03/12/2020

Conception et programmation objet avancées

TD4



Michael McCarthy, Baptiste Herr et Samuel Husson

DUT Informatique | université cÔte d’azur

Sommaire

[Vérifications principales 2](#_Toc57579335)

[Diagramme de classe 3](#_Toc57579336)

[Couverture des tests 4](#_Toc57579337)

[SonarLint 5](#_Toc57579338)

[Compte rendu 6](#_Toc57579339)

[1.1. Avez-vous bien respecté les exigences de l’ingénieur ? 6](#_Toc57579340)

[1.2. Couverture de tests : 6](#_Toc57579341)

[1.2.1. Quels sont les principaux points faibles de votre couverture de test? 6](#_Toc57579342)

[1.2.2. Quels points amélioreriez-vous si vous en aviez le temps? 6](#_Toc57579343)

[1.3. Expliquez comment vous avez cherché à respecter le principe ouvert/fermé, par exemple expliquez l'utilisation des interfaces, etc. 6](#_Toc57579344)

[1.4. Si vous pensez que votre code respecte d'autres principes expliquez. 6](#_Toc57579345)

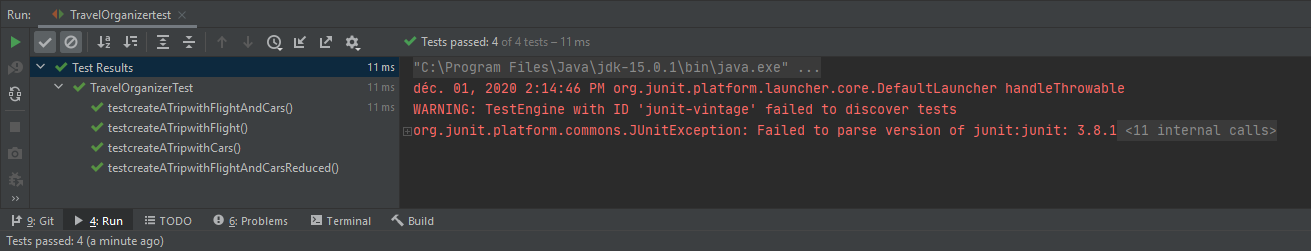
[1.5. Commentez le diagnostic de SonarLint. 6](#_Toc57579346)

[Instructions de remises 7](#_Toc57579347)

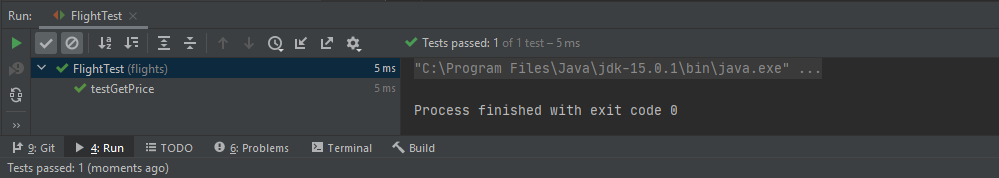
# Vérifications principales

1. Tous nos codes compilent.
2. Tous nos tests passent. Nous avons remarqué des warnings à propos de problèmes de versions JUnit. Nous pensons que c’est dû à certain import qui sont formatés pour des anciennes versions de JUnit alors que nous utilisons une version plus récente dans nos Path. En fonction de nos machines chacun avait des warnings différents c’est pour cela que nous avons préféré ne rien toucher. Voici nos résultats :

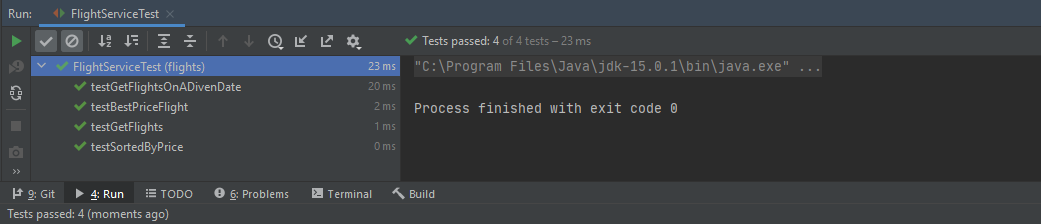
TravelOrganizerTest :



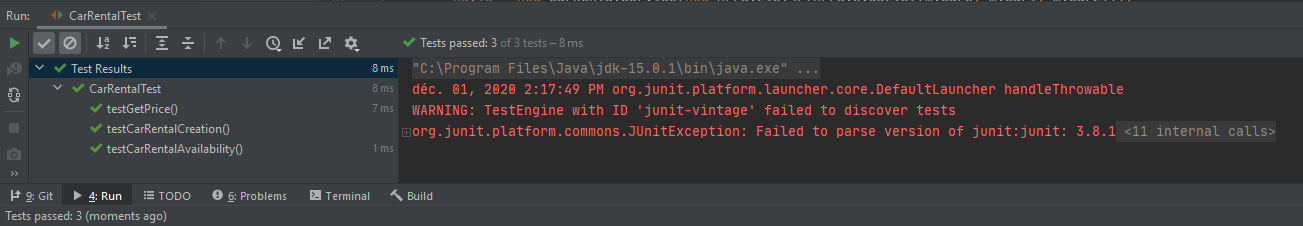
FlightTest :



