Plan de formation

TDD avancé

But de la formation

Perfectionner les pratiques de tests automatisés et de TDD afin d'améliorer la qualité et la maintenabilité de ceux-ci et ainsi être en mesure d'écrire de bons tests même en présence de cas difficiles à tester :

Description de la formation

Perfectionner les pratiques de TDD et l'art d'écrire de bons tests unitaires automatisés

Objectifs pédagogiques

1. Maîtriser les techniques avancées de tests automatisés et de TDD;
2. Savoir identifier les indices de non-qualité ou de problèmes architecturaux à partir des tests, du TDD et des Mocks (mauvaises odeurs);
3. Appliquer des techniques de conception permettant le pilotage de l'architecture à l'aide du TDD et des tests;
4. Comparer l'approche TDD Mockiste et traditionnelle;
5. Comprendre les défis et astuces concernant la maintenabilité des tests à long terme dans une équipe et une organisation;
6. Comprendre comment réaliser des tests dans différents contextes et technologies d'un logiciel d'entreprise.

Contenu

Introduction :

- rappel du TDD;
- exigences du logiciel (qualité);
- automatisation des tests;
- taxonomie des tests.

Trucs et astuces avec les tests :

- nommage;
- discussion : style BDD dans les tests unitaires;
- techniques (patrons) de tests;
- builder / objets mères.

Les « odeurs » avec le TDD et les tests (indices de qualité)

Maintenabilité d'un portefeuille de tests automatisés :

- défis et conséquences;
- tests fragiles;
- aspects humains et d'équipe.

Organisation des tests :

- suites;
- pyramide des tests;
- où se brancher (couches);
- tester via le UI ou le contourner?

Outils pour améliorer votre pratique :

- intégration continue (survol);
- métriques (dont la couverture et la dette technique);
- outils de visualisation de la qualité (outils d'analyse, détecteurs automatisés, tableau de bord de la qualité).

Architecture et tests :

- séparation des couches;
- principe d'inversion des dépendances (DIP);

TDD avancé

- délégation et stratégies;
- injection de dépendances / Service Locator pour les tests;
- agrégats (DDD).

Approche mockiste :

- introduction;
- différences avec l'approche classique;
- tests d'états versus comportements;
- développement du « haut vers le bas »;
- trucs et astuces.

Architecture émergente

Discussions sur les mocks et les doublures :

- types de doublures (stub, mocks, fakes…);
- quand / pourquoi;
- mocks d'interfaces versus de classes concrètes;
- odeurs.

Tests en situations difficiles :

- tester les classes abstraites et les implémentations;
- tester le UI (MVC/MVP, etc.);
- tester la base de données et les fichiers;
- tester les services web (WS) et le réseau;
- tester le Web (dont JavaScript, AJAX, WebApp);
- tester le matériel;
- tester la concurrence (threads et synchronisation);
- aperçu des tests pour les besoins fonctionnels (ATDD/BDD);
- aperçu des tests pour les besoins non fonctionnels (tests système, performance et sécurité);
- aperçu des tests dans un système patrimonial (legacy).

Méthodologie

Présentations interactives (45%); échanges et cas proposés par les participants (25%); démonstrations et exercices pratiques (30%) Préalable : Être familier avec un langage orienté objet. Les exercices pratiques pourront être réalisés en VB, C# ou Java avec un IDE (VS.NET, Eclipse ou IntelliJ) selon la préférence du participant.

Clientèle visée

Personne utilisant déjà le TDD ou possédant de l'expérience concrète avec les tests automatisés

Durée

2 jour(s)

Coût par participant en formation publique

0 $ (à déterminer)

ÉTS FORMATION est le leader universitaire en formation continue avec plus de 7 000 participants formés annuellement et une offre de plus de 300 différentes formations. Nos formations sont pratiques et pragmatiques et affichent un taux de satisfaction supérieur à 90 %. Consultez notre programmation complète au http://www.etsformation.ca/