Agilité

La rencontre du Loto et de la Course et boum… ça fait gagner de l’argent.  
Version: 1.1

11 – 02 – 2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

Sommaire

[Sommaire 2](#_Toc806308)

[Historique du document 3](#_Toc806309)

[1. Objectif & Contexte 4](#_Toc806310)

[1.1 Objectif 4](#_Toc806311)

[1.2 Contexte 4](#_Toc806312)

[2. Implémentation des Designs Patterns 4](#_Toc806313)

[2.1 Design Pattern Observer 4](#_Toc806314)

[2.2 Design Pattern AbstractFactory 4](#_Toc806315)

[3. Mini tutoriel 5](#_Toc806316)

[4. Tests JUnit 7](#_Toc806317)

[5. Cucumber 9](#_Toc806318)

[6. Annexe 11](#_Toc806319)

[6.1 Création d’un projet Eclipse 11](#_Toc806320)

[6.2 Tests JUnit 14](#_Toc806321)

[6.3 Cucumber via Eclipse 15](#_Toc806322)

[6.3.1 Installation de Cucumber 15](#_Toc806323)

[6.3.2 Configuration de Cucumber pour exécution des tests 16](#_Toc806324)

[6.3.3 Création des fichiers pour description des Users Story 17](#_Toc806325)

Historique du document

|  |
| --- |
| Informations générales |

|  |  |
| --- | --- |
| Rédacteur(s) : | HADDAD Abderraouf  AISSAT Yacine  TIOUCHICHINE Lounes  ANDRIAMBELOMAHERY Sam Michael |
| Valideur(s) : | ZAMFIROIU MICHEL |
|  |  |

|  |  |  |
| --- | --- | --- |
| Historique des mises à jour | | |
| Version | Date | **Description** |
| 1.1 | 11/01/2019 | Création du document |
| Documents de référence | | |
|  | | |

1. Objectif & Contexte
   1. Objectif

Ce document a pour objectif de décrire les étapes nécessaires pour l’apprentissage par l’expérimentation des concepts orientés objet comme la classe et l’objet, à l’aide d’un exemple évolutif original, issu de notre imagination.

* 1. Contexte

A une époque lointaine, où la Terre n’était composé que d’un seul continent, le monde était vaste et des lieux improbables existaient, citons par exemple Onizuka, Uranus, Numidia et l’incontournable Madina, des places mythiques où le lendemain de chaque pleine lune se déroulaient des courses mythiques. Durant ces courses, de nombreux paris étaient organisés et les joueurs tentaient de gagner le plus d’or possible en pariant sur le classement des courses.

« Uhtred le jeune », fils d’Uhtred de Madina, était un de ces parieurs. Paresseux, il ne s’intéressait que peu aux coureurs, à leurs capacités et à leurs résultats. Il n’avait que faire des favoris, des coureurs blessés ou d’autres caractéristiques. Sa manière de jeu était assez singulière. Il classait simplement les joueurs de manière aléatoire grâce à leur numéro de dossard, et à la fin de la course il comparait son classement avec le résultat pour savoir si oui ou non il avait gagné de l’or.

L’histoire d’Uhtred le jeune a inspiré, des siècles plus tard, de nombreux jeux, notamment le jeu du Loto.

Afin de retracer l’histoire de ce valeureux parieur, nous avons décidé d’unir les forces de la classe fétiche Loto et de la classe fétiche Course afin de recréer la méthode de pari d’Uhtred.

L’objectif est simple. 6 joueurs avec un numéro de dossard participent à une course. Le parieur va alors entrer ces 6 numéros de manière aléatoire afin d’avoir le résultat de sa course. Il aura la possibilité notamment de connaître le gain potentiel qu’il peut faire avec ce classement aléatoire, mais il pourra surtout savoir si oui ou non il a gagné à l’issue de la course.

1. Implémentation des Designs Patterns
   1. Design Pattern Observer
   2. Design Pattern AbstractFactory
2. Mini tutoriel

Dans cette partie, nous allons tenter d’expliquer de la manière la plus simple possible à l’utilisateur comment s’approprier notre code et surtout comment le faire fonctionner.

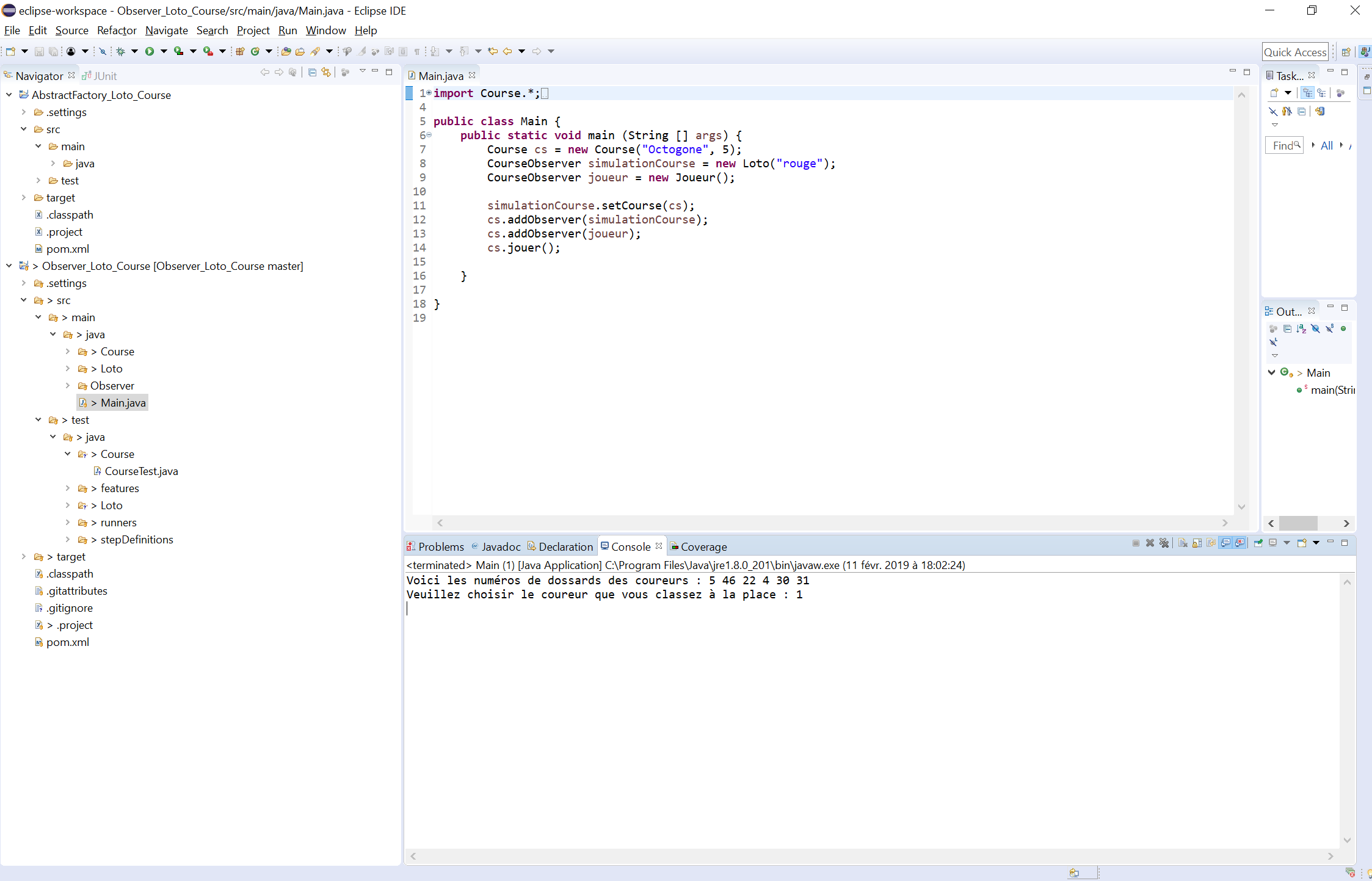
L’unique prérequis est d’avoir un IDE fonctionnel (Eclipse de préférence) sur son poste.

