POO

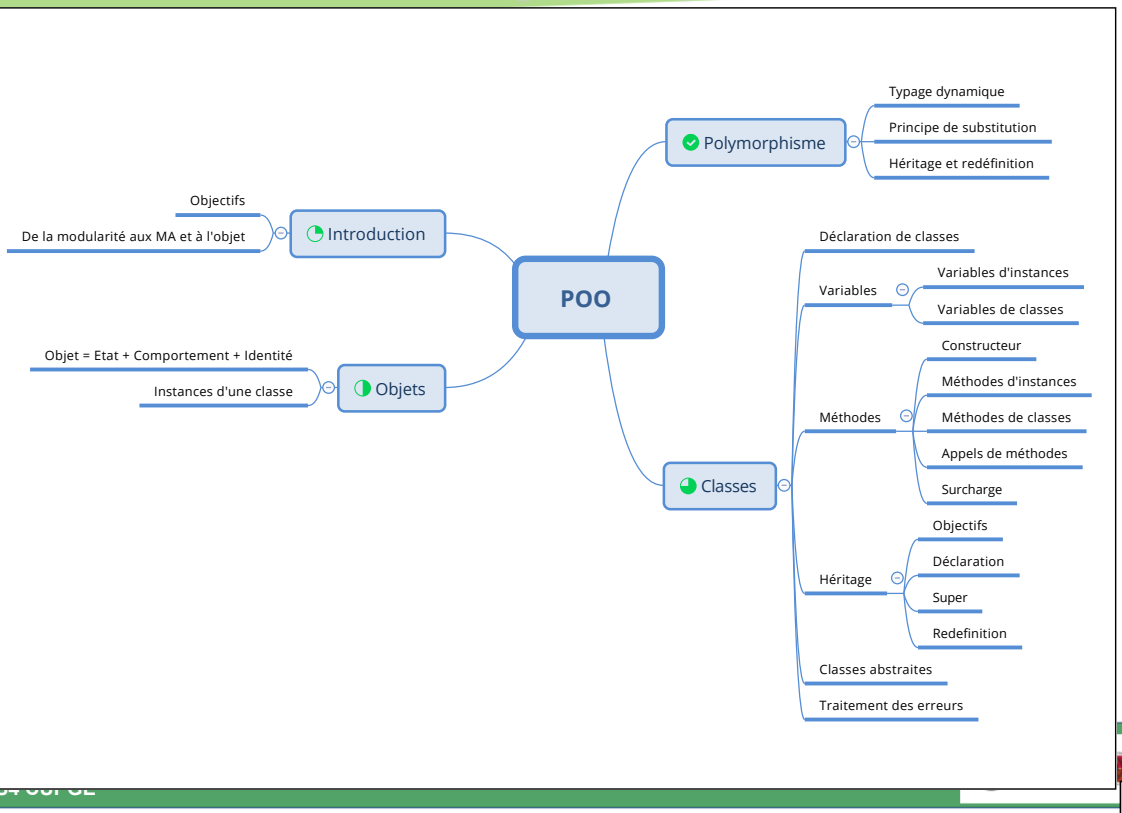
Programmation Orientée Objet
Polymorphisme

UNIVERSITÉ TOULOUSE III PAUL SABATIER

Université de Toulouse

1

POO : Polymorphisme



2

2

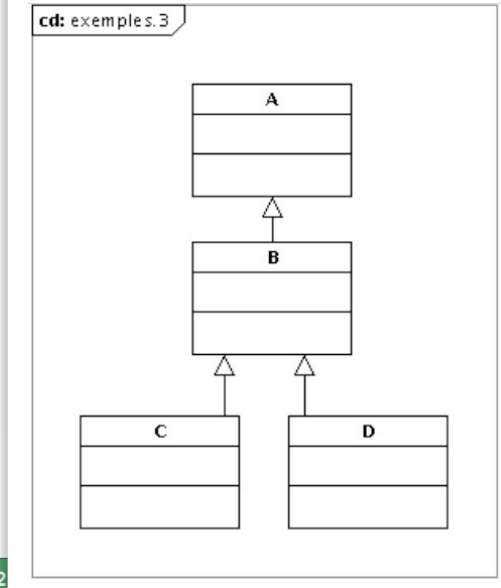
2_POOPolymorphisme.key - 3 mai 2023

Triptyque :

- **Polymorphisme**
  - **Le typage dynamique**
  - **L'héritage**
  - **La redéfinition**

Typage dynamique

- **Règle :**
  - **La référence d'une instance d'une classe héritière peut être affectée à toute instance d'une classe parente**

