TD - Programmation Orientée Objet (POO), Relations UML et Diagrammes de Classe

ali.zainoul.az@gmail.com

19 juin 2025

Objectifs du TD

Ce TD vise à approfondir la programmation orientée objet (POO) en Python, à modéliser les relations UML entre classes, et à maîtriser la conception de diagrammes de classes à partir de projets.

Partie 1 : Fondamentaux de la POO en Python

- Définir une classe et instancier des objets
- Constructeur `__init__`, attributs d'instance et de classe
- Méthodes : accesseurs, mutateurs, méthodes spéciales `__str__`, `__repr__`, etc.
- Encapsulation et propriétés
- Héritage simple et redéfinition de méthode

Exercice 1 : Classe de base

Créer une classe `Book` avec les attributs `title`, `author`, `year`. Ajouter une méthode `display()`.

Exercice 2 : Héritage simple

Créer une classe `Novel` héritant de `Book` et ajoutant un attribut `genre`. Redéfinir `__str__()`.

Exercice 3 : Attributs privés et propriétés

Rendre les attributs de `Book` privés. Ajouter des accesseurs/mutateurs avec `property()`.

Partie 2 : Relations UML

- Association
- Agrégation (relation "a un", mais les objets vivent indépendamment)
- Composition (relation forte, le cycle de vie est lié)

Exercice 4 : Agrégation

Créer une classe `Library` contenant une liste de `Book`. Ajouter des méthodes `add()`, `remove()`, `display()`.

Exercice 5 : Composition

Créer une classe `Page` et modifier `Book` pour qu'elle contienne une liste de `Page`. Implémenter une méthode `add_page()`.

Partie 3 : Diagrammes de classes UML

— Règles de modélisation UML
— Nom des classes, attributs, méthodes
— Visibilité (`+`, `-`, `#`)
— Relations : flèches d'héritage, agrégation, composition

Exercice 6 : UML à partir de code

Analyser le code de `Book`, `Page`, `Library` et dessiner le diagramme UML correspondant.

Exercice 7 : UML vers code Python

Un diagramme UML est donné avec les classes `Student`, `Course`, `Grade`. Écrire les classes Python correspondantes.

Exercice 8 : Multiplicité et associations

Créer une relation `Teacher` <–> `Course`. Un enseignant peut enseigner plusieurs cours. Implémenter les classes et produire le diagramme UML.

Exercice 9 : Classe abstraite et polymorphisme

Créer une classe abstraite `Document` (avec le module `abc`) et deux classes filles `PDF` et `Word`. Ajouter une méthode abstraite `render()`.

Partie 4 : Projet orienté objet et UML complet

Créer une application orientée objet permettant de gérer une plateforme d'apprentissage avec les classes suivantes : `User`, `Course`, `Lesson`, `Quiz`, `Question`, `Answer`.

— Implémenter les classes Python avec les bonnes relations (composition, ...).
— Utiliser les propriétés, méthodes spéciales et polymorphisme.
— Créer un diagramme de classes UML complet pour ce projet.