Pour commencer, l’utilisateur doit :

1. Récupérer le code via le lien Github suivant :

https://github.com/yacineaissat/Agilite\_Course\_Loto\_AAHT.git

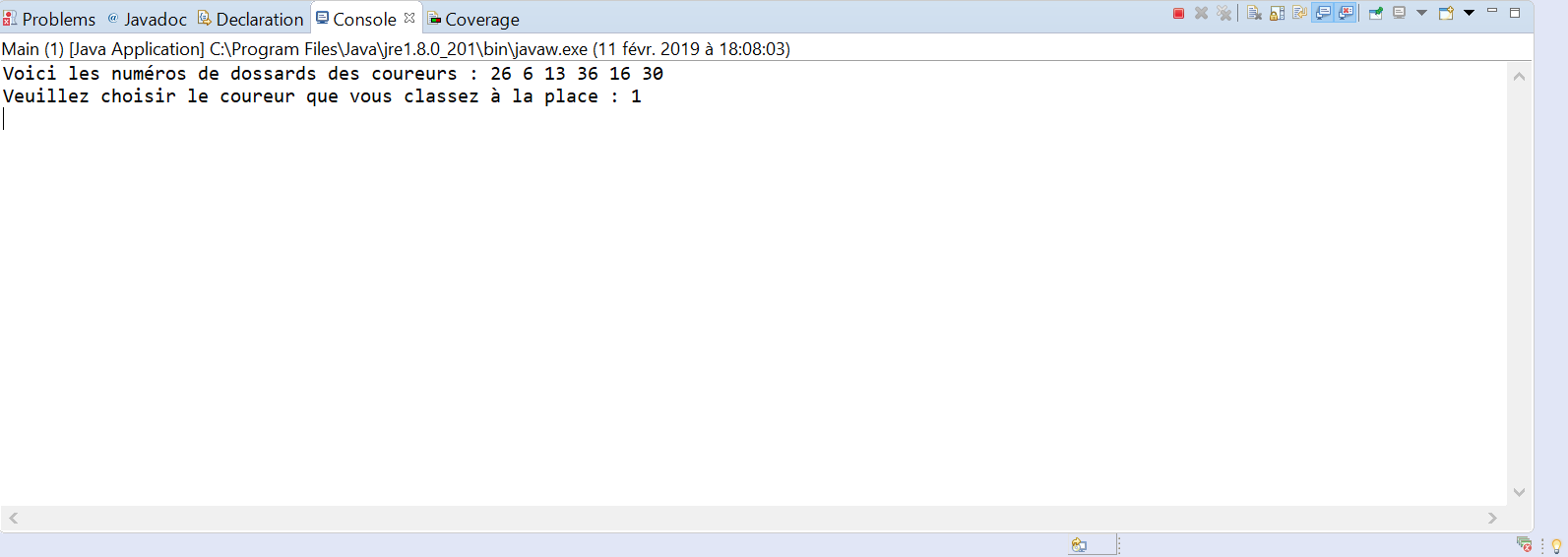
1. Importer le projet sur son environnement (Eclipse de préférence)



1. Une fois que les projets sont finalement ouverts, on va tester les deux Designs Patterns de manière distincte.

Tout d’abord, le pattern Observer.

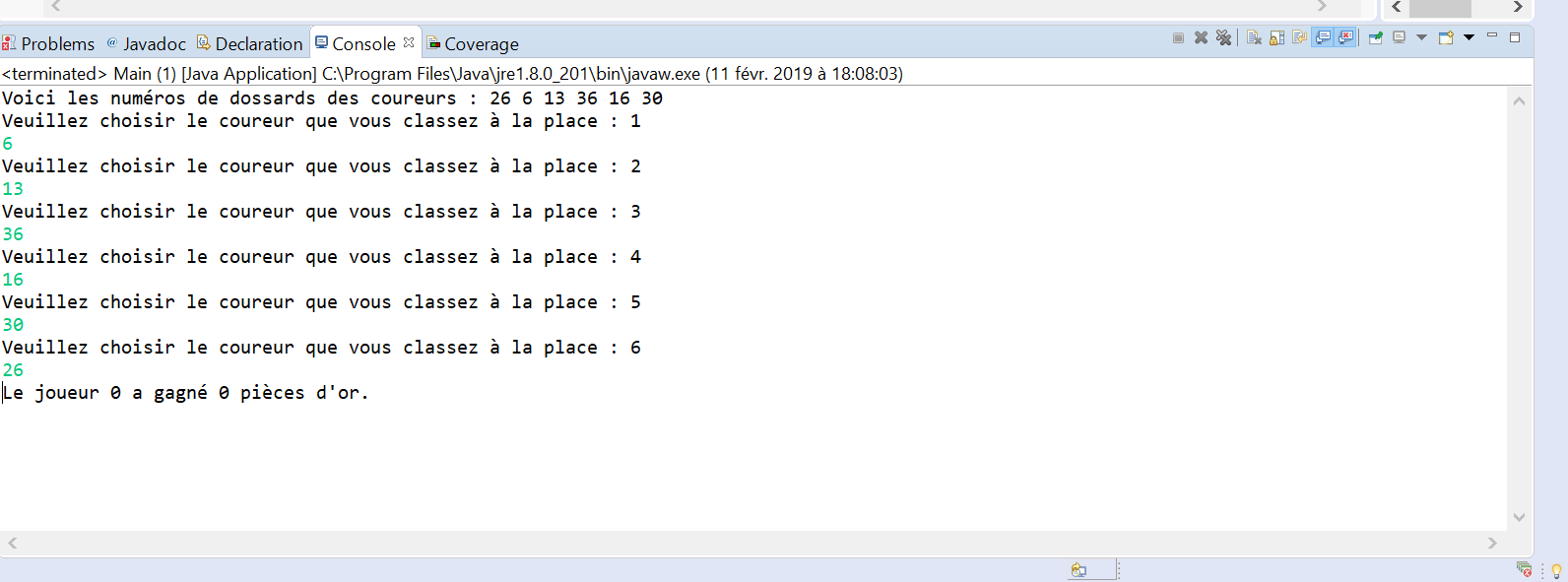
On lance la classe Main.java



On va alors tenter de deviner l’ordre d’arrivée des coureurs.

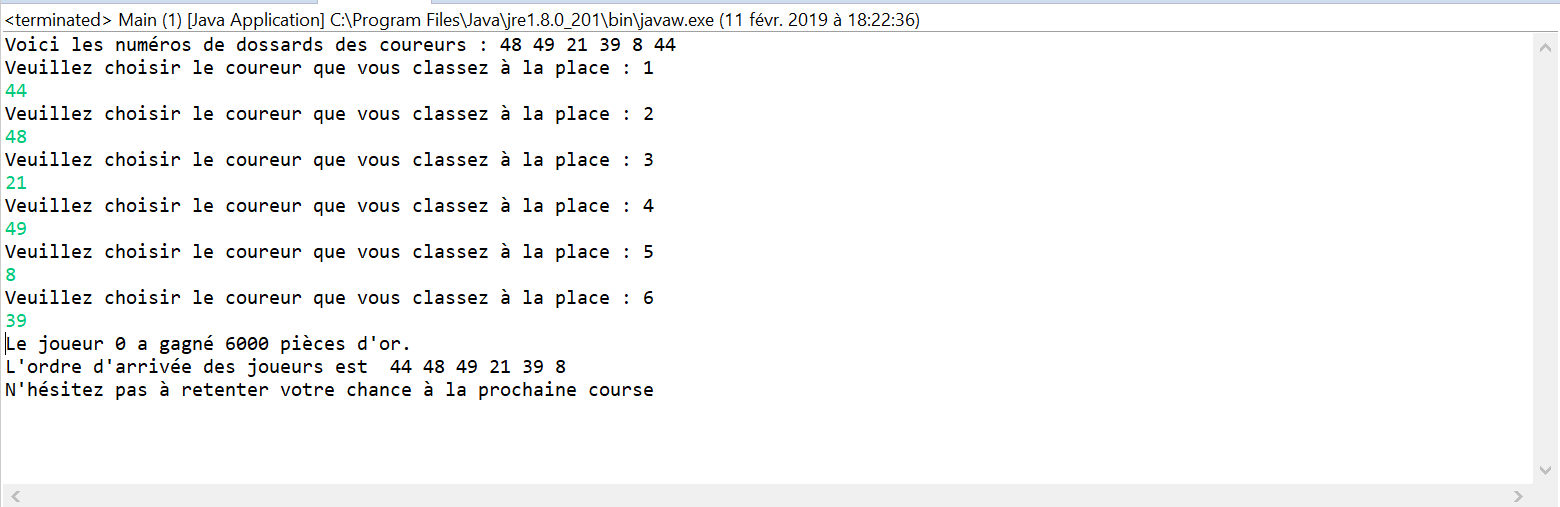
Pour cela, on va entrer un à un l’ordre d’arrivée des joueurs.

