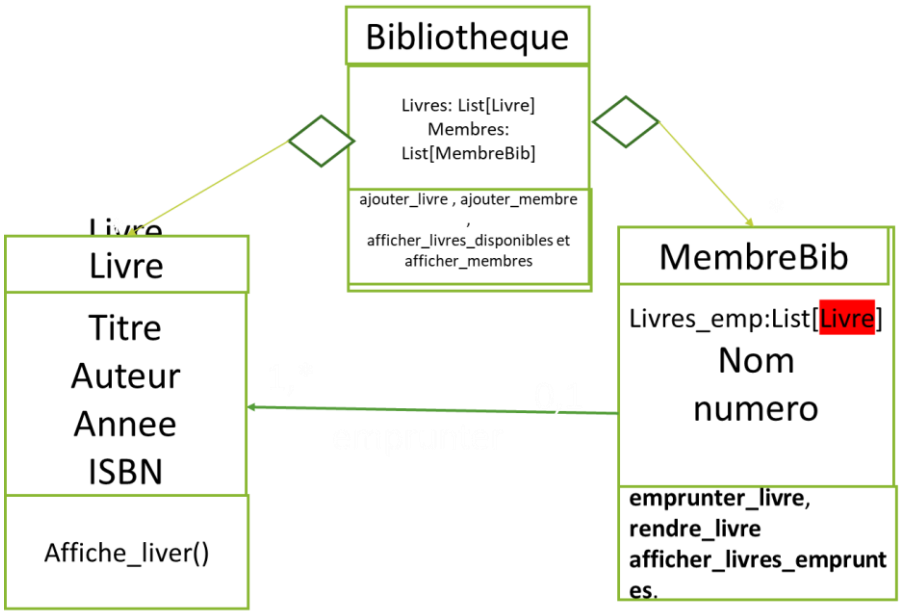
Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

# Atelier 1 : Python OO : les classes

## A. Exercice : Gestion d'une bibliothèque en utilisant des classes Python




### 1. Introduction :

Dans cet exercice, nous allons simuler la gestion d'une bibliothèque en utilisant les principes de la programmation orientée objet en Python. Vous serez amené à créer des classes pour modéliser les entités telles que les livres, les membres de la bibliothèque, et la bibliothèque elle-même. Chaque classe aura des attributs spécifiques et des méthodes pour effectuer des actions telles que l'emprunt et le retour de livres, l'ajout de nouveaux livres à la bibliothèque, et bien d'autres.

### 2. Objectifs :

- Comprendre la création de classes en Python.
- Mettre en œuvre des méthodes pour interagir avec les instances de classe.

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

- Utiliser des attributs pour stocker des informations spécifiques aux objets.
- Simuler des actions de gestion de bibliothèque, telles que l'emprunt, le retour, l'ajout de membres et de livres.

## 3. Enoncée

## Créer un notebook avec jupyter-lab ou Google Colab contenant les réponses aux exercices

1. **Classe Livre :**
    - Définissez une classe Python appelée **Livre** avec les attributs suivants : titre, auteur, annee, isbn.
    - Ajoutez une méthode **afficher_details** qui affiche les détails du livre.
2. **Création d'une instance de Livre :**
    - Créez une instance de la classe **Livre** avec les détails suivants : titre - "Harry Potter", auteur - "J.K. Rowling", année - 2005, ISBN - "9780545010221".
3. **Affichage des détails du livre :**
    - Appelez la méthode **afficher_details** sur l'instance créée pour afficher les détails du livre.
4. **Classe MembreBibliotheque :**
    - Définissez une classe Python appelée **MembreBibliotheque** avec les attributs suivants : nom, numero_membre, livres_empruntes (une liste).
    - Ajoutez les méthodes suivantes : **emprunter_livre**, **rendre_livre** et **afficher_livres_empruntes**.
5. **Création d'une instance de MembreBibliotheque :**
    - Créez une instance de la classe **MembreBibliotheque** avec les détails suivants : nom - "Alice", numero_membre - "123".
6. **Emprunt de livre par un membre :**
    - Appelez la méthode **emprunter_livre** sur l'instance du membre pour emprunter le livre créé précédemment.
7. **Affichage des livres empruntés par un membre :**
    - Appelez la méthode **afficher_livres_empruntes** pour afficher les livres empruntés par le membre.

