

Fadi.Alouani

BTS SIO1

groupe 1

15/09/25

Compte rendu TP3 java

Sommaire :

- **Introduction**
- **Réponse quizz java**
- **Exercice de programmation**
- **Exercices correction de bugs**
- **Cas pratiques**
- **Cas problème**
- **Conclusion**

Introduction :

Dans ce travail pratique, je vais renforcer mes compétences en programmation Java en répondant à des questions théoriques et en réalisant des exercices pratiques. Je vais approfondir mes connaissances sur les concepts de base du langage et apprendre à résoudre des erreurs courantes dans du code. Je vais également créer des applications simples pour mettre en pratique ce que j'ai appris. À la fin, je serai plus à l'aise avec les bases de Java et sa mise en œuvre dans des projets concrets.

Quizz

Quizz :

- 1) Le langage machine le plus basique niveau circuit est le langage machine, celui constitué du code binaire et qui est directement compris par le processeur.
- 2) Les langages qui permettent d'utiliser un vocabulaire comme read, write ou add sont de haut niveau car ils utilisent une syntaxe proche du langage humain.
- 3) Les règles du langage de programmation constituent la syntaxe car c'est elle qui définit les règles de grammaire d'un langage de programmation.
- 4) C'est le compilateur qui traduit les instructions de langage de haut niveau en code machine.
- 5) Les emplacements de mémoire nommés de l'ordinateur sont appelés : variables.
- 6) Les opérations individuelles utilisées dans un programme informatique sont souvent regroupées en unités logiques appelées : procédures
- 7) Une instance de classe est appelée un objet
- 8) Java a une architecture neutre car il peut s'exécuter sur n'importe quelle machine grâce à la JVM.
- 9) On doit compiler les classes écrites en Java dans un bytecode pour qu'il puisse être interprété par JVM.
- 10) Toutes les instructions de programmation Java doivent se terminer par un point-virgule car c'est ce qui permet de séparer les codes et de signaler la fin d'une ligne de code.

Exercice de programmation :

1. Identifiant de classe

- a. maClasse est légal mais non conventionnel, il vaut mieux rajouter une majuscule (MaClasse)
- b. void est illégal car c'est un mot clé réservé
- c. Golden Retriever est illégal car il contient un espace
- d. invoice# est illégal car # est un caractère non autorisé
- e. 36535CodePostal est illégal car il commence par un chiffre
- f. Appartement est légal et conventionnel car il commence par une Majuscule et que selon la convention les classes doivent commencer par une majuscule
- g. Même chose que f, légal et conventionnel et commence par une majuscule
- h. 8888 est illégal car commence par un chiffre
- i. Ecrantotal() est illégal car les parenthèses ne sont pas valides dans un identifiant
- j. Acompte_recevable est légal et conventionnel, les tirets du bas sont rares pour les classes.

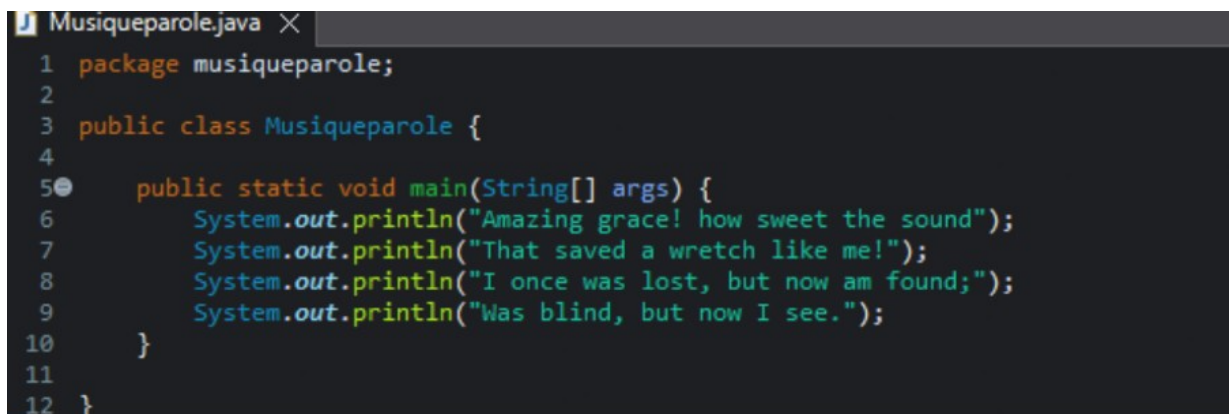
2. Identifiant de méthode

- a. associationRoles() est légal et conventionnel car il commence par une minuscule.
- b. void() est illégal car c'est un mot clé réservé
- c. Golden Retriever() est illégal car il contient un espace
- d. invoice#() est illégal car # est un caractère interdit
- e. 24500CodePostal() est illégal car il commence par un chiffre
- f. PayrollApp() est légal mais non conventionnel car il commence par une majuscule (pas recommandé)