Et là… notre gain s’affiche…



Dans notre cas, on a tout perdu !! Nous finissons pauvres et endettés.

Réessayons une deuxième fois… pour la course suivante.



YESSSS ! On a trouvé 2 bonnes positions ! On termine riches et on se met à la retraite pour le restant de nos jours 😊😊😊

Trêve de plaisanterie…

A vous de tenter votre chance maintenant !

On a pu tester le code du pattern Observer, maintenant, testons le code du pattern AbstractFactory.

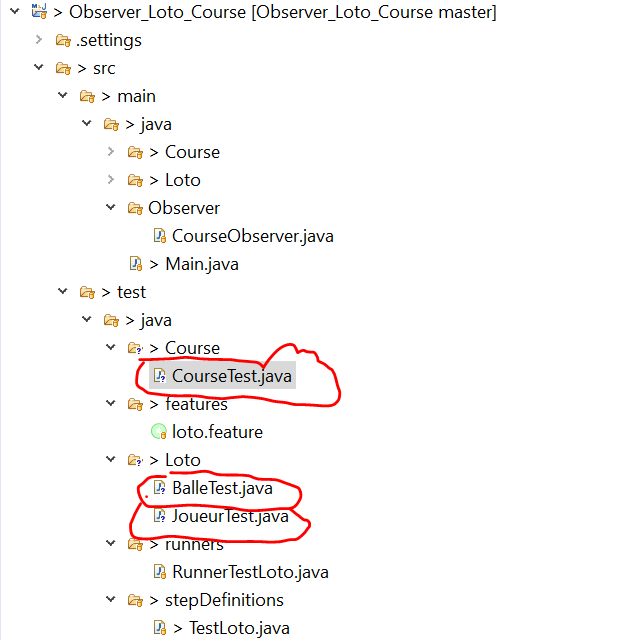
PATTERN ABSTRACTFACTORY

Maintenant, passons à la partie la plus intéressante qui est les tests JUnit et les tests Cucumber… accrochez-vous

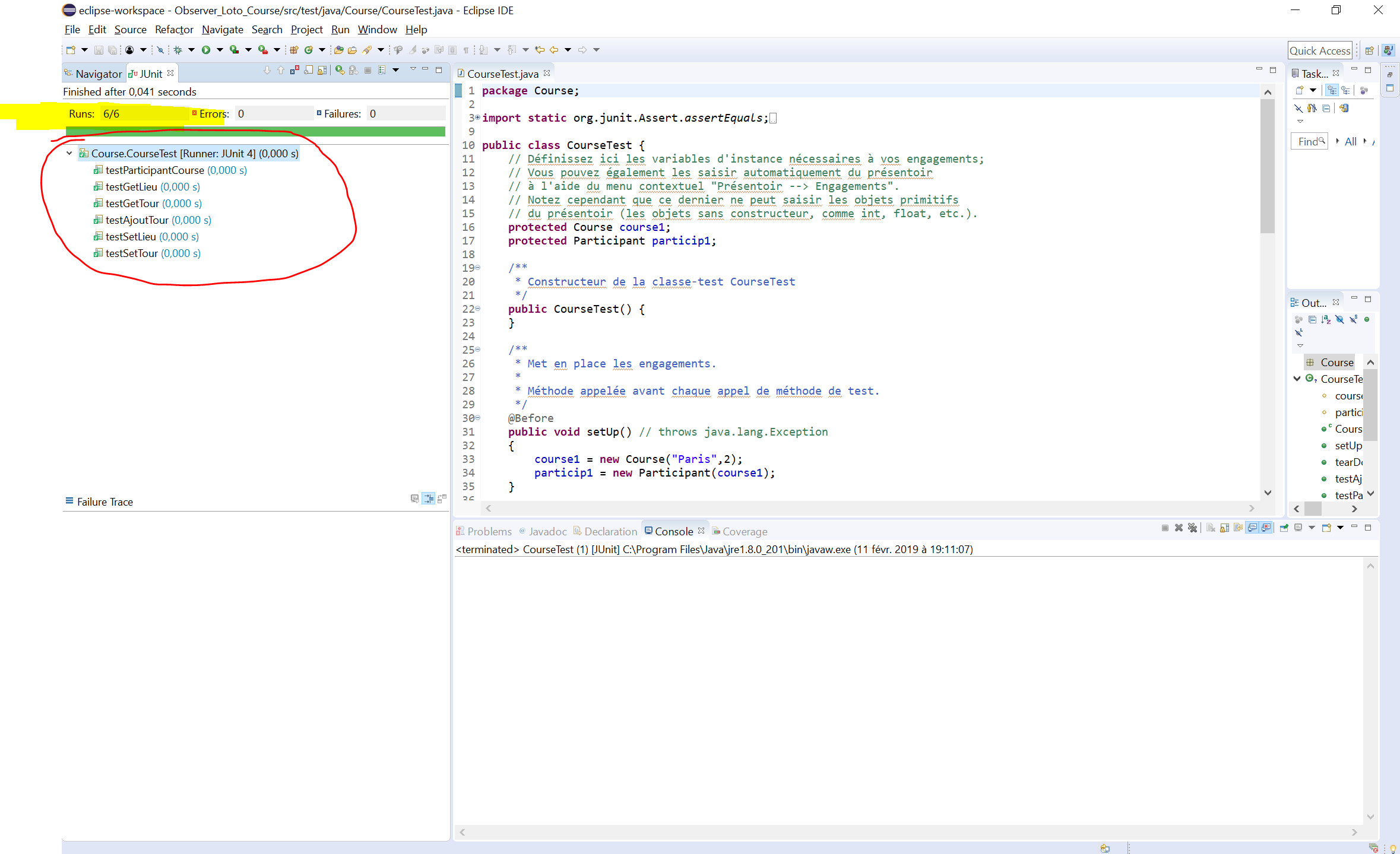
1. Tests JUnit

Afin d’exécuter les tests JUnit, il y a 3 classes à exécuter.

Vous les trouverez ci-dessous entourées en rouge.

