TP3 - JAVA : Les listes

BACHOUR Peter

12 Mai 2020

Table des matières

1 Introduction **2**

2 Définition **3**

3 Classe Cours **4**

4 Classe Formation **5**

5 Classe Main **6**

6 Tri de liste **8**

7 Conclusion **9**

Table des figures

1 ArrayList implémente List 3
2 Classe Cours 4
3 Classe Formation 5
4 Classe Main 6
5 addCours() et removeCours() 6
6 Affichage de la formation et de ses cours 7
7 Affichage de la formation et de ses cours après supression du 2^{me} cours 7
8 Affichage de la formation et de ses cours trier par rapport à leur intitulé 8

1 Introduction

Dans le cadre du troisième TP en JAVA, nous allons nous focaliser sur les listes ou autrement dit **ArrayList**<>.

2 Définition

Les listes ou ArrayList en Java sont utilisées pour stocker une collection d'éléments de taille dynamique. Contrairement aux tableaux dont la taille est fixe, une liste de tableaux augmente automatiquement sa taille lorsque de nouveaux éléments lui sont ajoutés.

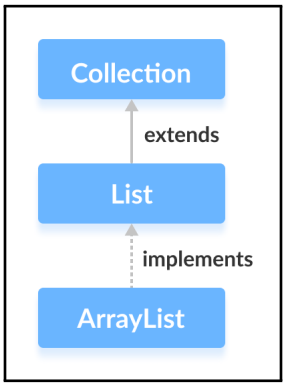

FIGURE 1: ArrayList implémente List

3 Classe Cours

Comme on a expliqué l'implémentation du'une classe dans les TP précédant, voici l'implémentation de la classe cours.

```java
public class Cours {

    private String code;
    private String intitule;
    private Double nbHeure;

    public Cours(String code, String intitule, Double nbHeure) {
        this.code = code;
        this.intitule = intitule;
        this.nbHeure = nbHeure;
    }

    public String getCode() { return code; }

    public void setCode(String code) { this.code = code; }

    public String getNom() { return intitule; }

    public void setNom(String nom) { this.intitule = nom; }

    public Double getNbHeure() { return nbHeure; }

    public void setNbHeure(Double nbHeure) { this.nbHeure = nbHeure; }

    @Override
    public String toString() {
        return "Le cours [" + code + "] est intitulé \r\n" +
                code + "et consiste de " +
                nbHeure + "nombre d'heures. \r\n";
    }
```

FIGURE 2: Classe Cours

4 Classe Formation

Voici l'implémentation de la classe formation. Comme on peut le remarquer

```java
private ArrayList<Cours> listeCours;
private String code_Formation;
private String nom_Formation;

public Formation(String code_Formation, String nom_Formation) {
    this.code_Formation = code_Formation;
    this.nom_Formation = nom_Formation;
    listeCours = new ArrayList<Cours>();
}

public ArrayList<Cours> getListeCours() { return listeCours; }

public void setListeCours(ArrayList<Cours> listeCours) { this.listeCours = listeCours; }

public String getCode_Formation() { return code_Formation; }

public void setCode_Formation(String code_Formation) { this.code_Formation = code_Formation; }

public String getNom_Formation() { return nom_Formation; }

public void setNom_Formation(String nom_Formation) { this.nom_Formation = nom_Formation; }

@Override
public String toString() {
    String out = "La formation [" + code_Formation + "] est intitulée " + nom_Formation +
            " et est constituée des cours suivants: \r\n";
        for(Cours cours : listeCours){
            out += cours;
        }
    return out;
}
```

FIGURE 3: Classe Formation

dans cette classe on a utilisé **ArrayList<Cours>** pour stocker les cours de la formation.

