IS2A4 – PPO – T.P. 1a

Résumé

Ce TP aborde les classes et les relations entre les classes en java.
N'hésitez pas à consulter la documentation en ligne sur le site d'oracle : `https://docs.oracle.com/javase/8/docs/api/` (regardez la version correspondant la version de java que vous utilisez).

Prise en main de Java (5 min)

Nous allons utiliser l'environnement standart JDK (Java Develpment Kit) qui comprend notament les commandes de compilation `javac` et d'exécution `java` et de nombreuses bibliothèques (packages de classes) dont la documentation (la "javadoc") est sur : `file:///usr/localTP/jdk/docs/api/index.html`.

Ouvrez ce fichier dans votre navigateur (firefox) (conservez ce lien dans vos bookmarks préférés). La documentation est entièrement navigable. À gauche, vous pouvez choisir une navigation alphabétique de "All Classes" ou d'un package particulier. À droite apparaissent les informations sur la sélection, notez les onglets :

- "Tree" : hiérarchie des packages et des classes de la sélection
- "Index" : index alphabétique de tous les symboles associés (classes, variables, méthodes, constructeurs, ...).

Observer comment sont décrites quelques classes vues en cours : `Object`, `String`, ...

I Préparation du TP (5 min)

I.1. Créer un dossier spécifique pour les TP de PPO qui vous appellerez `TP_PPO` par exemple.

I.2. Dans ce dossier, créer un dossier `TP1a`. C'est dans ce dossier que vous travaillerez aujourd'hui. Durant les prochaines séances de TP, essayez de garder cette pratique.

II Votre Première classe : `Livre` (40 min)

Dans ce premier TP, on va s'interesser à une bibliothèque et donc l'objectif va être de construire une bibliothèque avec des livres.
Pour commencer, on va aller pas à pas dans l'écriture de votre première classe en java qui sera votre classe `Livre`. À chaque étape, n'oubliez pas de compilez votre code pour vérifier qu'il n'y a pas d'erreurs de syntaxe.

II.1. Comme on l'a vu en cours, pour construire la classe `Livre`, il faut créer un fichier `Livre.java` qui va construire la classe `Livre`. Créer ce fichier en y ajoutant la classe demandée :

```
// Debut de la classe
class Livre {
    // Interieur de la classe
    // On y mettra les variables, les methodes et les constructeurs
}
// Fin de la classe
```

II.2. Nos livres ont chacun un auteur, un titre et un prix. Ajoutez à l'intérieur de la classe `Livre`, ces variables d'instances.

II.3. Pour pouvoir instanciez des objets, ajoutez un constructeur à la classe `Livre` qui prendra en paramètre un auteur, un titre et un prix.

II.4. Ajoutez un accesseur (une méthode qui renvoie une valeur) `getAuteur` qui va permettre de renvoyer l'auteur d'un objet `Livre`.

II.5. C'est bien joli de créer des classes mais c'est plus marrant en les instanciant. Créez une nouvelle classe `Main` (dans un fichier `Main.java` qui sera dans le même dossier que le fichier `Livre.java`) qui ne comportera, pour l'instant, qu'une méthode `main` : c'est dans cette méthode que nous instancierons les objets qui nous voudrons afficher sur le terminal. Instanciez un livre et affichez son auteur à l'aide de `System.out.println` (cette méthode prend en entrée une variable de type `String`).

Exemple de ce que vous devriez avoir comme code :

Fichier : `Livre.java`

```java
class Livre {
    String auteur;
    String titre;
    int prix;

    Livre(String auteur, String titre, int prix) {
        this.auteur = auteur;
        this.titre = titre;
        this.prix = prix;
    }

    String getAuteur() {
        return this.auteur;
    }
}
```

Fichier : `Main.java`

```java
class Main {
    public static void main(String[] args) {
        Livre monLivre = new Livre("Zola", "Nana",19);
        System.out.println(monLivre.getAuteur());
    }
}
```

II.6. Compilez vos classes et exécutez la classe `Main`. Vous devriez avoir ce type de sortie :

```
console$ javac Livre.java
console$ javac Main.java
console$ java Main
Zola
```

II.7. Ajoutez un mutateur (une méthode qui modifie une valeur) `setTitre` qui va permettre de modifier la variable `titre` d'un objet `Livre` et modifiez le titre du livre que vous avez instanciez dans la classe `Main`.

