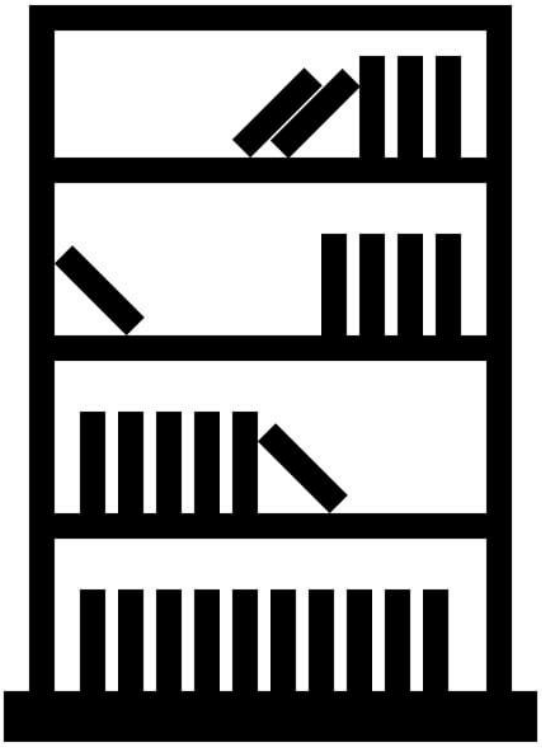
Bibliothèque



Ce TP Java a été donné par Mr Lefebvre Pierre,

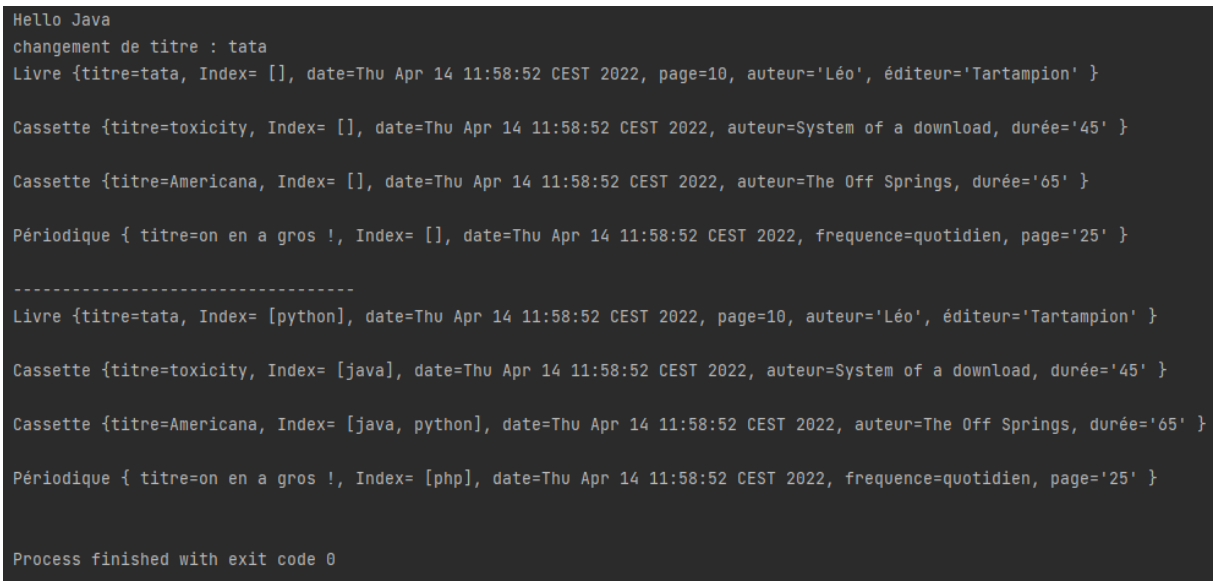
Professeur de programmation Pyhton, Java et C.

Il nous a fait travailler sur la gestion des objets et l'héritage sur plusieurs cours afin de mieux appréhender le codage orienté objet.

Instruction :

En partant de modèles UML, il nous a fallu interpréter et créer une bibliothèque qui contiendrait des documents de différents types : livre, cassette et périodique.

Présentation du visuel :



On peut remarquer que le visuel est coupé en deux.
La partie du haut était la première partie du projet : création des différents objets.

