
- Tests dynamiques
- Les tests imbriqués
- Tests dans les interfaces

Pour aller plus loin

- Principes du TDD (Test Driven Development)
- Principes du mutation testing
- Intégration continue
- Tests d'intégration
- Principes du BDD (Behavior Driven Development)