

Ecole National des Sciences Appliqués Oujda

Développement avancé: Tests unitaires automatisés

TP-2 : FrameWork JUnit 4.

I. Partie I (Premier contacte):

1. Créer un projet Java sous le nom TP2-1
2. Créer le package : **ma.junit4.tp.dev** qui va contenir les classes du TP
3. Créer le package : **ma.junit4.tp.test** qui va contenir les classes de test
4. Créer la classe **MathTools** dans le package **dev** et qui contient deux méthodes :
 - i. **public double carre(double n)** : qui prend comme argument un *double* et retourne son carré.
 - ii. **public int somme(int a, int b)** : qui calcule la somme de deux entiers.
5. Créer le corps de la classe **MathToolsTest** dans le package **test**, en utilisant eclipse :
 - i. File→New→ Other... → Java→ JUnit→ JUnit Test Case (cocher la case JUnit 4)
6. Créer un attribut de type **MathTools** dans la classe **MathToolsTest** et l'instancier dans la méthode **@Before** puis le libérer dans **@After**.
7. Utiliser des **AssertXYZ** pour automatiser les tests des deux méthodes carre et somme.
8. Exécuter les testes automatisés des deux méthodes carre et somme.

II. Partie II (Tests Paramétrables) :

1. Créer un projet Java sous le nom TP2-2
2. Créer le package : **ma.junit4.tp.dev** qui va contenir les classes du TP
3. Créer le package : **ma.junit4.tp.test** qui va contenir les classes de test
4. Créer la classe **Algorithme** dans le package **dev** et qui contient la méthode:
 - i. **public long allSum(long n)** : qui calcule la somme des n premiers entiers ($1 + 2 + \dots + n$).
5. En utilisant les tests paramétrés du JUnit 4 ; automatiser le test des cas suivants via une classe **AlgorithmeParamTest** du package **test** :

n	0	1	5	10
allSum(n)	0	1	15	55

6. Exécuter le teste paramétré.

III. Partie III (Test des performances)

1. En utilisant eclipse, créer le corps de la classe **AlgorithmeTest** dans le package **test** afin automatiser le test la méthode **allSum** de la **partie II**.
2. Tester le cas suivant et vérifier que le test ne dépasse pas **2 ms** :

n	100000000
allSum(n)	5000000050000000

3. Si le test est **KO** ; améliorer votre algorithme pour avoir un test **OK**.

IV. Partie III (Test des exceptions)

1. Améliorer la méthode **allSum** de la **partie II** pour déclencher une exception de type **IllegalArgumentException** dans le cas des entiers **négatifs**.
2. Ajouter une méthode **testSommeNegatif()** dans la classe **AlgorithmeTest** afin de vérifier que la méthode **allSum** déclenche une exception de type **IllegalArgumentException** dans le cas où n est négative.
3. Exécuter le test de la méthode **testSommeNegatif()**.