La partie du bas fut l'utilisation d'un nouveau type de variable : vector. Avec ce genre de variable, il est possible de créer un tableau dynamique et non plus statique.
Pour cette deuxième partie, nous avons aussi utilisé un type Enumeration afin d'afficher les différents éléments contenu dans le vector tDocument.

Le code source et leur action

Main.java

```java
package iu;

import om.*;


class Main {
```

Package, import et classe.

```java
Document tonton = new Livre( titre: "tonton",  page: 10,  auteur: "Léo",  editeur: "Tartampion");
tonton.setTitre("tata");
System.out.println("changement de titre : "+tonton.getTitre());
System.out.println(tonton.toString()+"\n");
tonton.setIndex("python");
```

Nouveau document tonton de type Livre.

```java
Document k7 = new Cassette( titre: "toxicity",  auteur: "System of a download",  duree: 45);
System.out.println(k7.toString()+"\n");
k7.setIndex("java");
Document k8 = new Cassette( titre: "Americana", auteur: "The Off Springs",  duree: 65);
System.out.println(k8.toString()+"\n");
k8.setIndex("java");
k8.setIndex("python");
```

Nouveaux documents de type Cassette.

```java
Document interview8 = new Periodique( titre: "on en a gros !",  frequence: "quotidien",  page: 25);
System.out.println(interview8.toString()+"\n");
interview8.setIndex("php");
```

Nouveau document de type Periodique.

```
29          System.out.println("-----------------------------------");
```

Séparation entre les différents cours.

```
31          Bibliotheque horotopia = new Bibliotheque( nom: "Horotopia");
32          horotopia.ajouterDocument(tonton);
33          horotopia.ajouterDocument(k7);
34          horotopia.ajouterDocument(k8);
35          horotopia.ajouterDocument(interview8);
36
37          horotopia.document();
```

Nouvelle Bibliothèque contenant les documents cités et appel de la méthode document.

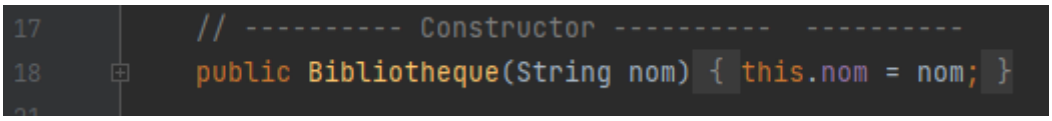
Bibliotheque.java

```
1   package om;
2
3   import java.util.Enumeration;
4   import java.util.Vector;
5
6   public class Bibliotheque {
```

Package, import et classe.

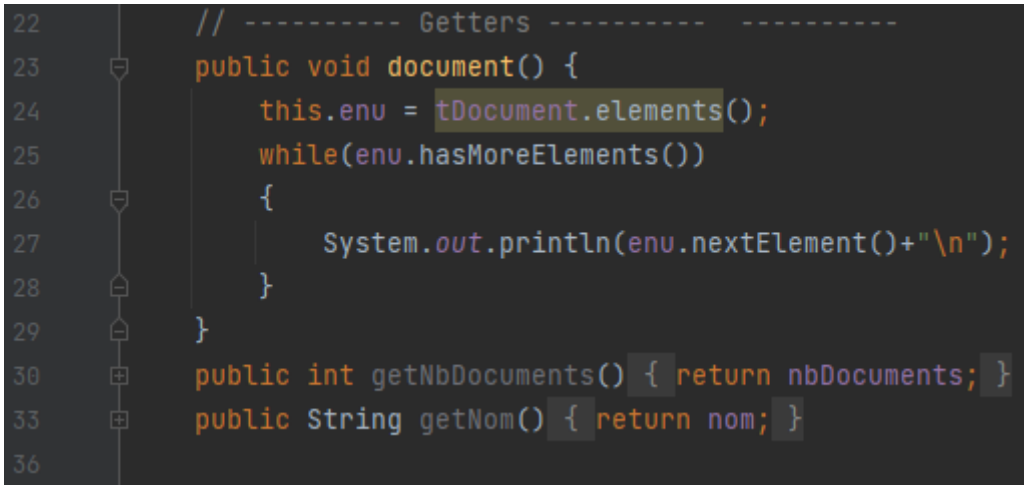
```
8       // on change le tableau statique
9       // private Document tDocument[] = new Document[4];
10      // en tableau dynamique
11      private Vector tDocument = new Vector();
12      private Enumeration<Document> enu;
13      private Vector list = new Vector();
14      private int nbDocuments = 0;
15      private String nom;
```

Déclaration des attributs.