5 Classe Main

La création de 7 cours et leurs enregistrements dans une formation : Afin d'ajou-

```
public class Main {

    public static void main(String[] args) {

        Formation formation = new Formation( code_Formation: "cc",  nom_Formation: "ccc");

        Cours cours1 = new Cours( code: "cours1",  intitule: "cours1",  nbHeure: 12.75);
        Cours cours2 = new Cours( code: "cours2",  intitule: "cours2",  nbHeure: 12.75);
        Cours cours3 = new Cours( code: "cours3",  intitule: "cours3",  nbHeure: 12.75);
        Cours cours4 = new Cours( code: "cours4",  intitule: "cours4",  nbHeure: 12.75);
        Cours cours5 = new Cours( code: "cours5",  intitule: "cours5",  nbHeure: 12.75);
        Cours cours6 = new Cours( code: "cours6",  intitule: "cours6",  nbHeure: 12.75);
        Cours cours7 = new Cours( code: "cours7",  intitule: "cours6",  nbHeure: 12.75);

        formation.addCours(cours1);
        formation.addCours(cours4);
        formation.addCours(cours5);
        formation.addCours(cours6);
        formation.addCours(cours2);
        formation.addCours(cours3);
        formation.addCours(cours7);
        System.out.println(formation);
```

FIGURE 4: Classe Main

ter ou de supprimer les cours de la formation, on a eu recours aux deux fonctions suivantes dans la classe Formation :

```
public void addCours(Cours cours){
    listeCours.add(cours);
}

public void removeCours(int index){
    listeCours.remove(index);
}
```

FIGURE 5: addCours() et removeCours()

L'affichage sera le suivant :

```
La formation [Formation] est intitulée Formation et est constituée des cours suivants:
Le cours [cours1] est intitulé
cours1 et consiste de 12.75 nombre d'heures.
Le cours [cours4] est intitulé
cours4 et consiste de 12.75 nombre d'heures.
Le cours [cours5] est intitulé
cours5 et consiste de 12.75 nombre d'heures.
Le cours [cours6] est intitulé
cours6 et consiste de 12.75 nombre d'heures.
Le cours [cours2] est intitulé
cours2 et consiste de 12.75 nombre d'heures.
Le cours [cours3] est intitulé
cours3 et consiste de 12.75 nombre d'heures.
Le cours [cours7] est intitulé
cours7 et consiste de 12.75 nombre d'heures.
```

FIGURE 6: Affichage de la formation et de ses cours

Après avoir supprimer le 2^{me} cours on obtient l'affichage suivant :

```
La formation [Formation] est intitulée Formation et est constituée des cours suivants:
Le cours [cours1] est intitulé
cours1 et consiste de 12.75 nombre d'heures.
Le cours [cours5] est intitulé
cours5 et consiste de 12.75 nombre d'heures.
Le cours [cours6] est intitulé
cours6 et consiste de 12.75 nombre d'heures.
Le cours [cours2] est intitulé
cours2 et consiste de 12.75 nombre d'heures.
Le cours [cours3] est intitulé
cours3 et consiste de 12.75 nombre d'heures.
Le cours [cours7] est intitulé
cours7 et consiste de 12.75 nombre d'heures.
```

FIGURE 7: Affichage de la formation et de ses cours après supression du 2^{me} cours

6 Tri de liste

L'implémentation de la fonction **compareTo()** est obligatoire après avoir ajouter la **implements Comparable** à la classe Cours.
Enfin, afin de trier la liste, il suffit d'utiliser la fonction **Collections.sort(formation.getListeCours()) ;** et la liste de cours sera trier par rapport à l'intitulé du cours :

```
La formation [Formation] est intitulée Formation et est constituée des cours suivants:
Le cours [cours1] est intitulé
cours1 et consiste de 12.75 nombre d'heures.
Le cours [cours2] est intitulé
cours2 et consiste de 12.75 nombre d'heures.
Le cours [cours3] est intitulé
cours3 et consiste de 12.75 nombre d'heures.
Le cours [cours5] est intitulé
cours5 et consiste de 12.75 nombre d'heures.
Le cours [cours6] est intitulé
cours6 et consiste de 12.75 nombre d'heures.
Le cours [cours7] est intitulé
cours7 et consiste de 12.75 nombre d'heures.
```

FIGURE 8: Affichage de la formation et de ses cours trier par rapport à leur intitulé

7 Conclusion

En conclusion, à travers ce TP, nous avons découvert les listes et les différentes fonctions qu'ils apportent.