ATELIER AUTOMATISATION DES TESTS LOGICIELS

Groupe: LEGRAND Etienne, SHIM Sang Min, FULSIN Julien, PLANIOL François

Table des matières

Partie 1 : Exercices de compréhension

Exercice 1 : Introduction aux tests automatisés

1) Définissez ce qu'est un test logiciel automatisé.

Un test logiciel automatisé est un test effectué à l'aide d'un logiciel spécifique qui exécute automatiquement des scripts de test sur le logiciel à tester. Ces tests sont conçus pour vérifier que le logiciel fonctionne correctement en simulant diverses actions et en comparant les résultats obtenus avec les résultats attendus.

2) Expliquez pourquoi les tests automatisés sont importants dans le cycle de développement logiciel.

Les tests automatisés sont importants dans le cycle de développement logiciel pour plusieurs raisons :

- Efficacité et rapidité : Ils permettent d'exécuter rapidement un grand nombre de tests, ce qui serait impraticable manuellement.
- Cohérence : Ils garantissent que les tests sont effectués de manière cohérente à chaque exécution, sans variation humaine.
- Détection précoce des défauts : Ils permettent de détecter les défauts tôt dans le cycle de développement, ce qui réduit le coût de correction des erreurs.
- Répétabilité : Ils facilitent la réexécution fréquente des tests après chaque modification du code, assurant ainsi la non-régression.

3) Donnez trois avantages et trois inconvénients des tests automatisés par rapport aux tests manuels.

Avantages :

- Rapidité : Les tests automatiques peuvent être exécutés beaucoup plus rapidement que les tests manuels.
- Précision : Ils éliminent le risque d'erreur humaine.
- Scalabilité : Ils peuvent facilement être étendus pour couvrir une large base de tests.

Inconvénients :

- Coût initial : Le développement et la mise en place de tests automatisés peuvent être coûteux et nécessitent un investissement initial important.
- Maintenance : Les scripts de test automatisés doivent être maintenus et mis à jour régulièrement pour refléter les changements dans le code.
- Complexité : La création de tests automatisés pour certaines applications complexes peut être difficile et exiger des compétences techniques avancées.

Exercice 2: Types de tests automatisés

1) Décrivez les différents types de tests automatisés : tests unitaires, tests d'intégration, tests fonctionnels et tests de performance.

Tests unitaires : Ces tests vérifient le bon fonctionnement de chaque unité individuelle du code (comme une fonction ou une méthode) en isolant cette unité des autres.

Tests d'intégration : Ils vérifient que différentes unités ou modules fonctionnent bien ensemble. Ils testent les interactions entre les modules intégrés pour détecter les défauts d'interface.

Tests fonctionnels : Ces tests vérifient que le système répond correctement aux exigences spécifiées en testant ses fonctionnalités comme un utilisateur final le ferait.

Tests de performance : Ils évaluent la performance du système sous différentes charges pour garantir qu'il répond aux exigences de performance, comme le temps de réponse et la capacité de gestion de la charge.

2) Pour chaque type de test, donnez un exemple concret de situation où ce type de test serait pertinent.

Tests unitaires : Lors de la création d'une nouvelle fonction de calcul, un test unitaire peut vérifier que cette fonction retourne les résultats corrects pour diverses entrées.

Tests d'intégration : Lorsque plusieurs services web doivent interagir, un test d'intégration peut vérifier que les appels d'API entre ces services fonctionnent correctement.

Tests fonctionnels : Pour une application web, un test fonctionnel peut vérifier que l'utilisateur peut s'inscrire, se connecter et utiliser les fonctionnalités principales de l'application.

Tests de performance : Lors du déploiement d'une nouvelle version d'un site e-commerce, un test de performance peut vérifier que le site peut gérer un grand nombre d'utilisateurs simultanément sans ralentir.

3) Comparez les tests manuels et les tests automatisés en termes de coût, de temps et de couverture de test.

Coût : Les tests manuels ont un coût initial plus faible mais deviennent coûteux à long terme en raison de la main-d'œuvre nécessaire. Les tests automatisés ont un coût initial plus élevé mais sont moins coûteux à long terme pour des tests répétés.

Temps : Les tests manuels prennent plus de temps car ils doivent être exécutés manuellement à chaque fois. Les tests automatisés sont beaucoup plus rapides une fois les scripts écrits.

Couverture de Test : Les tests automatisés peuvent couvrir une plus grande base de tests de manière cohérente, alors que les tests manuels sont limités par le temps et les ressources disponibles.

Exercice 3: Outils d'Automatisation

1) Quels sont les critères à considérer pour décider si un test doit être automatisé ou non?

