/Users/Nicolas/Documents/Etudes/Automne 2014/LOG2410/Travail personnel/PagePresentation/LogoPolytechnique.pdf

**LOG3430**

**Méthodes de test et de validation du logiciel**

Travail pratique 2

Section N°1

Nicolas Richard – 1681198

Adrien Budet – 1721823

David Kanaa – 1816481

Polytechnique Montréal

5 octobre 2015

Choix de conception des tests de type *boîte noire*

Classes implémentant l’interface *Calculator*

Si certes il existe de nombreuses similitudes entre les différentes implémentations de l’interface *Calculator* compte tenu de la nature identique des paramètres d’entrée, nous avons tout de même établi les catégories de tests de manière indépendante pour chacune des classes. Au final, nous avons obtenu 6 classes pour chaque test ; il s’agit des mêmes dans chaque cas. À ce titre, vous trouverez ci-dessous les différentes trames de tests :

Tableau 1 - Trames de test des classes *Addition*, *Soustraction* et *Multiplication*

|  |  |  |  |
| --- | --- | --- | --- |
| Paramètre | Catégories | Choix | Propriétés |
| Val1 | Valeur négative | -2147483648 |  |
| {-2147483647, -2} |  |
| -1 |  |
| Valeur nulle | 0 |  |
| Valeur positive | 1 |  |
| {2, 2147483646} |  |
| 2147483647 |  |
| Val2 | Valeur négative | -2147483648 |  |
| {-2147483647, -2} |  |
| -1 |  |
| Valeur nulle | 0 |  |
| Valeur positive | 1 |  |
| {2, 2147483646} |  |
| 2147483647 |  |

Tableau 2 - Trame de test la classe *Division*

|  |  |  |  |
| --- | --- | --- | --- |
| Paramètre | Catégories | Choix | Propriétés |
| Val1 | Valeur négative | -2147483648 |  |
| {-2147483647, -2} |  |
| -1 |  |
| Valeur nulle | 0 |  |
| Valeur positive | 1 |  |
| {2, 2147483646} |  |
| 2147483647 |  |
| Val2 | Valeur négative | -2147483648 |  |
| {-2147483647, -2} |  |
| -1 |  |
| Valeur nulle | 0 | [Erreur] |
| Valeur positive | 1 |  |
| {2, 2147483646} |  |
| 2147483647 |  |

À l’inverse de la catégorie *Valeur nulle* qui ne contient logiquement qu’un choix, nous avons appliqué le principe des valeurs limites dans la sélection des divers choix des catégories *Valeur négative* et *Valeur positive.* S’il est vrai en théorie que les entiers n’ont pas de limite mathématique, cela n’est toutefois pas des entiers de type *Integer* en *Java* qui ont une capacité limite allant de -2147483648 à 2147483647. Si la valeur d’un argument est à l’extérieur de cet intervalle, sa valeur se verra automatique convertie en une valeur dans l’intervalle, d’où le choix des valeurs définissant la capacité limite du type *Integer* à titre de valeurs limites des choix.

Une fois les trames de test réalisées, nous avons écrit les différents cas de tests correspondants à l’aide du cadre d’application JUnit. Pour ce faire, nous avons eu recours aux fonctionnalités de *tests paramétrisés*, permettant d’exécuter le même test plusieurs fois et ce, avec des paramètres d’entrée différents à chaque fois. Ainsi, nous évitons la duplication inutile du code, car nous n’avons qu’à écrire une fois la méthode de test au lieu d’écrire une méthode de test par paire de données de test, ce qui s’avéra immensément utile dans le cas des tests AC.

Constructeur de la classe principale

En ce qui a trait au constructeur de la classe *SuiteChainee*, seule méthode de cette classe à tester dans le cadre de ce travail, nous avons suivis les mêmes techniques que celles appliquées dans les tests de type *boîte noire* des classes implémentant l’interface *Calculator*. Ainsi, nous avons encore une fois eu recours aux *tests paramétrisés* pour implémenter nos cas de tests et ce, pour les mêmes raisons que celles citées plus haut. Évidemment, nous aurions ne pas utiliser cette technique et insérer plusieurs assertions au sein d’une même méthode de test de manière à simuler la technique utilisée. Toutefois, cette approche à comme désavantage de ne pas laisser les autres cas de test être exécutés dans le cas où l’un d’entre eux échoue.

Tableau 3 - Trame du constructeur de la classe *SuiteChainee*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Paramètre | Catégorie | Choix | Propriété | Contrainte |
| Chemin | Existant | *Nom d’un fichier* |  |  |
| Inexistant | Chaîne vide | [Erreur] | |
| f4f4 |  |  |
| Opérateur | Valide | Addition | O |  |
| Soustraction | O |  |
| Multiplication | O |  |
| Division |  |  |
| Invalide | dsfsf |  |  |
| Val1 | Valeur négative | -2147483648 |  |  |
| {-2147483647, -2} |  |  |
| -1 |  |  |
| Valeur nulle | 0 |  |  |
| Valeur positive | 1 |  |  |
| {2, 2147483646} |  |  |
| 2147483647 |  |  |
| Val2 | Valeur négative | -2147483648 |  |  |
| {-2147483647, -2} |  |  |
| -1 |  |  |
| Valeur nulle | 0 |  | Si O |
| Valeur positive | 1 |  |  |
| {2, 2147483646} |  |  |
| 2147483647 |  |  |
| Taille | Hors bornes | 1 | [Erreur] | |
| 11 | [Erreur] | |
| Dans les bornes | 2 | b |  |
| {3-9} |  |  |
| 10 |  |  |
| EtatVide | Vide | true |  |  |
| Non-vide | false |  |  |

Analyse des tests effectués