TP Noté : Video Store Kata

Auteur	Version	Date
Thomas Domingues	1.0.0	18/12/2020

Objectifs

- Refactoriser une application legacy en respectant les bonnes pratiques de code
- Documenter le processus de refactorisation appliqué

Travail à faire

Ce travail est à faire **individuellement**. Toute suspicion de triche occasionnera la note de **0/20** à tous les fraudeurs. Attention, je n'hésiterai pas à utiliser un logiciel de détection de plagiat !

Ce TP est à rendre pour le **vendredi 15 Janvier 2021 23h59**.

Description sommaire de l'application

"videostore" est une application de gestion utilisée par une entreprise louant des DVD. Elle permet au gérant :

- D'ajouter des films à louer ;
- D'ajouter de nouveaux clients ;
- D'enregistrer la réservation d'un film par un client donné ;
- D'obtenir en un clin œil des informations sur un client en particulier (points de fidélité accumulés, films loués, etc).

Consignes

Vous devez refactoriser l'application videostore pour la rendre compréhensible, maintenable, flexible et évolutive. Cette étape de nettoyage se fera en deux parties :

- Refactorisation du code
- Documentation des étapes réalisées

⚠ Veillez à démarrer depuis le tag *original* du dépôt unclebob/videostore, et non depuis la branche *master* ou depuis le tag *final*.

Refactorisation du code

Pour réussir cette partie, vous devez :

- Nettoyer le code en utilisant des techniques de refactorisation (n'hésitez pas à vous appuyer sur l'excellent site de Refactoring Guru)
- Corriger et/ou ajouter des tests unitaires pour vérifier le bon fonctionnement du programme, comme vu lors de la correction du TP Gilded Rose.
- Utiliser Apache Maven comme outil de build pour exécuter les tests, installer les dépendances et éventuellement exécuter le programme.

⚠ Veillez à ce que l'application fonctionne **exactement** à l'identique (à l'exception des tests unitaires et du main) de la version originale, ou vous perdrez des points.

Documentation du processus de refactoring

Pour réussir cette partie, vous devez documenter dans le fichier `README.md` toutes les étapes qui vous ont permis de refactoriser l'application.

Le `README.md` devra contenir différentes choses :

Auteur

Votre nom et prénom, ainsi que votre classe.

Statistiques

Cette partie, au tout début du `README.md`, devra résumer le nombre de code smells détectés ainsi que le nombre de techniques de refactoring appliquées.

Descriptif

Cette partie contiendra un tableau (ou une liste) avec, pour chaque ligne :

- Le code smell détecté
- La ou les technique(s) de refactoring appliquées
- Le ou les bénéfice(s) apporté par ce correctif

Conseils

- Lisez bien les tests unitaires présents dans le code initial pour comprendre le fonctionnement de l'application.
- N'hésitez pas à faire une classe contenant une méthode *main* pour tester les différentes classes de l'application.
- Vous devrez déterminer le fonctionnement détaillé de l'application par vous-même. Testez, testez et retestez !
- Observez le fonctionnement de la méthode `statement` de la classe `Customer`.
- Ne trichez pas.

Barème

- Refactorisation du code : 12 points
- Documentation du processus de refactoring : 8 points

Critères d'évaluation

Refactorisation du code

- Qualité du code rendu
- Fonctionnement de l'application identique à l'original
- Bonne exécution des tests unitaires
- Couverture de tests (plus il y a de tests couvrant les cas limites, mieux c'est)
- Utilisation de l'outil de build Apache Maven

Documentation du processus de refactoring

- Pertinence de l'analyse des code smells
- Pertinence du choix des techniques de refactoring
- Qualité de la documentation fournie