- Fréquence d'exécution : Les tests qui doivent être exécutés fréquemment bénéficient de l'automatisation.
- Répétitivité : Les tests répétitifs et fastidieux sont de bons candidats pour l'automatisation.
- Complexité : Les tests simples à automatiser sont préférables; les tests trop complexes peuvent ne pas être rentables à automatiser.
- Stabilité : Le code testé doit être suffisamment stable pour que les tests automatisés restent valides.
- Retour sur investissement (ROI) : L'automatisation doit offrir un bon retour sur investissement en termes de gain de temps et de réduction des erreurs.

2) Expliquez l'importance de la maintenance des scripts de tests automatisés et comment les changements dans le code source peuvent affecter les tests.

La maintenance des scripts de tests automatisés est cruciale car le code source change fréquemment au cours du développement logiciel. Sans maintenance régulière, les scripts de tests peuvent devenir obsolètes et ne plus refléter correctement le comportement du logiciel, entraînant des résultats de test incorrects. Les modifications du code source peuvent introduire de nouvelles fonctionnalités ou changer des fonctionnalités existantes, nécessitant des mises à jour des scripts de tests pour s'assurer qu'ils restent pertinents et efficaces.

3) Listez cinq outils populaires pour l'automatisation des tests logiciels.

- Selenium : Principalement utilisé pour les tests d'IHM des applications web.
- JUnit : Utilisé pour les tests unitaires en Java.
- JMeter : Utilisé pour les tests de performance et de charge.
- TestNG : Un autre framework de test unitaire pour Java qui offre plus de fonctionnalités que JUnit.
- Cucumber : Utilisé pour les tests fonctionnels avec une approche BDD (Behavior Driven Development).

4) Pour chaque outil, indiquer ses principales fonctionnalités et le type de tests qu'il supporte.

- Selenium : Permet l'automatisation des navigateurs web. Supporte les tests fonctionnels et de régression des applications web. Offre des fonctionnalités pour contrôler les actions de l'utilisateur dans le navigateur.
- JUnit : Framework de test unitaire pour Java. Supporte l'écriture et l'exécution de tests unitaires. Intégré avec des outils de construction comme Maven et Gradle.
- JMeter : Utilisé pour tester la performance des applications web. Peut simuler une charge lourde sur un serveur, un groupe de serveurs, un réseau ou un objet pour tester sa force ou analyser les performances globales sous différents types de charge.
- TestNG : Framework de test inspiré de JUnit mais avec de nouvelles fonctionnalités telles que les tests parallèles, la configuration flexible des tests, et le support des tests de groupes.
- Cucumber : Utilisé pour les tests BDD, permet d'écrire des tests dans un langage naturel compréhensible par les non-développeurs. Intégré avec des outils comme Selenium pour exécuter les tests sur les navigateurs web.

5) Comparez deux outils d'automatisation en termes de facilité d'utilisation, de coût et de support communautaire.

Selenium vs JUnit :

- Facilité d'utilisation : Selenium nécessite des connaissances en scripts de navigateur et en manipulation du DOM, tandis que JUnit est plus facile à utiliser pour les développeurs Java car il est directement intégré dans l'environnement de développement.
- Coût : Les deux outils sont open source et gratuits, mais Selenium peut entraîner des coûts supplémentaires en termes de configuration et de maintenance des environnements de test de navigateur.
- Support communautaire : Les deux outils bénéficient d'une large communauté de support, avec de nombreuses ressources en ligne, des forums, et des plugins disponibles. Toutefois, Selenium a une communauté plus large en raison de sa popularité pour les tests d'IHM web.

Partie 2 : Exercices de Code

Exercice 4: Écriture de tests unitaires

CalculatriceTest.java

```java
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

public class CalculatriceTest {

    @Test
    public void testAddition() {
        Calculatrice calc = new Calculatrice();
        assertEquals(5, calc.addition(2, 3));
    }

    @Test
    public void testSoustraction() {
        Calculatrice calc = new Calculatrice();
        assertEquals(1, calc.soustraction(3, 2));
    }

    @Test
    public void testMultiplication() {
        Calculatrice calc = new Calculatrice();
        assertEquals(6, calc.multiplication(2, 3));
    }

    @Test
    public void testDivision() {
        Calculatrice calc = new Calculatrice();
        assertEquals(2, calc.division(6, 3));
    }

    @Test
    public void testDivisionParZero() {
        Calculatrice calc = new Calculatrice();
        assertThrows(ArithmeticException.class, () -> {
            calc.division(1, 0);
        });
    }
}

```

Calculatrice.java

```java
public class Calculatrice {

  public int addition(int a, int b) {
    return a + b;
  }

  public int soustraction(int a, int b) {
    return a - b;
  }

  public int multiplication(int a, int b) {
    return a * b;
  }

  public int division(int a, int b) {
    if (b == 0) {
      throw new ArithmeticException("Division par zéro !");
    }
    return a / b;
  }
}
```