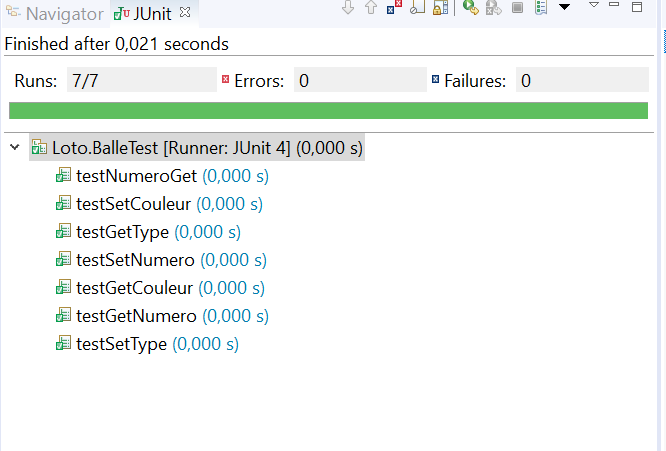


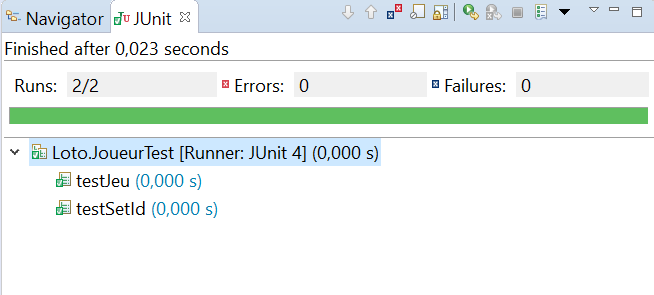
Exemple d’exécution de la classe : CourseTest.java



On voit à gauche que l’exécution s’est extrêmement bien passée, tout va bien dans le meilleur des mondes, on a la barre verte pour les 6 tests.

Exécution des deux autres classes de tests.

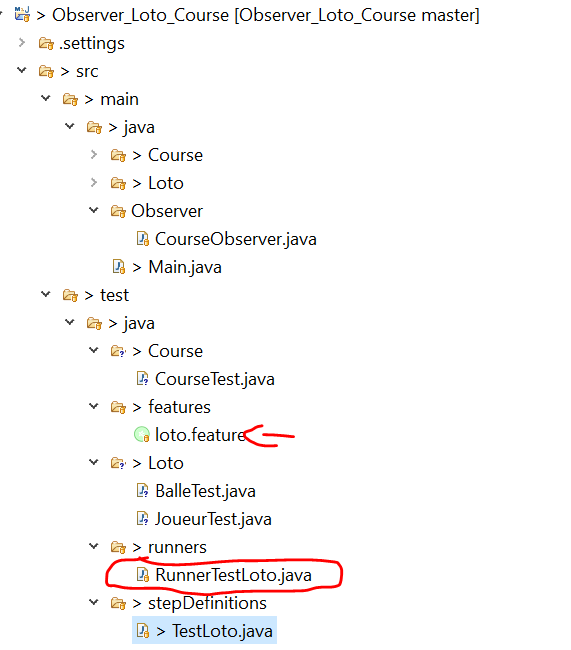




Les exécutions se déroulent bien pour chacun des classes… à vous maintenant de faire le test. Les tests sont identiques pour les deux patterns, cela n’a donc aucun intérêt de les exécuter pour les deux.

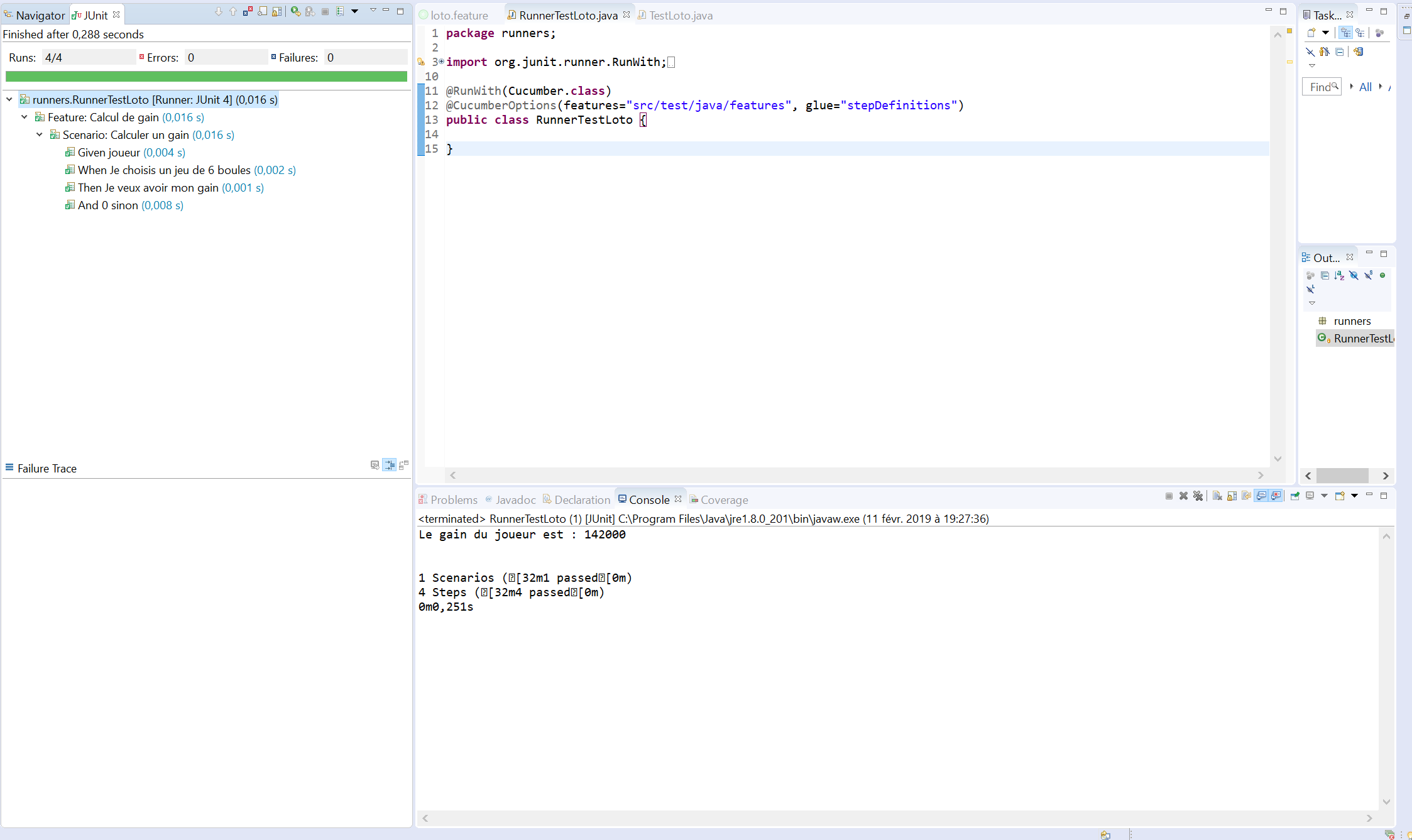
1. Cucumber

Afin d’exécuter les scenarii Cucumber, il va falloir éxécuter la classe RunnerTestLoto.java comme (celle entourée sur la capture suivante) :



Petit détail : le fichier loto.feature (celui suivi d’une flèche rouge) contient le scénario à exécuter.

