UTOPIOS
TRANSFORMATION DIGITALE INTELLIGENTE

JAVA CLEAN CODE Partie 4

Développement BDD

- Les paradigmes du BDD.
- Framework java pour le BDD.
- Les bonnes pratiques du BDD.

Développement BDD

Les paradigmes du BDD

- Le BDD est une pratique de développement de logiciel qui met l'accent sur la collaboration entre les différentes parties prenantes d'un projet (comme les développeurs, les testeurs, les responsables produit, etc.) et l'explication du comportement du système en termes compréhensibles par tous. Il est souvent utilisé dans le développement Agile
- **Communication et collaboration** : Le BDD insiste sur la nécessité d'une collaboration étroite entre toutes les parties prenantes du projet. Il met l'accent sur le "partage des connaissances", en veillant à ce que tout le monde comprenne le comportement désiré du système.
- **Développement basé sur le comportement** : Comme son nom l'indique, le BDD met l'accent sur le comportement du système plutôt que sur les détails techniques de son implémentation. Il s'agit de décrire ce que le système doit faire de manière compréhensible par tous, et non de se concentrer sur la manière dont ces fonctionnalités seront mises en œuvre.
- **Spécification exécutable** : Le BDD utilise un langage naturel pour décrire le comportement du système. Ces descriptions sont ensuite utilisées comme spécifications exécutables, qui peuvent être exécutées comme tests. Cela signifie que les spécifications servent à la fois de documentation et de vérification du système.

Développement BDD

Les paradigmes du BDD

- **Tests orientés comportement** : Le BDD utilise un format spécifique pour les tests, connu sous le nom de "Given-When-Then" (Etant donné - Quand - Alors). Cela décrit le contexte (Given), l'action qui est effectuée (When), et le résultat attendu (Then). Cela aide à structurer les tests de manière à ce qu'ils reflètent le comportement désiré du système.
- **Développement itératif** : Comme d'autres pratiques Agile, le BDD suit une approche itérative du développement. Il s'agit de construire progressivement le système, en ajoutant un comportement à la fois, et en vérifiant constamment que le système se comporte comme prévu.

Développement BDD

Framework java pour le BDD

Cucumber : Cucumber est l'un des frameworks les plus populaires pour le BDD. Il vous permet d'écrire des scénarios de tests en langage naturel qui peuvent être exécutés comme des tests automatisés. Cucumber dispose d'une intégration étroite avec Spring, ce qui vous permet d'utiliser des fonctionnalités comme l'injection de dépendances dans vos tests.

Développement BDD

Les bonnes pratiques du BDD

- **Définissez clairement les comportements** : Vos spécifications devraient décrire clairement le comportement attendu du système. Elles ne devraient pas se concentrer sur les détails techniques de l'implémentation, mais plutôt sur ce que l'utilisateur peut faire et ce qu'il peut s'attendre à voir.
- **Utilisez le format Given-When-Then** : Ce format est un excellent moyen de structurer vos scénarios de manière claire et compréhensible. Il vous aide à vous concentrer sur le contexte (Given), l'action (When) et le résultat (Then).
- **Gardez vos scénarios courts et concentrés** : Chaque scénario doit tester une seule fonctionnalité ou comportement. S'il y a trop de choses dans un seul scénario, il devient difficile à comprendre et à maintenir.
- **Automatisez vos scénarios** : Les scénarios BDD doivent être automatisés pour pouvoir les exécuter régulièrement. Cela vous permet de vérifier rapidement que votre système se comporte toujours comme prévu, même après des modifications.
- **Revoyez et affinez vos scénarios régulièrement** : Comme tout autre aspect de votre système, vos scénarios BDD devraient être revus et affinés régulièrement. Cela vous aide à maintenir leur pertinence et leur utilité.