Fichier : `Livre.java`

```java
    // ...
    void setTitre(String newTitre) {
        this.titre = newTitre;
    }
    // ...
```

Fichier : `Main.java`

```java
        // ...
        monLivre.setTitre("Germinal");
        // ...
```

II.8. Ajoutez un accesseur `getTitre` qui va vous permettre de récupérer la variable `titre` d'un objet `Livre`. Affichez la valeur du titre du livre instancié avant et après sa modification pour vérifier que la modification est bien effective.

III Amélioration : la classe `Auteur` (20 min)

III.1. Plusieurs objets de type `Livre` peuvent partager le même auteur. Afin d'uniformiser les auteurs, vous allez créer une classe `Auteur` qui a un `nom` et un accesseur `getNom` qui retourne le `nom` de l'objet.

III.2. Modifiez le code de la classe `Livre` pour que l'`auteur` soit un objet de type `Auteur` (n'oubliez pas de modifier le constructeur qui va prendre une variable de type `Auteur` en paramêtre).

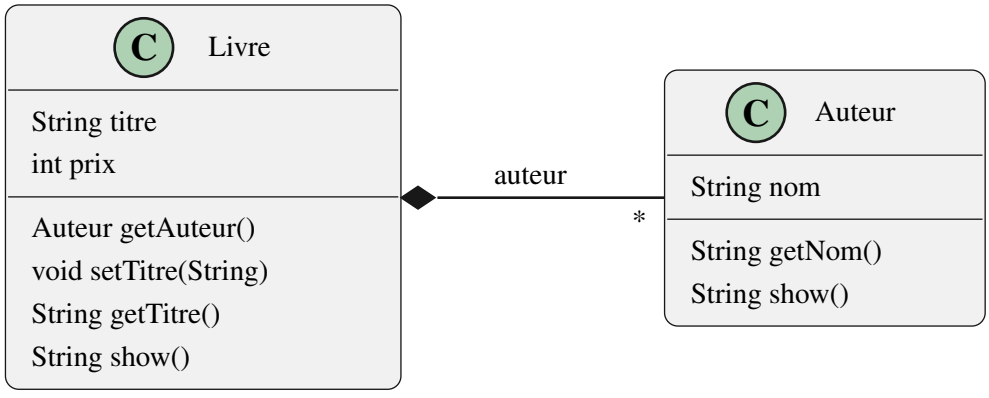

FIGURE 1 – Diagramme de classe avec `Livre` et `Auteur` (Final version).

III.3. Testez votre nouvelle classe `Auteur` dans votre classe `Main` en instanciant un auteur et en créant deux livres avec le même auteur :

Fichier : `Main.java`

```
4       // ...
5       Auteur zola = new Auteur("Zola");
6       Livre monLivre = new Livre(zola, "Nana",19);
7       Livre monLivre2 = new Livre(zola, "Germinal",24);
8       System.out.println(monLivre.show());
9       System.out.println(monLivre2.show());
10      // ...
```

III.4. Pour améliorer la visualisation de vos classe, rajoutez une méthode `show()` dans chacune des classes `Livre` et `Auteur` qui retournera dans un `String` les informations de chaque objet.