Voici le résultat de l’exécution :



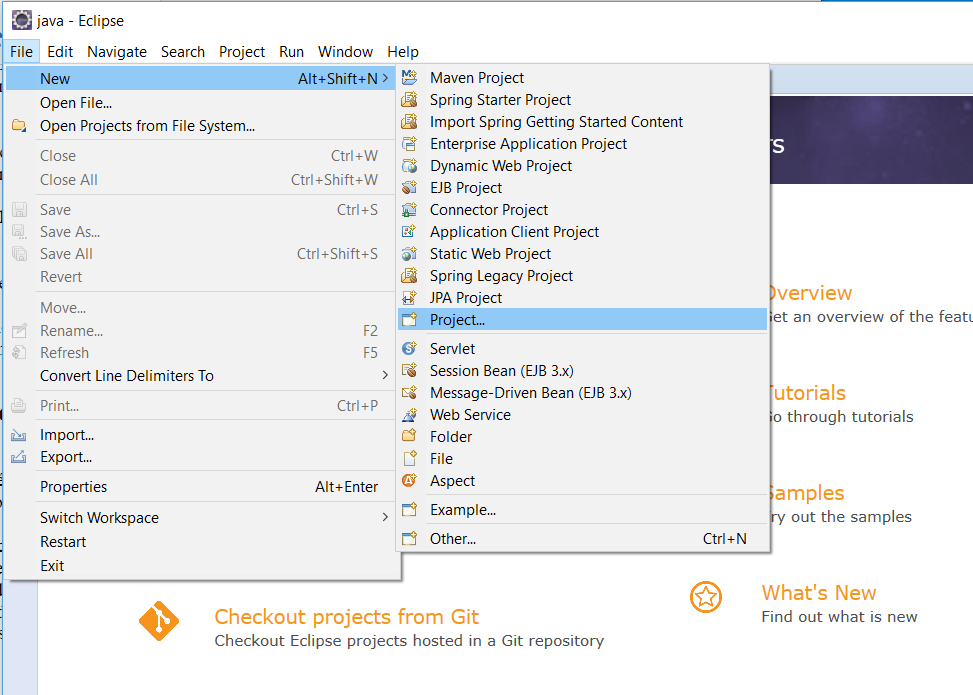
TADAM ! Encore des barres vertes qui signifient que les tests se sont bien exécutés…

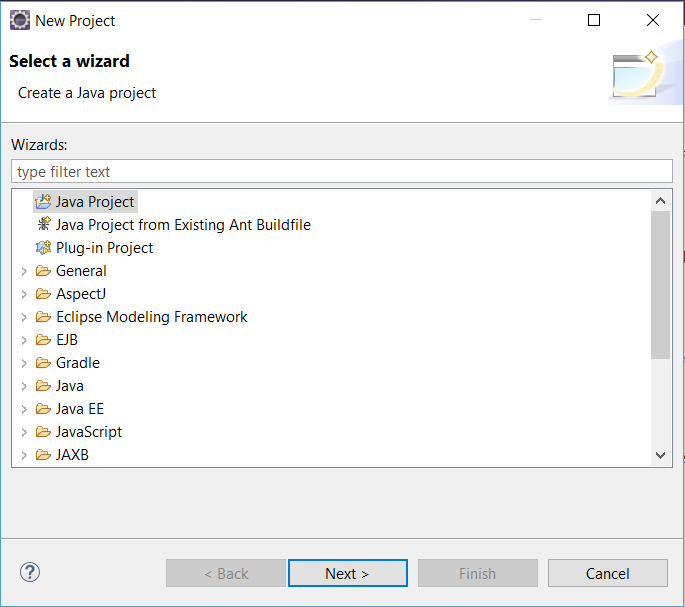
Encore une fois, nous vous laissons exécuter par vous-même les tests si l’envie vous prend… 😊