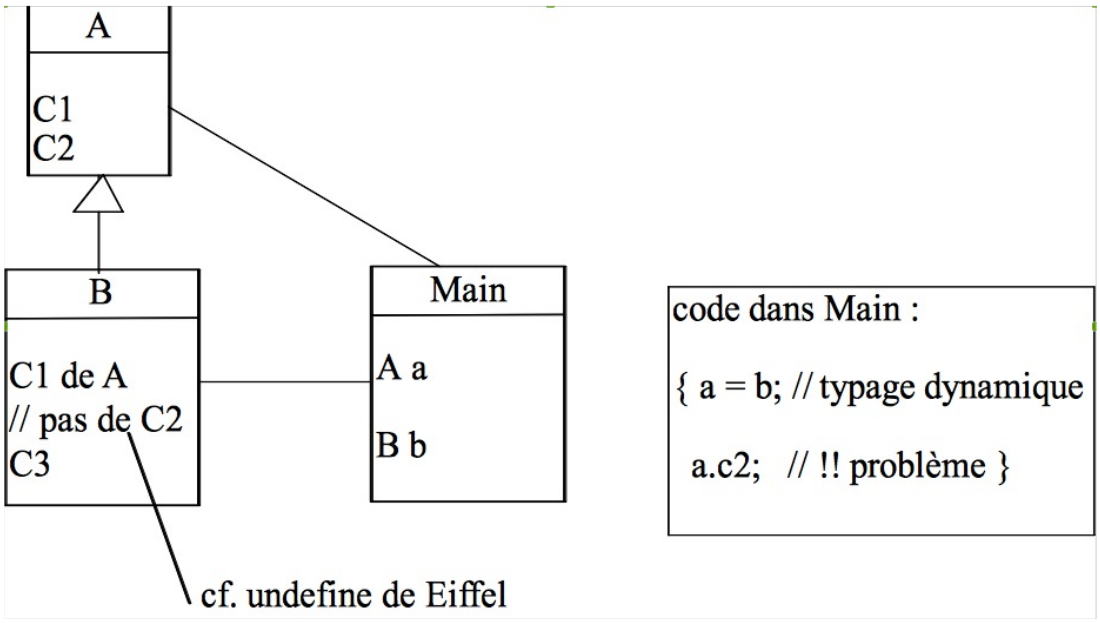



```
b = B()
a = b
print("1 a : "+a.methode())
c = C()
a = c
print("2 a : "+a.methode())
b = c
print("3 b : "+b.methode())
d = D()
a = d
print("4 a : "+a.methode())
b = d
print("5 b : "+b.methode())
print("6 d : "+d.methode())
c = d # illegal en java
print("7 c : "+c.methode())
b = c
d = b # illegal en java
print("8 d : "+d.methode())
a = e
b = a # illegal en java
print("9 b : "+b.methode())
```

Principe de substitution

- **Règle :**
  - **Il doit être possible de substituer n'importe quel objet instance d'une sous-classe à n'importe quel objet instance d'une super-classe sans que la sémantique du programme écrit dans les termes de la super-classe ne soit affectée**
  - **La classification propage l'état, le comportement et les contraintes**

Principe de substitution



- **Q : Comment respecter le principe de substitution ?**

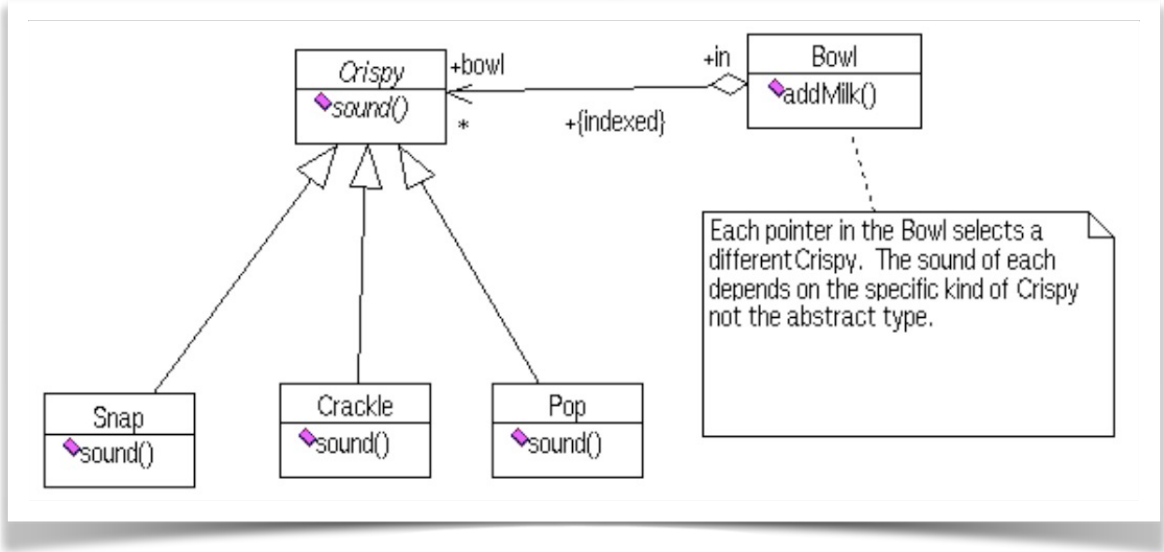
Principe de substitution

- Règle de substitution respectée
  - Tout ce qui est basé sur le comportement de A', fonctionnera sur A et sur B

