Matière : Programmation orientée objet

Date : 12.04.2022

Formateur : Paul-Henri Martelloni

Mail : martelloni.paulhenri@orange.fr

Auteur : Paul-Henri Martelloni

AFLOKKAT
Conseil et Formation

La programmation orientée objet

SOMMAIRE

AFLOKKAT
Conseil et Formation

AFLOKKAT
Conseil et Formation

01 Introduction

AFLOKKAT
Conseil et Formation

Introduction

La programmation procédurale

- Nous connaissons la programmation « procédurale » :
  - Des variables, des fonctions, des instructions qui s'exécutent dans un ordre particulier et manipulent des données facilement accessibles et modifiables pour résoudre un problème
- En programmation procédurale, nous découpons notre programme en fonction et procédures qui résolvent chacune une petite partie du problème
- Le programme dans son ensemble est donc découpé selon les sous problèmes que nous avons identifié
- La programmation orientée objet (POO) nous propose une autre façon de penser nos programmes.

AFLOKKAT
Conseil et Formation

Introduction

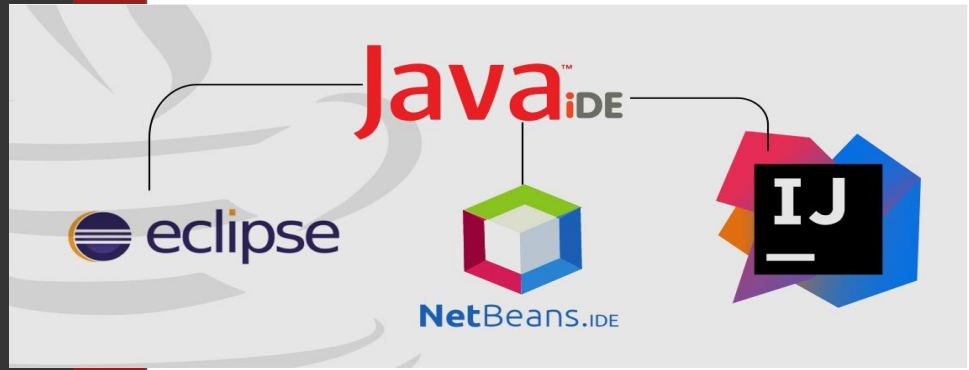
La programmation orientée objet (POO)

- En programmation orientée objet, nous allons créer et faire interagir des **objets**, des briques logiciel qui seront représentative de données que nous allons manipuler.
- En POO presque tout peut être vu comme un objet. Un ordinateur, une personne, un concept, toute entité que nous allons vouloir traiter.
- les objets de la POO sont proches de celles du monde réel. Les concepts utilisés sont donc proches des abstractions familières que nous exploitons.
- Pourquoi cette façon de penser est elle intéressante?
    - La structure d'objet en interaction donne une structure de code souvent plus claire
    - Plus modulable et maintenable
    - Plus de facilité à debugger

Introduction

La programmation orientée objet (POO)

- La plupart des langages permettent de faire de la POO
  - Java, python, php, ruby, C++, C#, VB.net, etc.
- Nous utiliseront java pour mettre en œuvre ces concepts
- Plusieurs IDE se proposent à vous



https://www.mindfiresolutions.com/blog/2018/05/best-java-ides-2018/

Introduction

Objectifs de ce cours

- Présenter les bases de la POO, et une nouvelle manière de penser nos programmes
- Utiliser un nouveau langage, java qui sera assez différent de python
- Découvrir le potentiel que peut apporter cette approche
- Nous n'aurons ni le temps ni la prétention de faire le tour du sujet dans ce cours, il y aurait beaucoup trop de chose a dire sur la programmation orientée objet pour la traiter entièrement

