

## QUALITÉ LOGICIELLE ET TESTS - LES FONDAMENTAUX

<b>Durée</b>	<b>4 jours</b>	<b>Référence Formation</b>	<b>4-IT-QBAS</b>
--------------	----------------	----------------------------	------------------

### Objectifs

- Comprendre la problématique de la complexité des développements logiciels
- Comprendre le bénéfice de l'intégration continue pour améliorer la qualité des développements
- Adopter les bonnes pratiques de rédaction des tests logiciels
- Mettre en place une stratégie de campagne de tests automatisés avec les Framework et outils du marché
- Travailler avec un gestionnaire de code source tel que Git, et adopter les réflexes du travail collaboratif
- Comprendre le rôle des différents outils d'une chaîne d'intégration continue
- Utiliser un outil de construction logiciel pour automatiser les tâches de développement
- Mettre en place une plateforme d'intégration continue autour de Jenkins

### Participants

Développeurs, chefs de projets, architectes logiciels

### Pré-requis

Posséder les connaissances et compétences équivalentes aux fondamentaux du développement Java ou aux fondamentaux du développement .NET ou aux fondamentaux de la programmation orientée objet en C++

### Moyens pédagogiques

- Réflexion de groupe et apports théoriques du formateur
- Travail d'échange avec les participants sous forme de réunion-discussion
- Utilisation de cas concrets issus de l'expérience professionnelle
- Validation des acquis par des questionnaires, des tests d'évaluation, des mises en situation et des jeux pédagogiques
- Remise d'un support de cours.

## PROGRAMME

### Introduction

- Pratiques d'ingénierie logicielle et méthodes Agiles
- Le développement incrémental et itératif
- L'équipe Agile. Scrum et XP

### Les tests agiles

- Définition et périmètre des tests agiles
- Cycle de développement : origine du TDD (Test Driven Development), ATDD, TDR, les types de tests...

### Les tests développeurs

- Définition et objectifs : les patterns basiques XUnit
- Principe des tests unitaires automatisés
- Règles de simplicité : règle des "3 A" (Arrange, Act, Assert)
- Mise en œuvre de tests unitaires avec JUnit, le framework de test en Java
- Lanceur de tests (TestRunner)
- Les méthodes d'Assertions

### Le TDD, développement guidé par les tests

- Le cycle de développement
- Le principe du TDD : "test first", "tester, coder, refactorer"

### CAP ÉLAN FORMATION

[www.capelanformation.fr](http://www.capelanformation.fr) - Tél : 04.86.01.20.50

Mail : [contact@capelanformation.fr](mailto:contact@capelanformation.fr)

Organisme enregistré sous le N° 76 34 0908834

version 2024

- TDD et pratiques agiles (XP) : l'intégration continue, le Pair Programming

### **"Refactoring", le remaniement de code**

- Principes du refactoring
- Réduire l'apparition de la dette technique, rendre le code compréhensible
- Comment identifier le code à risque ? La notion de "Code Smells", signes de danger potentiel
- Les principales opérations de refactoring
- Rappel sur les Design Patterns

### **Isolation des tests**

- Les doubles de test, leur utilisation
- Le "Mock Object" pour vérifier certaines hypothèses
- Le "Fake", pour la simulation
- Le "Stub" : fournir une réponse prédéfinie à un appel

### **Le test comme cahier des charges, la notion d'ATDD**

- Les principes et avantages de l'ATDD
- Du scénario au test de recette.
- Combiner ATDD, BDD et TDD
- Les outils (Fitnesse, Cucumber...)

### **Conclusions**

- Les bénéfices du TDD, le coût des tests
- Les autres types de tests (interface graphique, Web..)
- Quelques outils