Maîtriser le Test-Driven Developement et les patterns de test avancés

Date et durée

Code formation : DEV004-FR
Durée : 2 jours
Nombre d'heures : 14 heures

Description

Le Test-Driven Development (TDD) est une méthodologie de développement logiciel où **les tests sont écrits avant le code** lui-même. Cette approche, guidée par les tests, permet de produire un code de meilleure qualité, plus robuste et plus maintenable. En complément du TDD, les patterns de test avancés offrent **des techniques éprouvées** pour structurer et organiser les tests de manière efficace. Cette formation de 2 jours vous permettra d'acquérir une maîtrise solide de ces concepts et de les mettre en pratique dans un environnement JavaScript (TypeScript).
Au cours de cette formation, vous apprendrez à **mettre en œuvre le TDD de manière rigoureuse**, en maîtrisant les tests unitaires, l'intégration continue et les différentes stratégies de test. Vous explorerez les patterns de test avancés pour structurer votre code de tests de manière efficace et maintenir une base de tests solide au fil du temps. Vous découvrirez également **les concepts clés du refactoring et de l'injection de dépendances**, essentiels pour un développement agile et de qualité. Des exemples concrets d'applications web et Node.js vous permettront de mettre en pratique vos nouvelles compétences.
En conclusion, ce programme de 2 jours couvre l'ensemble des aspects du TDD et des patterns de test avancés. Vous apprendrez à **maîtriser les tests unitaires, le refactoring, l'injection de dépendances** et bien d'autres concepts clés.

Objectifs

A l'issue de cette formation avancée en TDD, vous atteindrez les objectifs suivants :

- mettre en œuvre le développement guidé par les tests (TDD) dans vos projets ;
- élaborer une couverture de tests exhaustive afin de garantir une qualité de code élevée et faciliter les déploiements fréquents ;
- maîtriser les différents types de tests (unitaires, d'intégration, etc.) et les appliquer efficacement ;
- structurer et maintenir une base de tests solide et évolutive ;
- approfondir les concepts de refactoring pour améliorer continuellement la qualité de votre code ;
- acquérir les fondamentaux de la clean architecture pour concevoir des applications robustes et évolutives.

Points forts

Un formateur expert et consultant en ingénierie logicielle, des démonstrations de live coding, des exercices pratiques en live, des exemples de clean architecture avec react.js tirés de projets réels et un code source disponible sur GitHub.

Modalités d'évaluation

Travaux Pratiques

Pré-requis

Suivre cette formation nécessite les prérequis suivants :

- **Matériel :** un ordinateur équipé d'une caméra, d'un micro et d'une connexion internet stable.
- **Logiciels :**
  - la dernière version stable de Node.js ;
  - la dernière version stable de Docker ;
  - Un environnement de développement tel que Visual Studio Code ou WebStorm.
- **Compétences :**
  - de solides connaissances en JavaScript et TypeScript (compréhension des types, interfaces, etc.) ;
  - une expérience dans l'écriture de tests unitaires ;
  - des notions de base sur les concepts de programmation orientée objet.

Info Stack Technique : TypeScript et Vitest.

Public

Cette formation s'adresse aux publics suivants :

- tous professionnels ayant au moins 2 ans d'expérience en développement back-end comme :
  - les développeurs back-end souhaitant approfondir leurs connaissances en architecture logicielle et passer au niveau supérieur ;
  - les tech leads désireux d'optimiser la conception et la maintenance des applications de leur équipe ;
  - les architectes logiciels en quête de nouvelles approches pour concevoir des systèmes plus robustes et évolutifs.

Programme

Jour 1 : Introduction aux fondamentaux du TDD

- Accueil et tour de table :
  - vos attentes et présentation du programme prévu.
- Premiers pas avec le TDD :
  - les bénéfices du TDD (qualité, agilité et productivité) ;
  - les principes clés du TDD (Red-Green-Refactor) ;
  - le choix du langage et du modèle de test.
- Labs en ligne :
  - exercices de katas de niveau 1 (FizzBuzz, calculatrice, etc.) pour se familiariser avec le cycle du TDD.
- Test doubles et injection de dépendances :
  - les différents types de test doubles (mocks, stubs et spies) ;
  - les avantages de l'injection de dépendances pour l'écriture de tests isolés.
- Labs en ligne :
  - exercices de katas de niveau 2 impliquant des interactions avec des dépendances externes (bases de données, API, etc.).

Jour 2 : Approfondissement et montée en compétences

- London School of TDD et Endo-Testing :
  - Présentation de l'approche de la London School of TDD ;
  - L'utilisation de l'Endo-Testing pour tester le comportement interne d'une classe.
- Mocks avancés :
  - l'utilisation des mocks pour simuler des comportements complexes et vérifier les interactions ;
  - les pièges à éviter avec les mocks (over-mocking et fragile tests).

- Labs en ligne :
    - exercices de katas de niveau 3 avec 1 ou 2 vrais projets complets (générateur de QCM sur console, application de todos React, plateforme de création de compte, etc.).
- Retour d'expérience :
    - discussion sur les difficultés rencontrées et les leçons apprises sous forme de questions réponses.