

Automatiser les tests logiciels



Spoc pratique
**Travaux pratiques
continus**



Durée
4 jours



Votre animateur
Olivier Charles



Code Kertify : **ATL**

S'inscrire →

www.kertify.com/inscription

Mieux comprendre l'automatisation tout au long du processus projet/produit

Cette formation vous donnera une vision globale sur l'automatisation des tests d'un logiciel tout au long de son processus de fabrication. Les maîtrise d'ouvrage et maîtrise d'œuvre pourront compléter leur culture et acquérir un vocabulaire complémentaire qui facilitera la communication et la compréhension avec les autres parties prenantes.

Les objectifs de la formation

- ◆ Comprendre les outils et techniques d'automatisation de la gestion des tests
- ◆ Comprendre les outils et techniques d'automatisation de l'exécution des tests
- ◆ Comprendre les enjeux, les risques liés à l'automatisation des tests fonctionnels

Participants

- ◆ Toute personne intéressée par les tests automatiques des logiciels et leur gestion de la conception à la livraison

Prérequis

- ◆ Ouverture vers la technique et le développement conseillée pour les profils maîtrise d'ouvrage.
- ◆ Ouverture vers la gestion des tests et les exigences pour les profils maîtrise d'œuvre logiciel est souhaitable

Le programme de notre formation

1. Introduction, rappels sur le processus de test d'un logiciel

- ◆ Le rôle du test dans le processus de développement
- ◆ Les tests : unitaires, fonctionnels, etc.
- ◆ Les différentes méthodes de test
- ◆ Le processus de test et la stratégie de test
- ◆ Les outils et méthodes intervenant à différentes étapes

2. Automatisation de la gestion des tests

- ◆ Gestion de la couverture des exigences par les tests. Notion de couverture et de granularité
- ◆ Démarche de mise au point : organisation des suites de tests et création des cas
- ◆ Faut-il automatiser un test ? Critères à prendre en compte ?

- ◆ Préparation à l'automatisation
- ◆ Construction de la population de test
- ◆ Mise au point et vérification des tests (Revue)
- ◆ Exécution, enregistrement des anomalies
- ◆ Notion de rapport d'incident d'après l'IEEE
- ◆ Gestionnaires d'anomalies
- ◆ Automatisation de la création des anomalies
- ◆ Analyse de résultats d'exécution de tests
- ◆ Consolidation des tests
- ◆ **Travaux pratiques.** Utilisation d'un référentiel de tests, paramétrages de ponts vers des outils BTS

3. Automatisation des tests unitaires

- ◆ Organisation et bonnes pratiques pour les tests unitaires
- ◆ Critères d'automatisation
- ◆ Tests unitaires : Tests Driven Development
- ◆ Mesure de la couverture de code : couverture des tests structurels, couverture d'instructions et branches
- ◆ Analyse statique de code : analyse outillée du code source hors exécution (règles de codage) : Checkstyle, Cobertura
- ◆ Automatisation avec un fichier de configuration
- ◆ Analyse dynamique de code : couverture des instructions, des branches, des prédicats...
- ◆ Automatisation avec un outil d'analyse de couverture
- ◆ Organisation des tests unitaires, pair programming, pair testing
- ◆ Utilisation des frameworks : gestion des scripts de tests, gestion des données de tests, récupération des résultats
- ◆ **Travaux pratiques.** Mise en œuvre de tests unitaires avec Junit/TestNG. Intégration dans une chaîne de fabrication de logiciels (Ant/Maven...)

4. Automatisation des tests intégration

- ◆ Stratégies propres à l'intégration : big-bang, "au fil de l'eau", par incréments etc.
- ◆ Intégration ascendante versus descendante. Intégration mixte
- ◆ Objets simulacres : bouchons pour simuler les fonctions appelées, mocking pour remplacer un objet. Les frameworks
- ◆ Intégration continue : mise en place de gestionnaire de configuration, constructeur de build, outils de pilotage
- ◆ Focus sur un gestionnaire de configuration logiciel
- ◆ Signalement automatique des anomalies
- ◆ Exécution automatique et cyclique des tests logiciels
- ◆ Focus sur un constructeur de build
- ◆ Focus sur un serveur d'intégration continue
- ◆ **Travaux pratiques.** Utilisation d'objets simulacres. Mise en place d'un serveur d'intégration continue (Jenkins, Hudson ou Continuum)

5. Automatisation des tests fonctionnels

- ◆ Définition du test fonctionnel, non-régression
- ◆ Le test simulant l'action des utilisateurs à partir des interfaces utilisateurs (IHM)
- ◆ Constats sur l'automatisation du test fonctionnel
- ◆ Automatisation des tests via l'IHM, via des interfaces de programmation (API)
- ◆ Chaîne d'outils, robots de test, script (API publiques)
- ◆ Gestion de l'obsolescence des tests
- ◆ **Travaux pratiques.** Automatisation de tests fonctionnels sur client lourd et sur Web
- ◆ Utilisation de robots, d'une couche logicielle publique (API). Mise en œuvre d'un outil d'automatisation (Katalon Studio avec Selenium. Ranorex ou TestComplete)

6. Automatisation des tests de performance

- ◆ Les différents types de tests système : performance, charge, stress
- ◆ Constats sur les tests de performance
- ◆ Tests de performance, définition des indicateurs
- ◆ Scripts de tests de performance. Mise en œuvre d'outils et analyse
- ◆ Tests de montée en charge, tests de stress : présentation des indicateurs clés et de l'outillage
- ◆ Gérer la base de données de tests. Gestion du contexte
- ◆ **Travaux pratiques.** Mise en place d'un test de charge

7. Synthèse

- ◆ Automatisation des tests : coûts directs, indirects
- ◆ Gains quantitatifs de l'automatisation : volumétrie, modularité ...
- ◆ Gains qualitatifs de l'automatisation : mode opératoire sauvegardé, pas d'erreur, confiance...
- ◆ Mesure du retour sur investissement