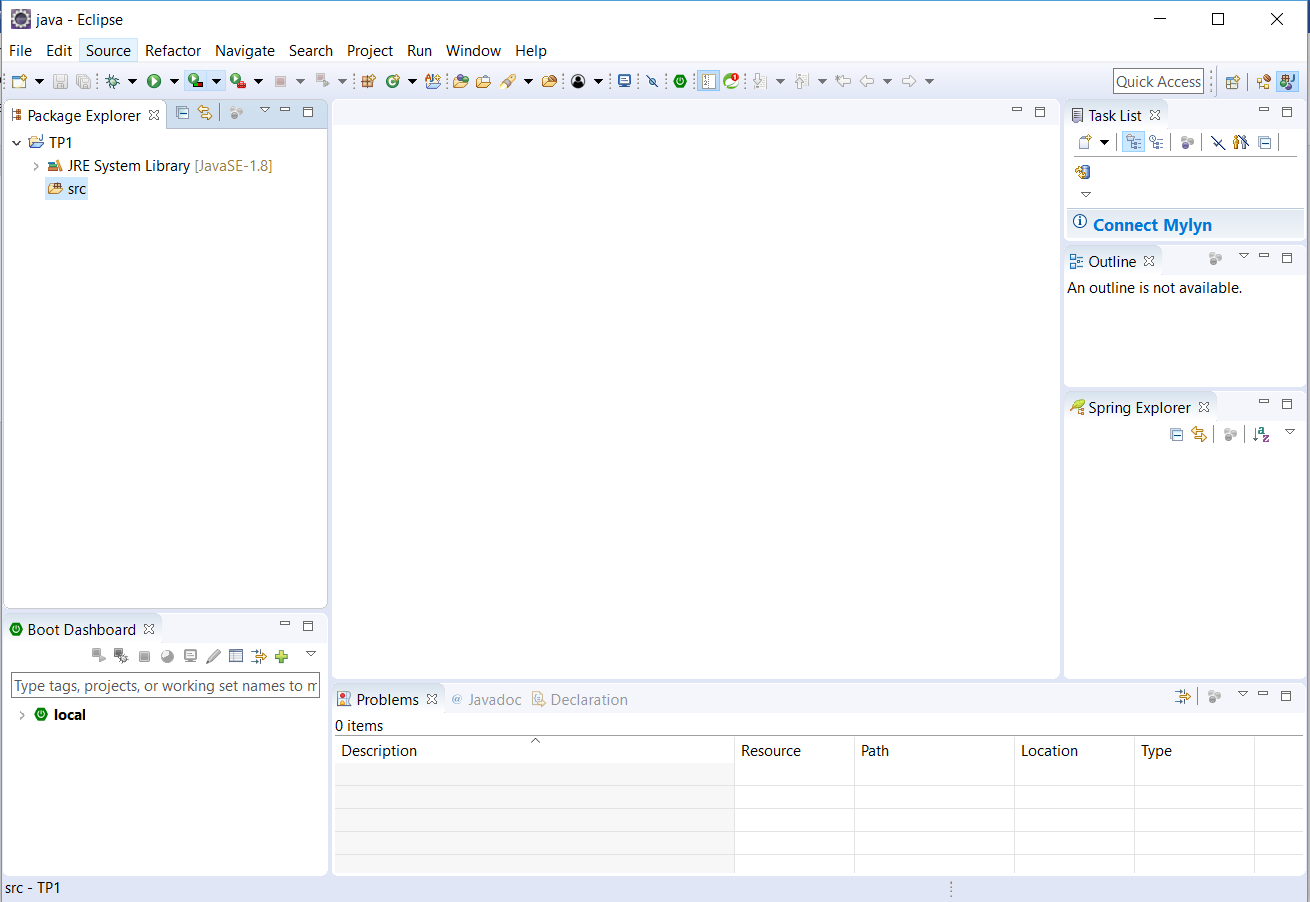
1. Annexe

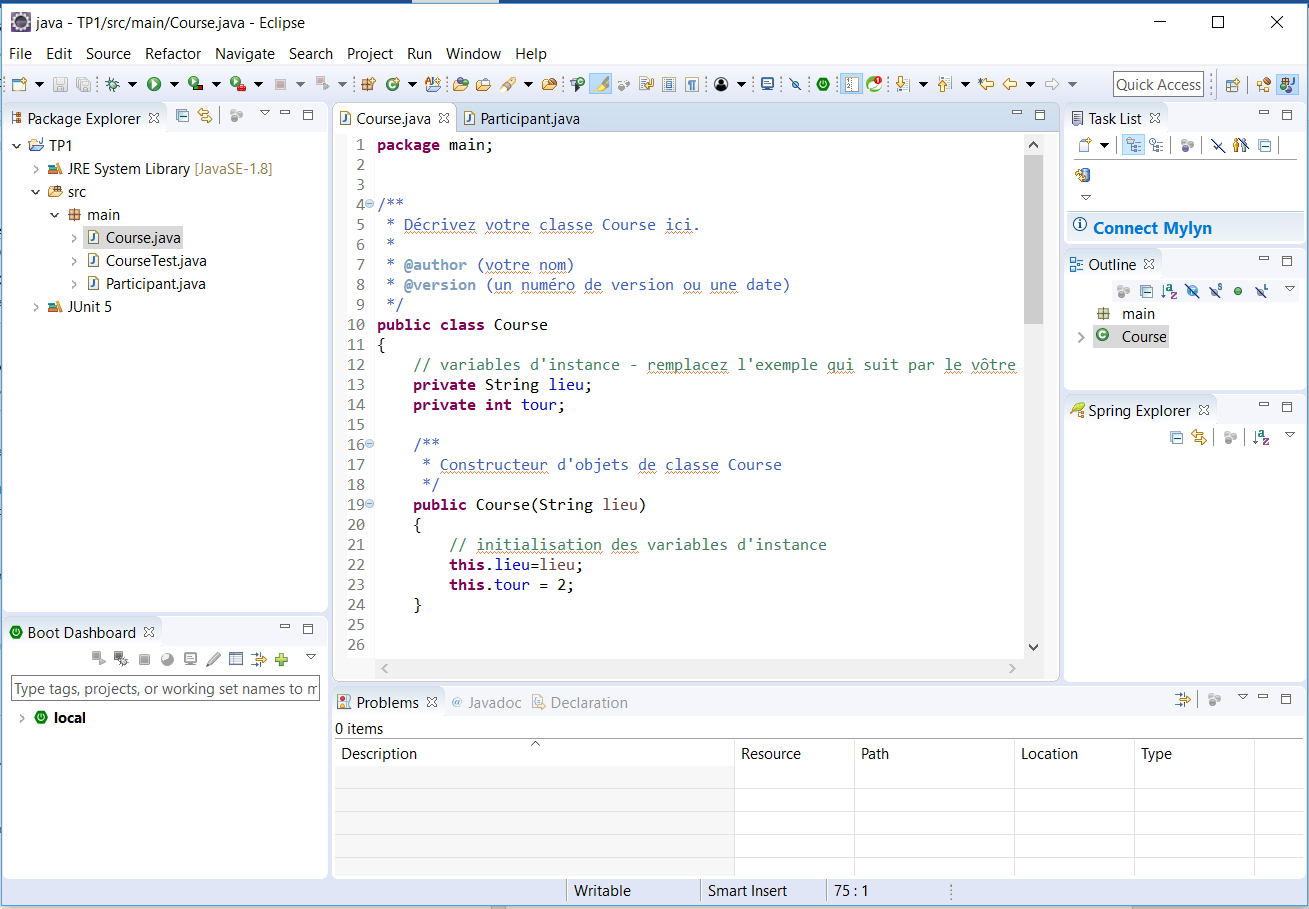
En annexe, vous trouverez quelques trucs et astuces qui nous ont beaucoup servi tout au long de la réalisation de notre projet. Cela est une sorte de mini

