

# International e-miage

## Module C306

### Activité n°2 - 2017-2: Test Unitaires + Couverture de Code

[Jeefntumba@gmail.com](mailto:Jeefntumba@gmail.com)

#### NTUMBA KALENGA JOSE EMMANUEL

Exercice couverture de code :

NB : **j'ai utilisé l'outil EclEmma (dans l'IDE Eclipse) pour la couverture de code vu qu'avec le système Windows 10 qui est sur ma machine Je n'arrivais pas à faire tourner convenablement Corbetura.**

Je précise que j'ai juste amélioré et reconduit la résolution de l'activité2 du semestre passé vu que l'activité n'a pas changé, je n'avais pas validé le module C306 par manque d'un binôme avec qui travaillé l'activité N°3.

#### Question 1

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Devoir1C306	98,3 %	474	8	482
src	98,3 %	474	8	482
devoir	98,3 %	474	8	482
TabAlgos.java	95,7 %	179	8	187
TabAlgos	95,7 %	179	8	187
similaires(int[], int[])	100,0 %	8	0	8
moyenne(int[])	100,0 %	23	0	23
plusGrand(int[])	100,0 %	23	0	23
egaux(int[], int[])	84,4 %	27	5	32
mediane(int[])	100,0 %	41	0	41
trier(int[])	100,0 %	57	0	57
TestTabAlgos.java	100,0 %	295	0	295
TestTabAlgos	100,0 %	295	0	295
testMedianeTableauImpair()	100,0 %	31	0	31
testMoyenne()	100,0 %	31	0	31
testPlusGrand()	100,0 %	31	0	31
testMedianeTableauPair()	100,0 %	35	0	35
testTrier()	100,0 %	52	0	52
testEgaux()	100,0 %	56	0	56
testSimilaires()	100,0 %	56	0	56

#### Question 2

Pour moi c'est une meilleure couverture de code.

#### Exercice sudoku

Question 1 : Ecrire les tests

NB : **n'ayant pas bien compris l'importance de la méthode `public int getDimension();` je l'ai supprimée du programme. Parce que à mon avis la largeur et la hauteur d'une grille sudoku seront toujours égales (soit 9 X 9 ou 16 X 16),  $9/9 = 1$  ou  $16/16 = 1$ , au moins que j'aie mal compris.**

```

1  /**
2   *
3   */
4  package miage.c306.dev2;
5
6  /**
7   * @author jeefntumba@gmail.com
8   *
9   */
10 public interface Grille {
11
12     /**
13      * Caractere de case vide
14      */
15     static final char EMPTY = '@';
16
17     /**
18      * Caractere possible a mettre dans la grille
19      *
20      * pour une grille 9x9 : 1..9
21      *
22      * pour une grille 16x16: 0..9-a..f
23      */
24     static final char[] possible = new char[] { '1', '2', '3', '4', '5', '6',
        '7', '8', '9', '0', 'a', 'b', 'c', 'd', 'e', 'f' };
25
26     /**
27      * Affecte une valeur dans la grille
28      *
29      * @param x position x dans la grille
30      * @param y position y dans la grille
31      * @param value
32      * valeur a mettre dans la case
33      * @throw IllegalArgumentException si x ou y sont hors bornes (0-8)
34      * @throw IllegalArgumentException si la valeur est interdite aux vues des
35      * autres valeurs de la grille
36      * @throw IllegalArgumentException si value n'est pas un caractere autorise
37      * ('1', ..., '9')
38      */
39     public void setValue(int x, int y, char value) throws IllegalArgumentException;
40
41     /**
42      * Recupere une valeur de la grille
43      *
44      * @param x position x dans la grille
45      * @param y position y dans la grille
46      * @return valeur dans la case x,y
47      * @throw IllegalArgumentException si x ou y sont hors bornes (0-8)
48      */