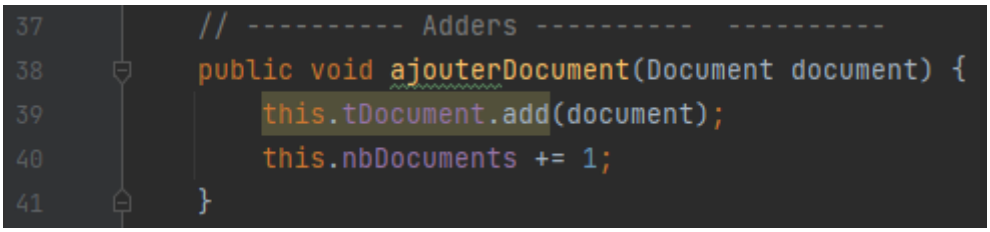
```
17      // ---------- Constructor ----------  ----------
18      public Bibliotheque(String nom) { this.nom = nom; }
```

Constructor et l'attribut demandé en paramètre.



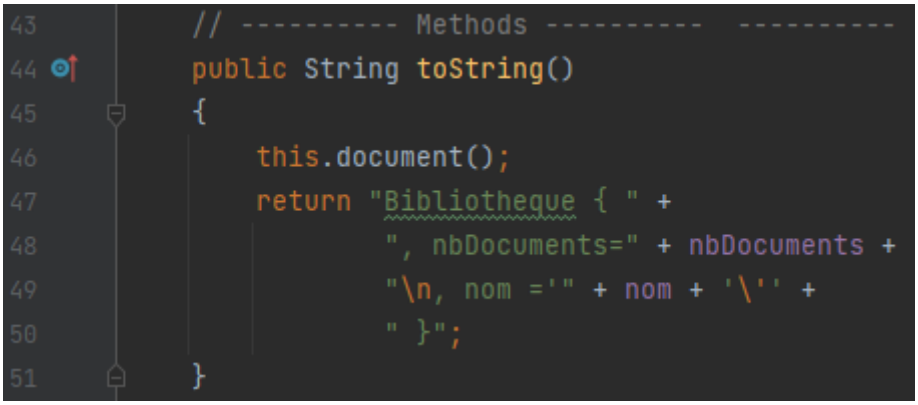
```
22      // ---------- Getters ----------  ----------
23      public void document() {
24          this.enu = tDocument.elements();
25          while(enu.hasMoreElements())
26          {
27              System.out.println(enu.nextElement()+"\n");
28          }
29      }
30      public int getNbDocuments() { return nbDocuments; }
33      public String getNom() { return nom; }
36
```

La fonction document() affiche les différents documents contenu dans tDocument.



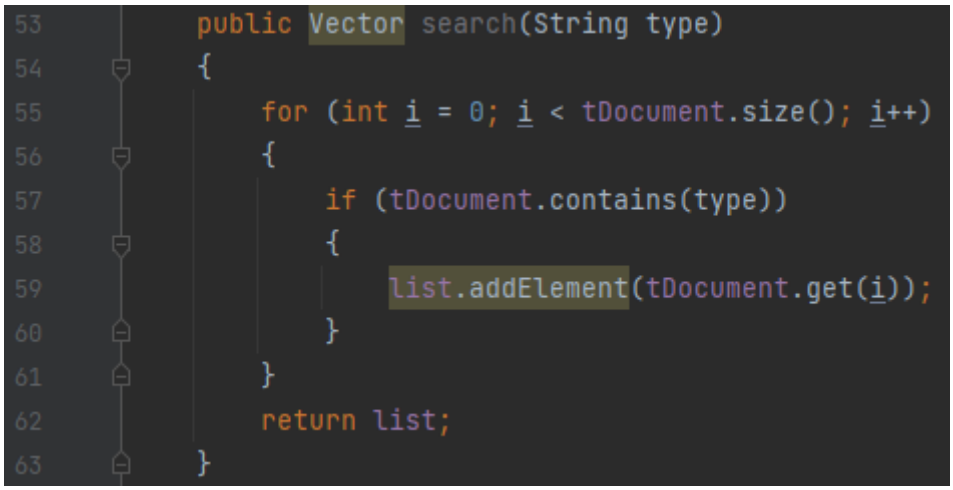
```
37      // ---------- Adders ----------  ----------
38      public void ajouterDocument(Document document) {
39          this.tDocument.add(document);
40          this.nbDocuments += 1;
41      }
```