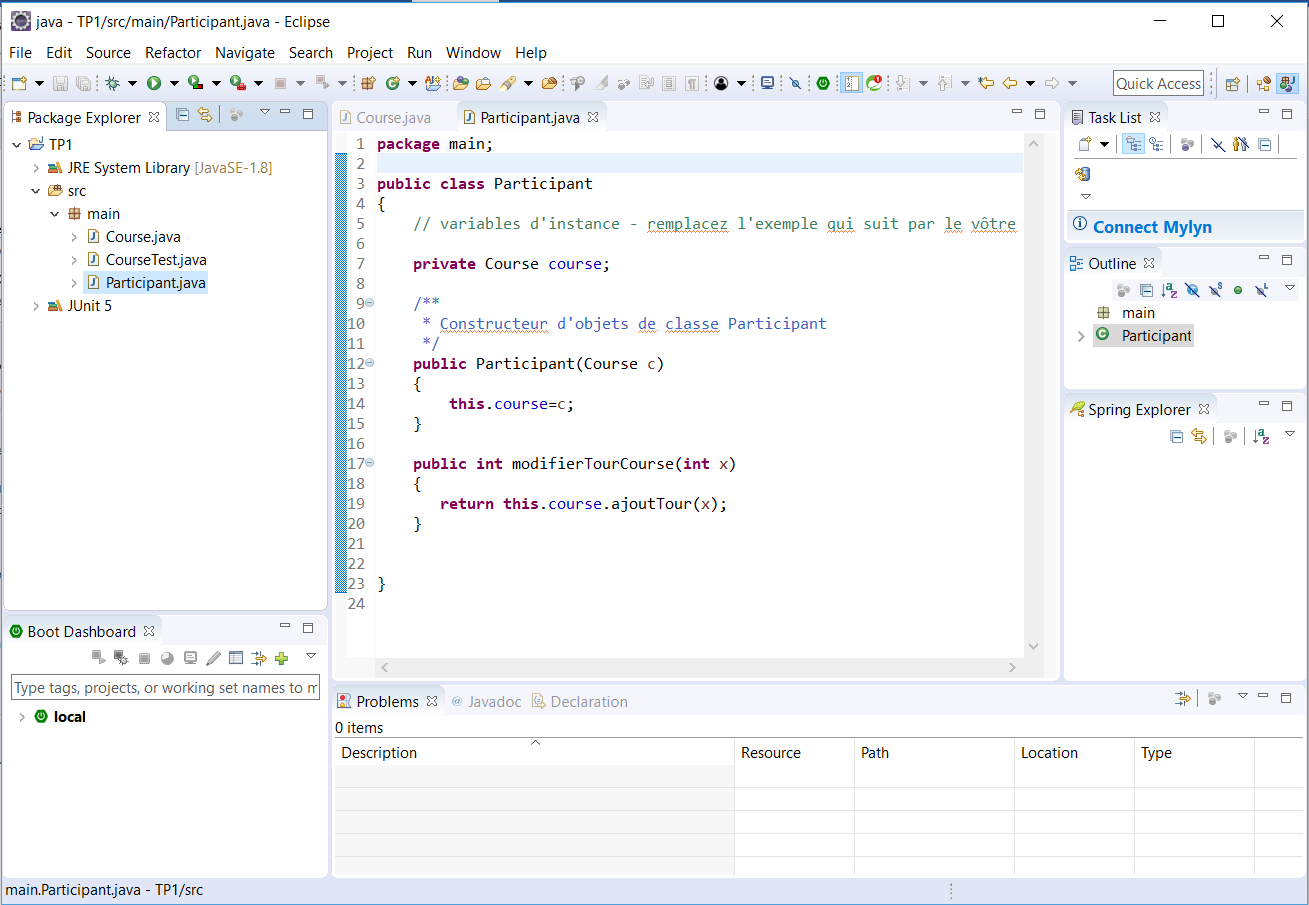
* 1. Création d’un projet Eclipse

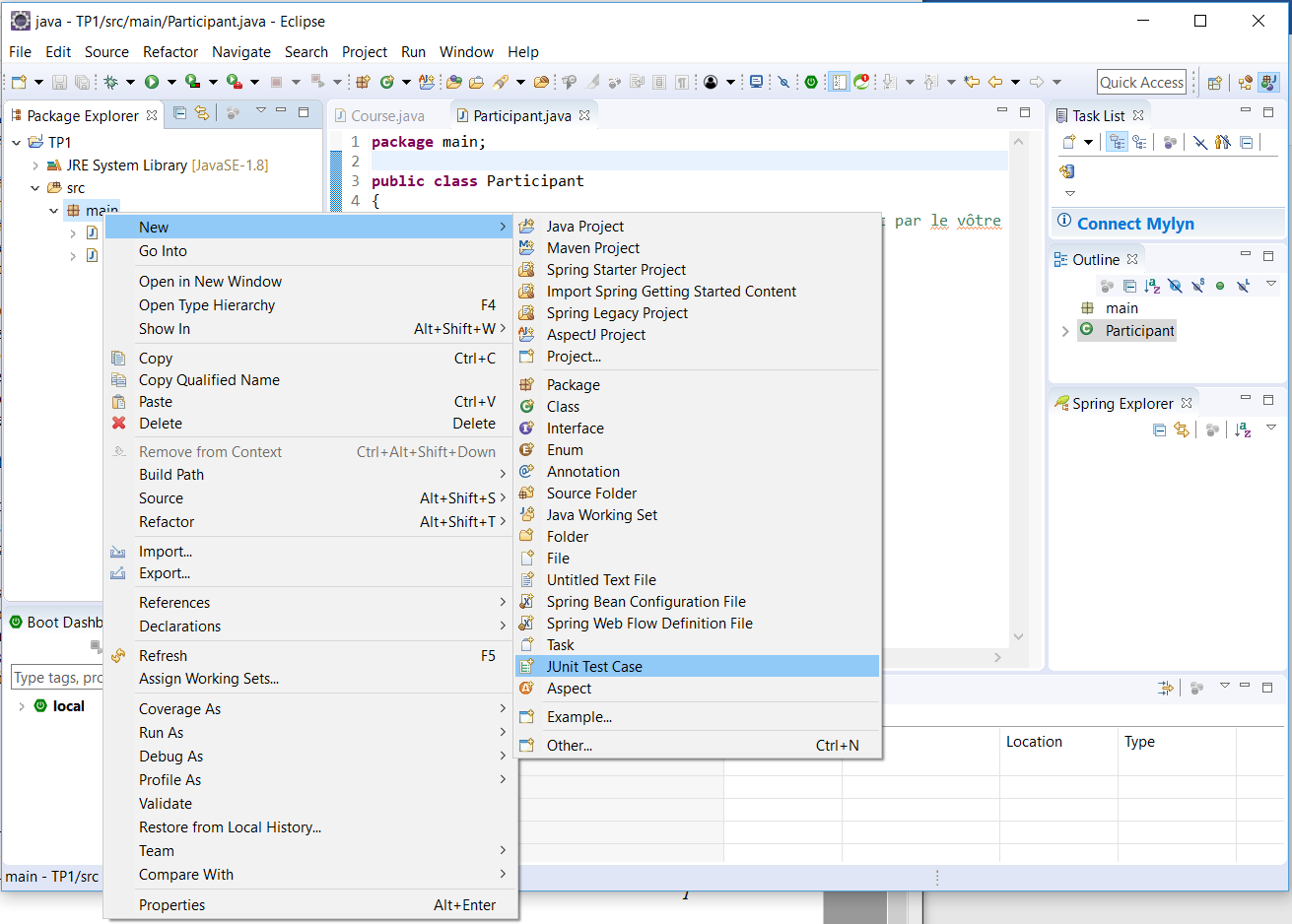


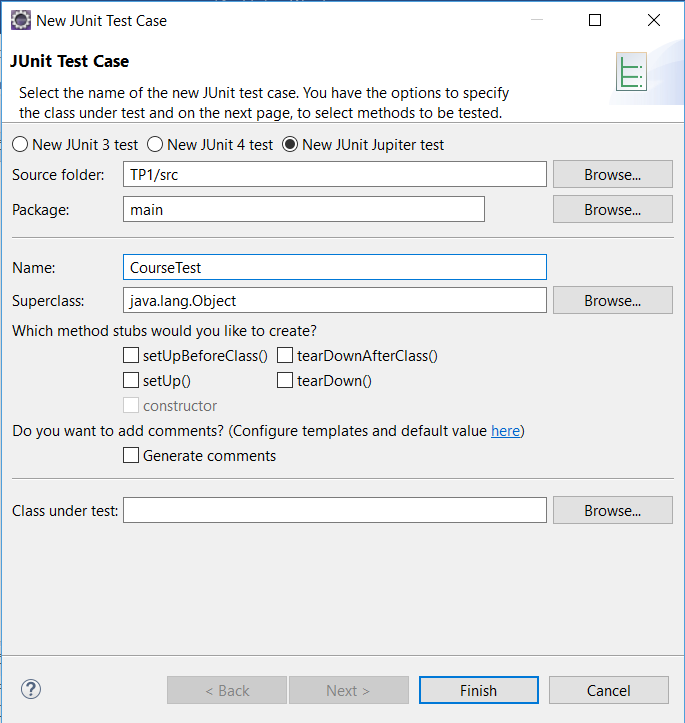




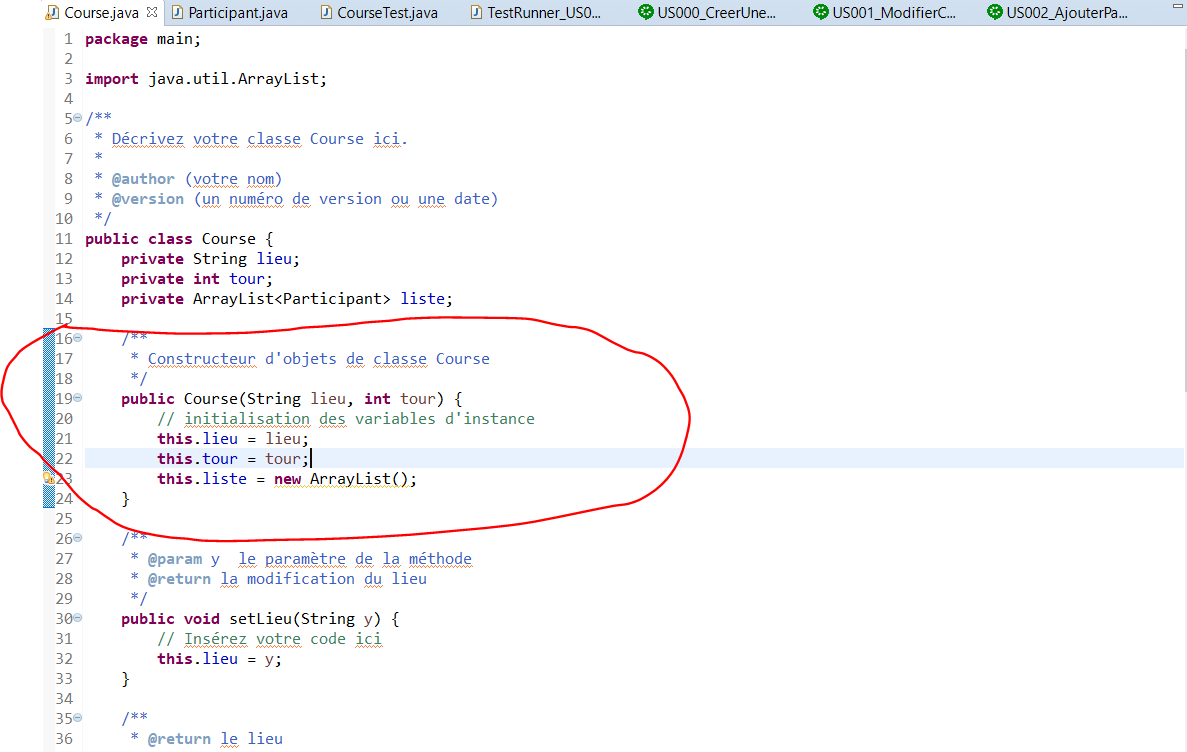


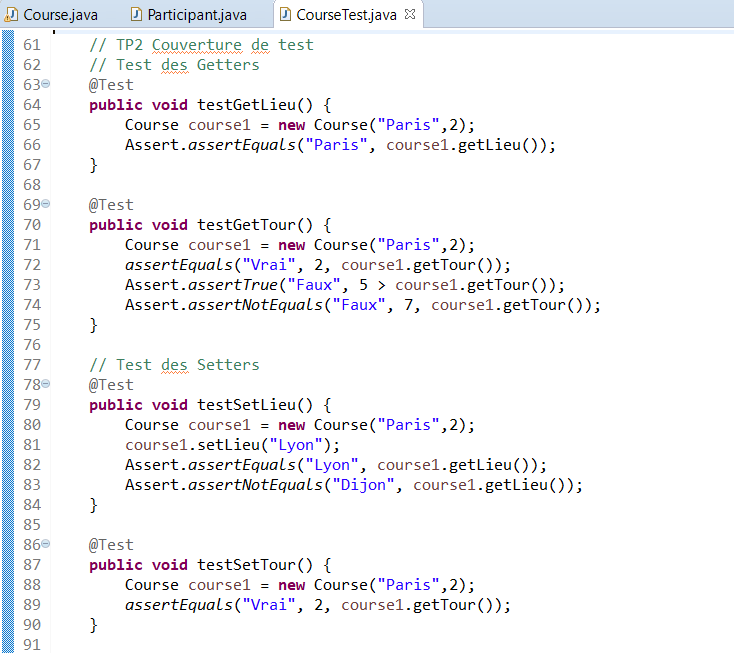


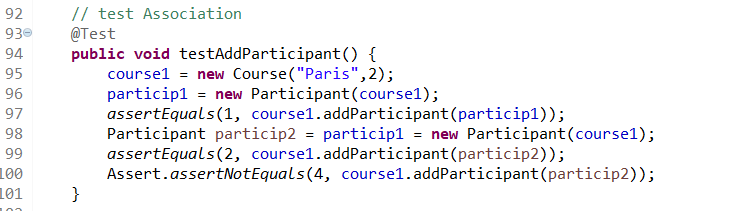


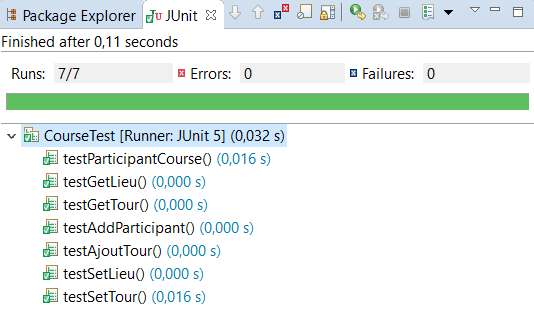


* 1. Tests JUnit