```

```

47 public char getValue(int x, int y) throws IllegalArgumentException;
48 /**
49  * Test si une grille est terminee
50  *
51  * @return true si la grille est complete
52  */
53 public boolean complete();
54 /**
55  * Test si une valeur est possible dans la grille par rapport a ce qu'elle
56  * contient deja
57  *
58  * @param x position x dans la grille
59  * @param y position y dans la grille
60  * @param value
61  * valeur a mettre dans la case
62  * @throw IllegalArgumentException si x ou y sont hors bornes (0-8)
63  * @throw IllegalArgumentException si value n'est pas un caractere autorise
64  * ('1', ..., '9', ...)
65  */
66 public boolean possible(int x, int y, char value) throws IllegalArgumentException;
67 }
68

```

Question 2 : Ecrire une implémentation dans une classe

```
1 package miage.c306.dev2;
2
3 /**
4  * @author jeefntumba@gmail.com
5  *
6  */
7 public class GrilleImpl implements Grille {
8     /**
9      * borne region 1 (0,0 - 2,2) 3X3.
10     */
11     static final int REGION1 = 0;
12     /**
13      * borne region 2 (3,3 - 5,5) 3X3.
14     */
15     static final int REGION2 = 3;
16     /**
17      * borne region 3 (6,6 - 8,8) 3X3.
18     */
19     static final int REGION3 = 6;
20     /**
21      * borne sup. 9X9.
22     */
23     static final int BORNE = 9;
```

```

24  /**
25  /**
26  * borne sup. 9X9.
27  */
28  static final int BORNE_SUP_TOUS = 16;
29  /**
30  * calcule borne inferieure region 2.
31  */
32  static final int BORNER2 = 3;
33  /**
34  * calcule borne inferieure region 2.
35  */
36  static final int BORNER3 = 6;
37  /**
38  * la grille sudoku.
39  */
40  char[][] grille;
41  /**
42  * constructeur cree une grille 9X9.
43  */
44  public GrilleImpl() {
45      this.grille = new char[BORNE][BORNE];
46      generateGrille();
47  }

```

```

48  @Override
49  public final void setValue(final int x, final int y, final char value)
50      throws IllegalArgumentException {
51      if (possible(x, y, value)) {
52          if (this.grille[y][x] == Grille.EMPTY) {
53              this.grille[y][x] = value;
54          }
55      }
56  }
57  @Override
58  public final char getValue(final int x, final int y)
59      throws IllegalArgumentException {
60      if (isBornePossible(x, y)) {
61          throw new IllegalArgumentException();
62      }
63      return this.grille[y][x];
64  }

```

```

66 @Override
67 public final boolean complete() {
68     for (int y = 0; y < BORNE; y++) {
69         for (int x = 0; x < BORNE; x++) {
70             if (grille [y][x] == Grille.EMPTY) {
71                 return false;
72             }
73         }
74     }
75     return true;
76 }
77 @Override
78 public final boolean possible(final int x,
79 final int y, final char value)
80     throws IllegalArgumentException {
81     //verification dans la ligne
82     for (int i = 0; i < BORNE; i++) {
83         if (this.grille[y][i] == value) {
84             return false;
85         }
86     }
87     //verification dans la colonne
88     for (int i = 0; i < BORNE; i++) {
89         if (this.grille[i][x] == value) {
90             return false;
91         }
92     }
93     int regionx = regionCord(x);
94     int regiony = regionCord(y);
95     for (int j = regiony;
96 j <= regiony + 2; j++) {
97         for (int i = regionx;
98 i <= regionx + 2; i++) {
99             if (this.grille[j][i]
100 == value) {
101                 return false;
102             }
103         }
104     }
105     if (isBornePossible(x, y)
106 || isPossible(value)) {
107         throw new IllegalArgumentException();
108     }
109     return true;
110 }

```

```

111 /**
112  * cherche une region 3X3 sudoku.
113  * @param cord le parametre
114  * @return la borne inferieure pou x ou y
115  */
116 private int regionCord(final int cord) {
117     if (cord >= 0 && cord < BORNER2) {
118         return REGION1;
119     } else if (cord >= BORNER2 && cord < BORNER3) {
120         return REGION2;
121     } else if (cord >= BORNER3 && cord < BORNE) {
122         return REGION3;
123     }
124     return -1;
125 }
126 /**
127  * genere une grille 9 X 9 contenant '@'.
128  */
129 public final void generateGrille() {
130     for (int y = 0; y < BORNE; y++) {
131         for (int x = 0; x < BORNE; x++) {
132             grille [y][x] = Grille.EMPTY;
133         }
134     }
135 }