La fonction ajouterDocument permet d'augmenter le contenu de tDocument.



```
43      // ---------- Methods ----------  ----------
44      public String toString()
45      {
46          this.document();
47          return "Bibliotheque { " +
48                  ", nbDocuments=" + nbDocuments +
49                  "\n, nom ='" + nom + '\'' +
50                  " }";
51      }
```

La fonction toString affiche toutes les informations sous forme de String.



```
53      public Vector search(String type)
54      {
55          for (int i = 0; i < tDocument.size(); i++)
56          {
57              if (tDocument.contains(type))
58              {
59                  list.addElement(tDocument.get(i));
60              }
61          }
62          return list;
63      }
```

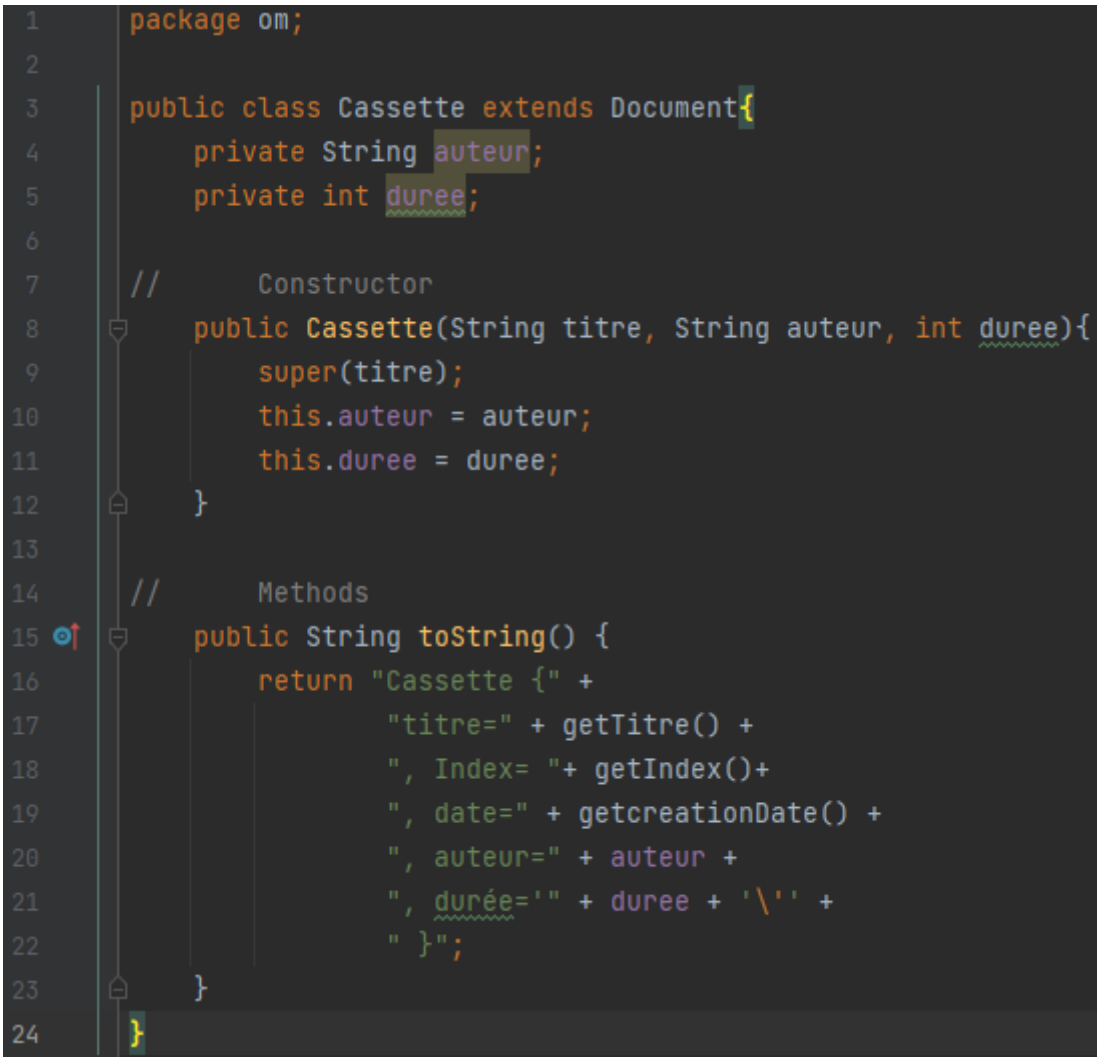
La fonction Search recherche les occurences.