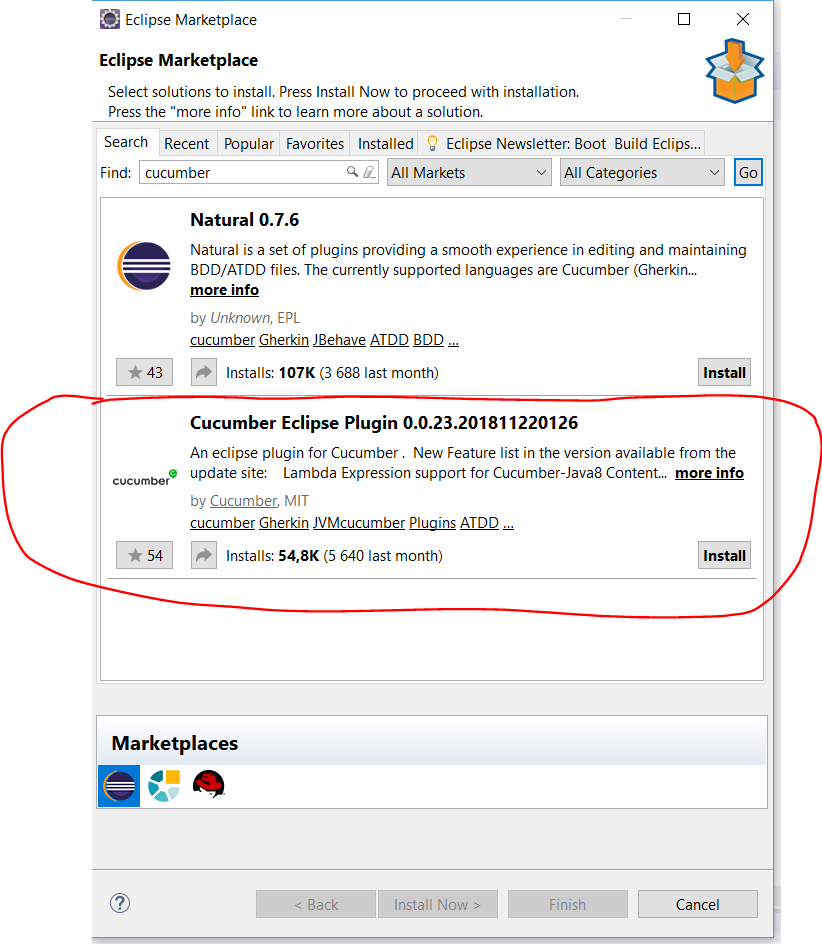




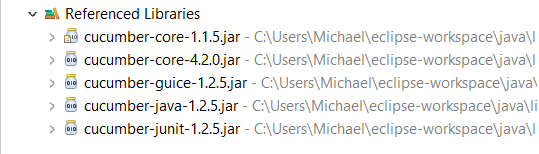


* 1. Cucumber via Eclipse
     1. Installation de Cucumber

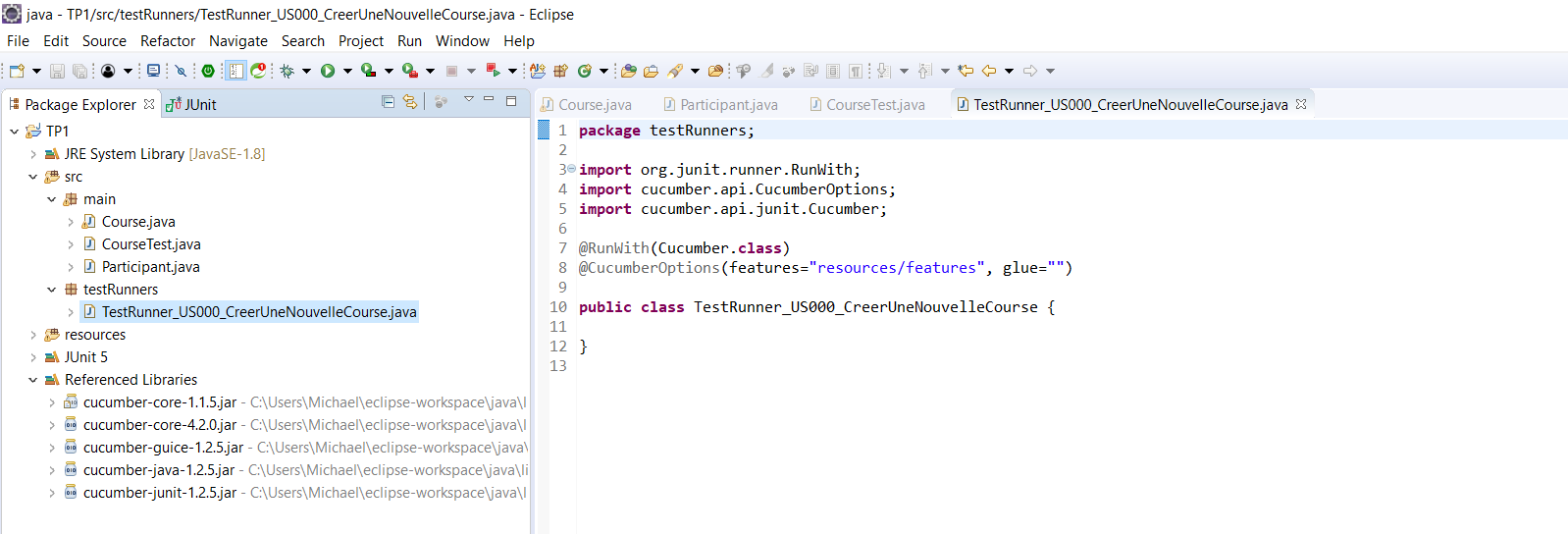
Installation de Cucumber



Ajout via le Build Path Eclipse :



* + 1. Configuration de Cucumber pour exécution des tests



* + 1. Création des fichiers pour description des Users Story