Fichier : `Livre.java`

```
21  // ...
22  String show() {
23      return this.titre + ",␣" + this.auteur.show() + ",␣" + this.prix;
24  }
25  // ...
```

Fichier : `Auteur.java`

```
13  // ...
14  String show() {
15      return "Auteur␣:␣" + this.nom;
16  }
17  // ...
```

IV Somme des livres (15 min)

IV.1. Pour garder la somme des prix des livres créés, ajoutez une variable de classe `somme` à la classe `Livre` qui sera initialisé à 0 et qui ajoutera dans le constructeur le prix du nouvel objet qui sera instancié :

Fichier : `Livre.java`

```
6   // ...
7   static int somme = 0;
8   // ...
```

Fichier : `Livre.java`

```
13  Livre(Auteur auteur, String titre, int prix) {
14      this.auteur = auteur;
15      this.titre = titre;
16      this.prix = prix;
17      Livre.somme += this.prix;
18  }
```

IV.2. Ajoutez une méthode statique `getSomme` dans la classe `Livre` qui retourne la variable de la variable statique `somme` :

Fichier : `Livre.java`

```
34  // ...
35  static int getSomme() {
36      return Livre.somme;
37  }
38  // ...
```

IV.3. Dans votre classe `Main`, calculez la somme totale des livres qui vous avez déjà instanciez :

Fichier : `Main.java`

```
9       // ...
10      System.out.println(Livre.getSomme());
11      // ...
```

V Bibliothèque (15 min)

V.1. On voudrait regrouper certains livres en une bibliothèque. Ajoutez une variable `biblio` dans le fichier `Main.java` qui sera une tableau de trois livres.

Fichier : `Main.java`

```
10      Livre[] biblio = {new Livre(zola, "Nana",19),
11                        new Livre(zola, "Germinal",24),
12                        new Livre(new Auteur("Hugo"), "Claude␣Gueux",5)};
```

V.2. Faites une boucle sur les différents livres de votre tableau de livres `biblio` pour récupérer la somme des prix et affichez le. Attention une accesseur pour accéder au prix d'un livre peut être utile.

Fichier : `Main.java`

```
15          int somme = 0;
16          for (Livre l : biblio) {
17              somme += l.getPrix();
18          }
19          System.out.println(somme);
```

V.3. Créez une classe `Bibliotheque` qui :

- se construit à partir d'un tableau de livres
- garde en mémoire cette liste de livres
- a une méthode `getSomme` qui donne la somme des prix des livres
- a une méthode `show` qui retourne un `String` correspondant à l'affichage de cette liste de livres.

VI Encapsulation et package (20 min)

VI.1. Modifiez les classes `Auteur`, `Livre` et `Bibliotheque` pour respecter l'encapsulation.

VI.2. Créez deux packages : le package `media` avec les classes `Auteur` et `Livre` et le package `mediatheque` avec la classe `Bibliotheque`.

VII Bibliothèque avancée (bonus)

VII.1. On voudrait rendre notre classe `Bibliotheque` dynamique, c'est à dire qu'on voudrait pouvoir ajouter des livres au fur et à mesure dans notre bibliothèque avec une méthode `add`. Pour faire cela, dupliquer votre classe `Bibliotheque` en une nouvelle classe `BibliothequeAdd` dans le package `mediatheque` et ajoutez un constructeur qui, pour un nombre de livres maximum, crée une bibliothèque vide (mais initialise un tableau de livre de cette longueur en gardant en mémoire le nombre de livres ajoutés). Faites en sorte que si vous ajoutez un livre dans une bibliothèque déjà pleine alors celui-ci prend la place du livre le plus ancien (le plus ancien a avoir été ajouté) de votre bibliothèque.

VII.2. On voudrait rendre notre classe `BibliothequeAdd` encore plus dynamique, en autorisant la suppression d'un livre de la bibliothèque à partir de sa position dans le tableau : la méthode `remove(i)` supprimera le livre à la position `i` dans le tableau de livres de la bibliothèque (pour l'ajout, les rêgles sont les mêmes que précédemment). Créez une classe `BibliothequeAddRemove` dans le package `mediatheque` qui permet l'ajout et la suppression de livres.

`Temps total du sujet : 2 heures et 0 minutes`