+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Mis à jour le 05/08/2025

S'inscrire

Formation JUnit : Test Java efficace

2 jours (14 heures)

Présentation

Maîtrisez les tests unitaires en Java avec JUnit 5 grâce à cette formation complète et opérationnelle. Conçue pour les développeurs et équipes qualité, elle vous apportera les compétences essentielles pour écrire, structurer et maintenir des tests efficaces, fiables et automatisables dans vos projets Java.

Vous débuterez par les fondamentaux du testing, les types de tests et l'architecture de JUnit 5. Vous apprendrez à rédiger des tests clairs et robustes avec les annotations clés, les assertions avancées, et l'exécution dans différents environnements.

La formation vous initiera aux tests paramétrés, aux tests imbriqués, à l'utilisation de Mockito pour simuler les dépendances, et aux bonnes pratiques d'organisation de votre code de test pour un projet maintenable.

Vous intégrerez vos tests dans une chaîne CI/CD et découvrirez les principes du TDD (Test-Driven Development) avec des ateliers pratiques sur des cas métiers réels et testables via Spring Boot.

Comme pour toutes nos formations, celle-ci vous sera présentée avec les toutes dernières actualisations de JUnit.

Objectifs

- Comprendre l'écosystème RapidMiner et le cycle de vie CRISP-DM
- Configurer l'environnement RapidMiner Studio et importer des données multi-sources
- Préparer, transformer et enrichir les jeux de données pour l'analyse
- Concevoir, entraîner et évaluer des modèles prédictifs et non supervisés
- Automatiser les workflows analytiques avec Auto Model et l'optimisation des paramètres
- Intégrer RapidMiner dans un processus de production analytique fiable et industrialisé

Public visé

- Développeurs Java
- Ingénieurs QA

Pré-requis

- Maîtrise des bases du langage Java

Programme de la formation JUnit : Test Java efficace

Introduction aux tests en Java

- Détection précoce des bugs
- Documentation vivante
- Régression et refactoring sécurisé
- Tests unitaires vs tests d’intégration
- Tests fonctionnels et E2E
- Test pyramid et bonnes pratiques d’équilibrage
- Présentation de JUnit
- Alternatives : TestNG, Spock, Mockito, AssertJ

Prise en main de JUnit 5

- Jupiter
- Platform
- Vintage
- Création de projet Maven/Gradle avec dépendances JUnit
- Exemple de test simple avec @Test
- Exécution avec IDE et ligne de commande
- @Test, @BeforeEach, @AfterEach, @BeforeAll, @AfterAll
- @Disabled, @DisplayName
- Utilisation des assertions

Écrire des tests unitaires efficaces

- Données d’entrée, action, vérification

- Lisibilité et clarté
- @ParameterizedTest
- Sources : @ValueSource, @CsvSource, @MethodSource, @EnumSource
- @Nested pour structurer les tests
- Regrouper les cas de test similaires
- assertAll, assertThrows, assertTimeout
- Comparaison avec AssertJ pour des assertions plus lisibles

Doubles de test

- Pourquoi utiliser des mocks ?
- Concepts : stub vs mock vs spy
- Intégration de Mockito avec JUnit 5 (@ExtendWith(MockitoExtension.class))
- @Mock, @InjectMocks, when(), verify()
- Tester un service dépendant d’un DAO ou d’un web service
- Simuler des exceptions ou comportements spécifiques

Organisation et gestion des tests

- Convention de nommage
- Organisation par package/module
- Isolation des cas de test
- Méthodes d’aide
- Fixtures et Builders
- @Tag, exécution conditionnelle
- Groupes de tests : unitaires, intégration, lente, rapide

Bonnes pratiques et pièges à éviter

- Rapide, fiable, lisible, isolé, reproductible
- Tests trop complexes
- Tests dépendants de l’environnement
- Assertions multiples non pertinentes
- Outils : JaCoCo, SonarQube
- Mesurer != garantir la qualité

Intégration dans le cycle de développement

- Intégration avec GitHub Actions, GitLab CI, Jenkins

- Failsafe vs Surefire pour les tests d’intégration
- Introduction au Test-Driven Development
- Cycle : Rouge ? Vert ? Refactor
- Atelier pratique de TDD
- Spring Boot : @SpringBootTest, @WebMvcTest
- Tests de repository avec @DataJpaTest

Atelier final de mise en pratique

- Application Java simple
- Structuration du code de test
- Couverture des cas nominaux et d’erreur
- Tests de service avec mocking
- Tests d’un contrôleur REST avec Spring

Sociétés concernées

Cette formation s’adresse à la fois aux particuliers ainsi qu’aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée ou bien à acquérir des connaissances métiers spécifiques ou des méthodes modernes.

Positionnement à l'entrée en formation

Le positionnement à l'entrée en formation respecte les critères qualité Qualiopi. Dès son inscription définitive, l'apprenant reçoit un questionnaire d’auto-évaluation nous permettant d’apprécier son niveau estimé sur différents types de technologies, ses attentes et objectifs personnels quant à la formation à venir, dans les limites imposées par le format sélectionné. Ce questionnaire nous permet également d’anticiper certaines difficultés de connexion ou de sécurité interne en entreprise (intraentreprise ou classe virtuelle) qui pourraient être problématiques pour le suivi et le bon déroulement de la session de formation.

Méthodes pédagogiques

Stage Pratique : 60% Pratique, 40% Théorie. Support de la formation distribué au format numérique à tous les participants.

Organisation

Le cours alterne les apports théoriques du formateur soutenus par des exemples et des séances de réflexions, et de travail en groupe.

Validation

À la fin de la session, un questionnaire à choix multiples permet de vérifier l’acquisition correcte des compétences.

Sanction

Une attestation sera remise à chaque stagiaire qui aura suivi la totalité de la formation.