- g. `getReady()` est légal et conventionnel car il commence par une minuscule
- h. `911()` est illégal car il commence par un chiffre
- i. `EcranTotal()` est légal mais non conventionnel car il commence par une majuscule
- j. `Acompte_Recevable()` est légal mais non conventionnel car il commence par une majuscule et les tirets du bas sont peu utilisés dans les méthodes Java

Comme on peut le voir, la différence entre les classes et les méthodes est que les classes doivent commencer par une majuscule pour respecter la convention tandis que les méthodes doivent commencer par une minuscule et posséder `()` à la fin pour respecter leur convention.

Code affichage lignes de chanson



```
Musiqueparole.java X
1 package musiqueparole;
2
3 public class Musiqueparole {
4
5     public static void main(String[] args) {
6         System.out.println("Amazing grace! how sweet the sound");
7         System.out.println("That saved a wretch like me!");
8         System.out.println("I once was lost, but now am found;");
9         System.out.println("Was blind, but now I see.");
10    }
11
12 }
```

Ici on affiche simplement 4 lignes différentes des 4 premières lignes de la chanson « Amazing Grace » de John Newton voici le resultat

Affichage d'un dessin d'une table et deux chaises

```
1 package tableetchaises;
2
3 public class TableetChaises {
4
5     public static void main(String[] args) {
6         System.out.println("X                X");
7         System.out.println("X                X");
8         System.out.println("X  XXXXXXXXXXXX  X");
9         System.out.println("XXXX  X  X  XXXX");
10        System.out.println("X  X  X  X  X  X");
11        System.out.println("X  X  X  X  X  X");
12    }
13 }
14 }
```

Affichage triangle en T

```
public class triangle {

    public static void main(String[] args) {
        System.out.println("    T");
        System.out.println("   TTT");
        System.out.println("  TTTT");
        System.out.println(" TTTTTT");
        System.out.println("TTTTTTTT");
        System.out.println("TTTTTTTTTT");
        System.out.println("TTTTTTTTTTTT");
    }
}
```

Exercices de correction de bugs

debugs 01

avant correction:

```
public class Debug1

    /* This program displays a greeting */
    public static void main(String[] args)
    {
        Systemoutprintln("Salut).
    }
```

apres correction:

```
public class Debug1 {

    /* This program displays a greeting */
    public static void main(String[] args)
    {
        System.out.println("Salut");
    }
}
```

```
TERMINAL
Salut
```

Nous pouvons constater que le code fonctionne affichant: salut

debugs 02

avant correction:

```
public class Debug2
{
    /* This program displays some output
    public static void main(String args)
    {
        System.out.println("Programmer en java est fun.");
        System.out.println("Faire un programme");
        System.out.println("peut être un challenge,");
        System.out.prnitln("mais quand la syntaxe est correcte,");
        System.out.println("c'est satisfaisant");
    }
}
```

apres correction:

```
1 public class Debug2
2 {
3     /* This program displays some output */
4     public static void main(String[] args)
5     {
6         System.out.println("Programmer en java est fun.");
7         System.out.println("Faire un programme");
8         System.out.println("peut être un challenge,");
9         System.out.println("mais quand la syntaxe est correcte,");
10        System.out.println("c'est satisfaisant");
11    }
12 }
```

voici ce qui est afficher dans le terminale

```
TERMINAL

Programmer en java est fun.
Faire un programme
peut etre un challenge,
mais quand la syntaxe est correcte,
c'est satisfaisant
```

debugs 03

avant correction:

```
public class Debug33
{
    public static void main(String[] args)
    {
        System.out.println("Derrière la rivière");
        system.out.println("et au dela du bois");
        SysTem.out.println("à la maison du garde nous irons");
    }
}
```

apres correction:

```
public class Debug3
{
    public static void main(String[] args)
    {
        System.out.println("Derriere la riviere");
        System.out.println("et au dela du bois");
        System.out.println("a la maison du garde nous irons");
    }
}
```


debugs 04

avant correction:

```
import javax.swing.JOptionPane;
public class Debug4
{
    public static main(String[] args)
    {
        JOptionPane.showMessageDialog(null, 1er GUI program)!
    }
}
```

apres correction:

```
import javax.swing.JOptionPane;

public class Debug4 {
    public static void main(String[] args) {
        JOptionPane.showMessageDialog(null, "1er GUI program");
    }
}
```

ayant une mauvaise connaissances des code et de ce qui les compose je me suis bien sur servi de l'ia pour m'aider.