Periodique.java

```java
package om;

public class Periodique extends Document{
    private String frequence;
    private int page;

//      Constructor

    public Periodique(String titre, String frequence, int page){
        super(titre);
        this.frequence = frequence;
        this.page = page;
    }

//      Methods
    public String toString() {
        return "Périodique { " +
                "titre=" + getTitre() +
                ", Index= "+ getIndex()+
                ", date=" + getcreationDate() +
                ", frequence=" + frequence +
                ", page='" + page + '\'' +
                " }";
    }
}
```

Ce fichier permet de définir les attributs d'un périodique.

Cassette.java



```java
package om;

public class Cassette extends Document{
    private String auteur;
    private int duree;

//      Constructor
    public Cassette(String titre, String auteur, int duree){
        super(titre);
        this.auteur = auteur;
        this.duree = duree;
    }

//      Methods
    public String toString() {
        return "Cassette {" +
                "titre=" + getTitre() +
                ", Index= "+ getIndex()+
                ", date=" + getcreationDate() +
                ", auteur=" + auteur +
                ", durée='" + duree + '\'' +
                " }";
    }
}
```

Ce fichier permet de définir les attributs d'une cassette.

Livre.java

```java
package om;

public class Livre extends Document{
    private int page;
    private String auteur;
    private String editeur;

//      Constructor
    public Livre(String titre, int page, String auteur, String editeur){
        super(titre);
        this.page = page;
        this.auteur = auteur;
        this.editeur = editeur;
    }

//      Methods
    public String toString() {
        return "Livre {" +
                "titre=" + getTitre() +
                ", Index= "+ getIndex()+
                ", date=" + getcreationDate() +
                ", page=" + page +
                ", auteur='" + auteur + '\'' +
                ", éditeur='" + editeur + '\'' +
                " }";
    }
}
```

Ce fichier permet de définir les attributs d'un livre.

Document.java

```java
package om;

import java.util.Date;
import java.util.Enumeration;
import java.util.Vector;

public abstract class Document {
    private String titre;
    private Date creationDate;
    private Vector index = new Vector();

//      constructor
    protected Document(){
        this.titre = null;
        this.creationDate = new Date();
    }
    protected Document(String titre) {
        this.titre = titre;
        this.creationDate = new Date();
    }
```

Autre que le package, les imports et les déclarations d'attributs, on peut ici voir une surchage du constructor.

```java
//      Getters
    public String getTitre() { return titre; }
    public String getcreationDate() { return ""+creationDate; }
    public Vector getIndex() {return index;}
```

Les getters retournent les attributs dont le dev aura besoin.

```java
//      Setters
    public void setTitre(String titre) { this.titre = titre; }
    public void setIndex(String type) { this.index.add(type);}
```

Les setters modifient les attributs de la classe.

```
36        //      Methods
37        public String toString() {
38            return "Document { Titre= "+getTitre()+
39                    ", Date de création= "+ getcreationDate()+
40                    " }";
41        }
```

La méthode toString renvoi les informations sous forme d'une String.

Ce que ce projet m’a apporté :

Ce TP m'a aider à mieux comprendre l'utilisation d'objets, les packages, les modèles UML et être plus fluide dans ma façon de coder en Java. Pour la première fois depuis le début de mon apprentissage dans ce langage, je me sentais à peu près à l'aise et le code venait presque instinctivement.

Ce qui pourrait être amélioré :

- Je n'ai pas encore bien intégré la fonction toString ni le type Enumeration et son utilisation.
- Mon code manque encore de finesse.
- La mise en place d'une Base de Données pourrait être intéressante.