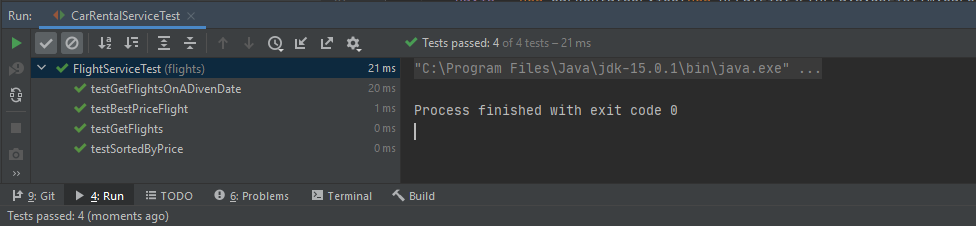
FlightServiceTest :



CarRentalTest :



CarRentalServiceTest :



1. Nous avons implémenté l'organisateur de voyage.
2. Nous avons également implémenté le service de réservation de nuits d’hôtels qui nous sert en cas de réserver des hôtels pour tout voyage durant plus d’un jour.

Diagramme de classe

Couverture des tests

SonarLint

Compte rendu

## Avez-vous bien respecté les exigences de l’ingénieur ?

Oui, après avoir suivi la totalité des consignes, notre diagramme final est plus complet que je diagramme de classe fourni dans le sujet. Nous pensons donc que les exigences de l’ingénieur ont bien toutes été respectées.

## Couverture de tests :

### Quels sont les principaux points faibles de votre couverture de test?

Tous les tests fournis avec le sujet sont passés comme montré en page 2. Néanmoins nous sommes conscients qu’il manque des tests en rapport avec certaines méthodes implémentées.

### Quels points amélioreriez-vous si vous en aviez le temps?

Si nous avions le temps, nous essaierions d’ajouter dans nos tests, la couverture de ces différentes méthodes.

## Expliquez comment vous avez cherché à respecter le principe ouvert/fermé, par exemple expliquez l'utilisation des interfaces, etc.

Pour respecter le OCP (Open/Closed Principle), nous avons ouvert nos classes à l’extension mais pas à la modification. La grande majorité de nos attributs sont déclarés en « private » et les autres en « protected ». Nous avons donc utilisé les principes d’héritages et d’interfaces pour coller au mieux aux exigences du OCP. Par exemple notre classe CarRentalService qui est une extension de Sercive4PI et qui implémente notre interface Service.

## Si vous pensez que votre code respecte d'autres principes expliquez.

Notre code respecte d'autres principes comme le SRP car certaines classes n'ont qu'une seule responsabilité comme la classe Cars qui sert uniquement à manipuler des voitures par exemple et l'ISP car nous utilisons plusieurs interfaces plutôt qu'une seule globale.

## Commentez le diagnostic de SonarLint.

Instructions de remises  
Vous rendez

1. Un document pdf contenant :
   1. Comme d'habitude soignez la 1e page avec les informations le groupe et le nom du ou des étudiants.
      1. Préciser sur la première page si :
         1. [] vos codes compilent
         2. [] les tests passent
         3. [] vous avez implémenté l'organisateur de voyage
         4. [] vous avez implémenté le service de réservation de nuits d'hôtels
   2. Un diagramme de classe au format pdf dans lequel apparait toute vos classes dans leur version finale
   3. La couverture de tests (pas de souci si elle n'est pas optimale)
   4. Le diagnostic SonarLint.
   5. La réponse aux questions suivantes :
      1. Avez-vous bien respecté les exigences de l'ingénieur?
         1. Expliquez-vous, en une ou deux lignes. (e.g. oui, j'ai exactement le même diagramme de classes mais...; oui, mais j'ai ajouté...; non, ....)
      2. Couverture de tests :
         1. Quels sont les principaux points faibles de votre couverture de test?
         2. Quels points amélioreriez-vous si vous en aviez le temps?
      3. Expliquez comment vous avez cherché à respecter le principe ouvert/fermé, par exemple expliquez l'utilisation des interfaces, etc.
      4. Si vous pensez que votre code respecte d'autres principes expliquez.
      5. Commentez le diagnostic de Sonarlint.
2. Vos codes sous la forme d'un .zip contenant exclusivement
   1. La dernière version des codes sources et des tests.
   2. Et pas de binaires ou d'anciennes versions.

**Le rendu se fait en binôme ou en trinôme**

* UN des étudiants rend le devoir qui contient le nom de son ou ses camarades,
* L’autre étudiant rend simplement le devoir par un message précisant le nom de son camarade qui a rendu le devoir pour lui.

Si cela n'est pas fait (a) aucun recours possible en cas d'absence de rendu (b) un malus est appliqué pour non-respect de la consigne à l'étudiant qui n'a rien rendu.