Cas pratique

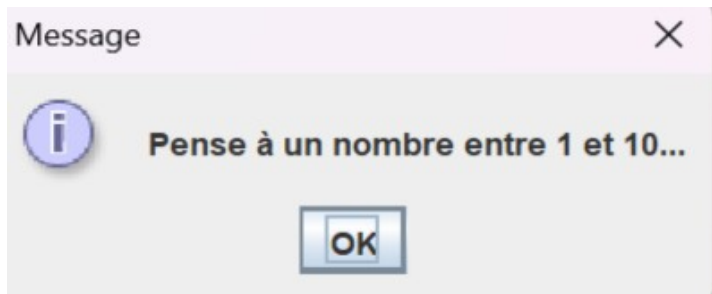
Pour pouvoir créer ce jeu de hasard, nous devons utiliser un logiciel de compilation JAVA avec partie interface graphique, pour cela j'ai choisi BlueJ.

Voici le code utilisé :

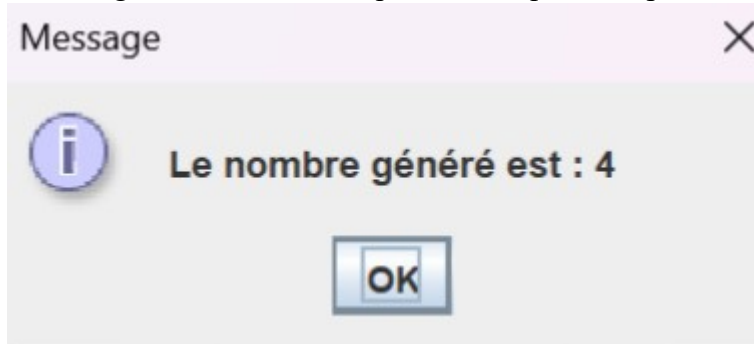
```
1 import javax.swing.JOptionPane;
2
3 public class Game {
4
5
6     public static void main(String[] args) {
7
8         JOptionPane.showMessageDialog(null, "Pense à un nombre entre 1 et 10...");
9
10
11         int nombreAleatoire = 1 + (int)(Math.random() * 10);
12
13
14         JOptionPane.showMessageDialog(null, "Le nombre généré est : " + nombreAleatoire);
15     }
16 }
```

Et voici le résultat :

une boîte de message demande de penser à un nombre entre 1 et 10 :



lorsqu'on clique sur OK, une deuxième fenêtre s'ouvre avec un nombre aléatoire, si le nombre généré est le même que celui auquel on a pensé alors on a perdu, sinon on gagne :



Cas-problème

On veut faire en sorte de pouvoir afficher un message seul, puis avec une bordure d'astérisques et enfin avec une bordure de Ss.

Tout d'abord on va utiliser BlueJ, on va créer un projet nommé « Yummy », ensuite on va créer un paquetage nommé Yummy, puis on va à l'intérieur de celui-ci créer trois classes, la première pour le message sans bordure, la deuxième pour le message entouré d'astérisques et le troisième avec la bordure en Ss.

Voici à quoi ressemble le code de Yummy.java :

```
1 package yummy;
2
3 public class Yummy {
4     public static void main(String[] args) {
5         System.out.println("Yummy prépare les meilleurs plats pour vos fêtes");
6     }
7 }
```

Et voici ce que ça affiche dans le terminal :

```
Yummy prépare les meilleurs plats pour vos fêtes
```

Ensuite voici Yummy2.java :

```
package yummy;

public class Yummy2 {
    public static void main(String[] args) {
        String message = "Yummy prépare les meilleurs plats pour vos fêtes";
        String border = "*".repeat(message.length() + 4);

        System.out.println(border);
        System.out.println("* " + message + " *");
        System.out.println(border);
    }
}
```

Et voilà ce que le code nous fait :