```

```

136  /**
137   * Test si est un caractere autorise.
138   * @param value  valeur
139   * @return vrai si valeur possible
140   */
141  public final boolean isPossible(final char value) {
142      for (int i = 0; i < BORNE_SUP_TOUS; i++) {
143          if (Grille.possible[i] == value) {
144              return true;
145          }
146      }
147      return false;
148  }
149  /**
150   * Test si est un caractere autorise.
151   * @param x  borne colonne
152   * @param y  borne ligne
153   * @return vrai si borne normale
154   */
155  public final boolean isBornePossible(final int x, final int y) {
156      if ((x >= 0 && x < BORNE) && (y >= 0 && y < BORNE)) {
157          return true;
158      }
159      return false;
160  }
161  }

```

### Question 3 : Résultats des tests unitaires

```

-----
T E S T S
-----
Running miage.c306.dev2.TestGrilleImpl
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.08 sec - in miage.c306.dev2.TestGrilleImpl

Results :

Tests run: 4, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.893 s
[INFO] Finished at: 2016-10-18T16:58:09+02:00
[INFO] Final Memory: 15M/163M
[INFO] -----
C:\apache-maven-3.3.9\tps\Devoir2_C306>

```



## Couverture de code

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
▼ totoMiageAsupprimei	60,2 %	254	168	422
▼ src	60,2 %	254	168	422
▼ miage.c306.dev2	60,2 %	254	168	422
▼ Grille.java	0,0 %	0	68	68
▼ TestGrilleImpl.java	77,0 %	87	26	113
▼ TestGrilleImpl	77,0 %	87	26	113
testGetValue()	0,0 %	0	9	9
testComplete()	100,0 %	5	0	5
testSetValue()	66,7 %	18	9	27
testPossible()	87,5 %	56	8	64
▼ GrilleImpl.java	69,3 %	167	74	241
▼ GrilleImpl	69,3 %	167	74	241
getDimension()	0,0 %	0	2	2
isPossible(char)	0,0 %	0	16	16
regionCord(int)	28,0 %	7	18	25
setValue(int, int, char)	31,8 %	7	15	22
getValue(int, int)	56,2 %	9	7	16
isBornePossible(int, int)	85,7 %	12	2	14
GrilleImpl()	100,0 %	13	0	13
complete()	84,6 %	22	4	26
generateGrille()	100,0 %	22	0	22
possible(int, int, char)	88,2 %	75	10	85

## Rapport chekstyle pour TestGrilleImpl

NB : **il reste 11 erreurs parce que je ne voulais pas créer ces différentes constantes, selon moi trouvées inutiles.**

```
C:\apache-maven-3.3.9\tps\Devoir2_C306>java -jar lib\checkstyle-6.10-all.jar -c sun_checks.xml src\test\java\TestGrilleImpl.java
Starting audit...
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:14:1: File contains tab characters (this is the first instance).
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:28:21: '7' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:28:24: '6' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:29:32: '6' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:29:35: '7' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:37:45: '6' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:37:48: '7' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:54:38: '3' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:55:38: '4' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:56:38: '6' devrait être défini comme une constante.
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\test\java\TestGrilleImpl.java:57:38: '8' devrait être défini comme une constante.
Audit done.
Checkstyle ends with 11 errors.
```

## Rapport chekstyle pour GrilleImpl

```
C:\apache-maven-3.3.9\tps\Devoir2_C306>java -jar lib\checkstyle-6.10-all.jar -c sun_checks.xml src\main\java\GrilleImpl.java
Starting audit...
C:\apache-maven-3.3.9\tps\Devoir2_C306\src\main\java\GrilleImpl.java:8:1: File contains tab characters (this is the first instance).
Audit done.
Checkstyle ends with 1 errors.
```