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

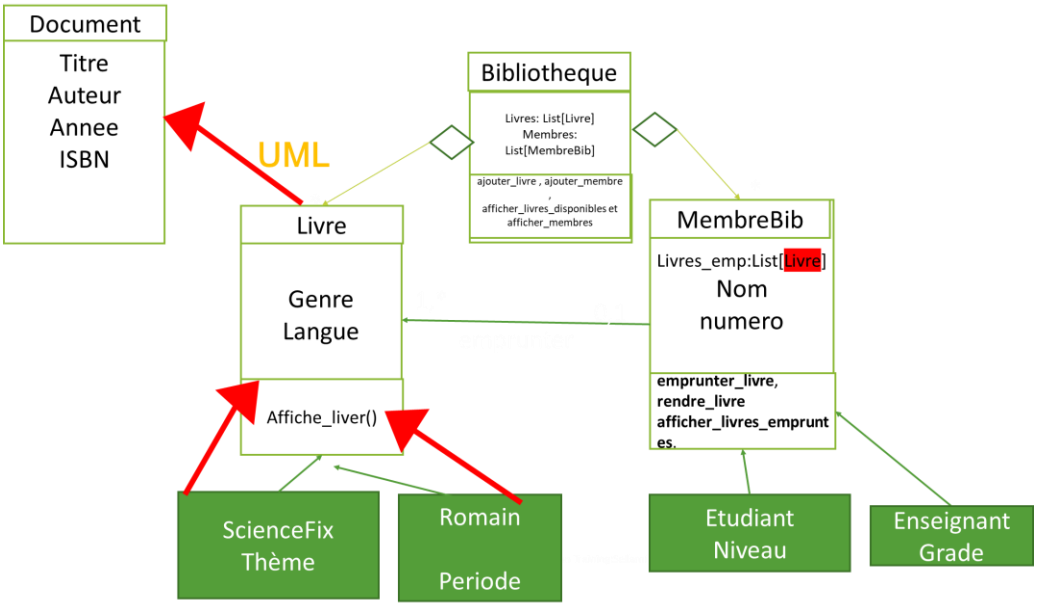
8. **Classe Bibliotheque :**
    - Définissez une classe Python appelée **Bibliotheque** avec les attributs : livres_disponibles (une liste) et membres (une liste).
    - Ajoutez les méthodes suivantes : **ajouter_livre**, **ajouter_membre**, **afficher_livres_disponibles** et **afficher_membres**.
9. **Ajout de livre à la bibliothèque :**
    - Créez une instance de la classe **Bibliotheque** et utilisez la méthode **ajouter_livre** pour ajouter le livre créé initialement.
10. **Ajout de membre à la bibliothèque :**
    - Utilisez la méthode **ajouter_membre** pour ajouter le membre créé précédemment à la bibliothèque.
11. **Affichage des livres disponibles :**
    - Utilisez la méthode **afficher_livres_disponibles** pour afficher la liste des livres disponibles dans la bibliothèque.
12. **Affichage des membres de la bibliothèque :**
    - Utilisez la méthode **afficher_membres** pour afficher la liste des membres de la bibliothèque.

**Note :** Complétez chaque étape en implémentant les classes et les méthodes nécessaires, sans fournir les réponses. Utilisez les instances créées pour tester votre implémentation à chaque étape.

## B. Exercice étendu : Gestion avancée d'une bibliothèque avec l'héritage

# Formation: Python Avancée

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com




Considérons une gestion avancée de la bibliothèque en introduisant le concept d'héritage. Nous allons créer une classe de base **Document** et dériver la classe **Livre** de cette classe de base. La classe **Livre** servira de classe de base à d'autres sous-classes spécifiques aux types de livres, par exemple, **Roman** et **ScienceFiction**.

1. **Classe Document :**
    - Créez une classe **Document** avec les attributs suivants : **titre**, **auteur**, **annee**, **isbn**.
    - Ajoutez une méthode **afficher_details** qui affiche les détails généraux du document.
2. **Classe Livre (héritage) :**
    - Héritez de la classe **Document** pour créer la classe **Livre**.
    - Ajoutez des attributs spécifiques aux livres, tels que **genre** et **langue**.
    - Ajoutez une méthode **afficher_details** dans la classe **Livre** qui appelle la méthode de la classe de base pour afficher les détails généraux et ajoute les détails spécifiques aux livres.
3. **Sous-classes de Livre :**
    - Créez au moins deux sous-classes de la classe **Livre**, par exemple, **Roman** et **ScienceFiction**.
    - Ajoutez des attributs spécifiques à chaque sous-classe, par exemple, **periode** pour la classe **Roman** et **theme** pour la classe **ScienceFiction**.

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

- Ajoutez une méthode **afficher_details** pour chaque sous-classe qui appelle la méthode de la classe **Livre** et ajoute les détails spécifiques à chaque type de livre.

4. **Création d'instances de sous-classes :**
   - Créez des instances de chaque sous-classe de livre avec des détails spécifiques.
   - Appelez la méthode **afficher_details** pour afficher tous les détails de chaque type de livre.
5. **Extension de la gestion de la bibliothèque :**
   - Modifiez les fonctions existantes pour utiliser les classes dérivées de **Livre** au lieu du dictionnaire pour représenter les livres.