Exemple

- Un bol contient des céréales de type "Crispy".
  - Snap
  - Crackle
  - Pop
- La méthode addMilk() "verse" du lait sur les céréales
- Chaque type de céréale réagit en produisant un son, méthode sound()
- Q : Comment modéliser ce pb pour mettre en oeuvre le polymorphisme ?

Solution



Exemple

❑ So

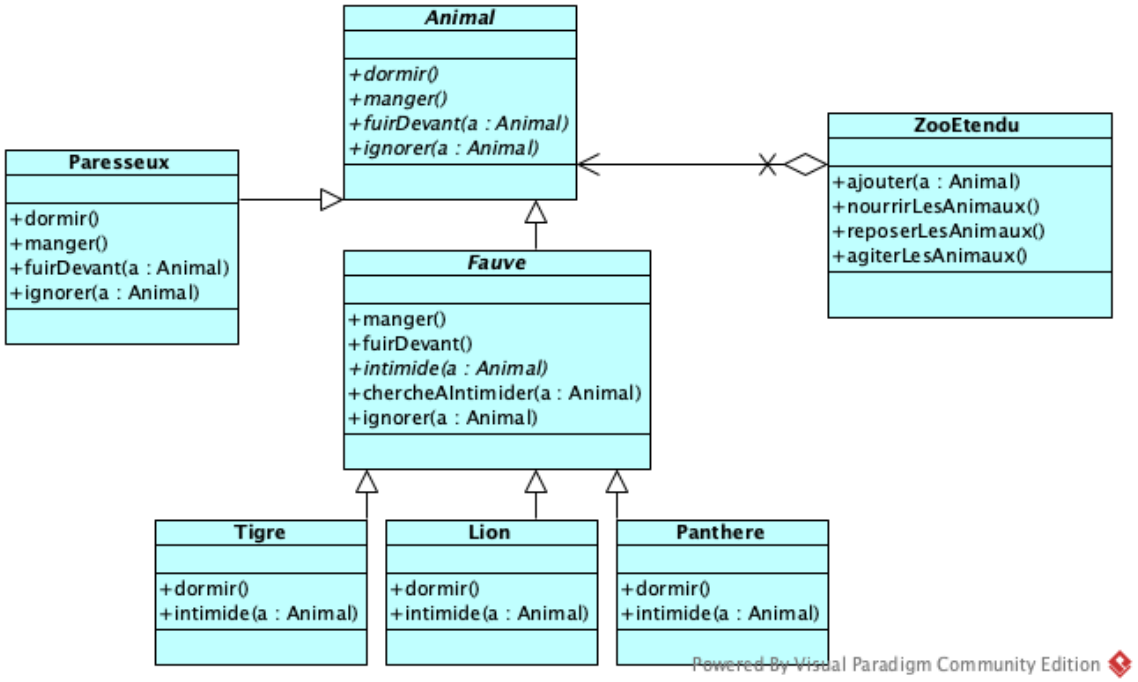
```
from abc import ABC, abstractmethod
class Crispy(ABC): # hérite de Abstract Base Class
    @abstractmethod
    def sound(self):
        return
class Snap(Crispy):
    def sound(self):
        print("snap", end=' ')
class Pop(Crispy):
    def sound(self):
        print("pop", end=' ')
class Crakle(Crispy):
    def sound(self):
        print("crack", end=' ')
class Bolw:
    def __init__(self):
        self.__cereales = []

    def add(self, cereale : Crispy):
        self.__cereales.append(cereale)
    def addMilk(self):
        for c in self.__cereales:
            c.sound()
```

Exercice

- **Zoo**
  - **Modéliser un zoo qui contient des fauves**
    - Lion
    - Panthère
    - Tigre
  - **Chaque fauve mange de la viande**
  - **Chacun dort différemment**
    - Le lion dort sur la colline (c'est le roi !)
    - La panthère dort dans un arbre
    - Le tigre dort dans la savane
- **Q : comment ajouter un paresseux qui mange des feuilles et qui dort tête en bas ?**

Zoo étendu



Récapitulatif

- **Polymorphisme**

  - **Héritage**
  - **Typage dynamique**
    - Avec principe de substitution
  - **Redéfinition**