AFLOKKAT
Conseil et Formation

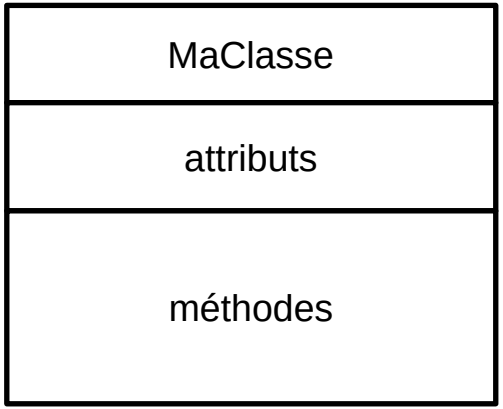
02 La notion de classe

AFLOKKAT
Conseil et Formation

La notion de classe

- Classes et objets
- L'encapsulation
- La programmation de tout cela

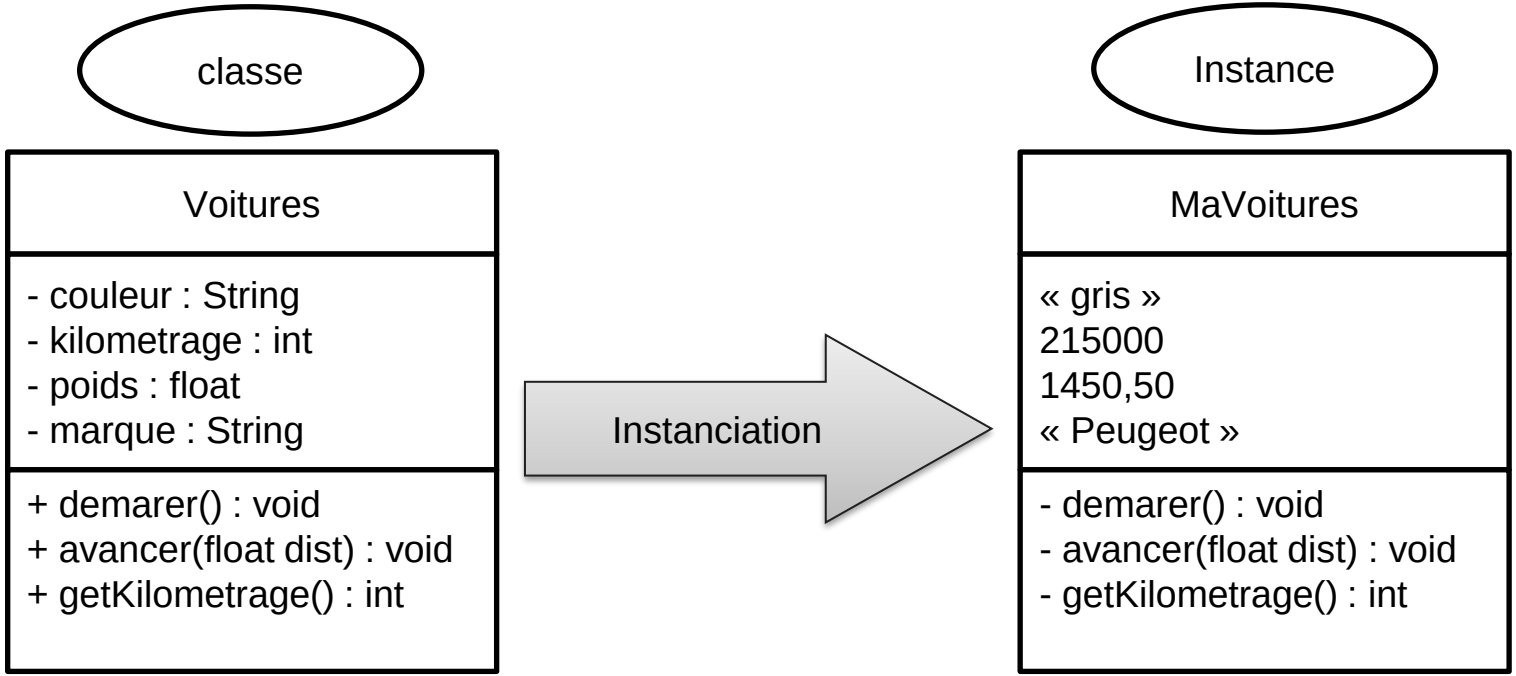
Classes et objets



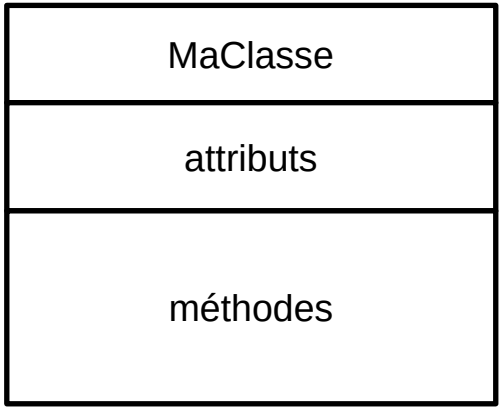
- Une classe est une sorte de plan de construction d'un type d'objet
- Une classe à :
  - Un **nom** que nous considèrerons comme un **type** au sens programmation du terme
  - Des **attributs**, des variables internes à l'objet
  - Des **méthodes**, qui sont des fonctions permettant de manipuler l'objet
- Une **instance** sera un objet créé à partir de ce plan
- Nous pouvons créer et gérer autant d'intances que nous voulons

Classes et objets

- Exemple de classe et d'objet



Classes et objets



Les méthodes

- Des fonctions internes à la classe permettant de la manipuler

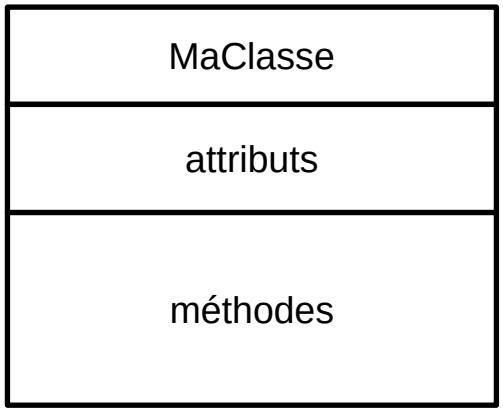
Notion de **signature** :

- Une méthode est caractérisée par sa **signature.**
- Elle permet d'identifier sans ambiguïté la méthode à exécuter

Elle comprend **le nom de la méthode, du nombre et du type de ses paramètres, et du type qu'elle renvoie**

2 méthodes peuvent avoir le même nom mais se différencient par leur signature complète

Classes et objets



Des méthodes particulières :

- Les constructeurs :
  - Définit comme des méthodes, ont le même nom que la classe (en java), n'ont pas de type de retour (c'est les seules)
  - Permettent de créer les instances
  - Nous pouvons en définir plusieurs (leur signature les différencient)
  - Le mot clef **new** (en java) nous permettra d'appeler les constructeur à partir du programme principal
- Les accesseurs :
  - Permettent d'accéder aux attributs
  - « getteurs » permettent de lire les attribut
  - « setters » permettent de modifier les attributs

Classes et objets

AFLOKKAT
Conseil et Formation

Méthodes et attributs d'instance

- Par défaut
- Ce sont les méthodes et attribut dont nous avons parlé jusqu'à maintenant.
- Ces méthodes et attributs sont liés à une instance, à un objet concret.

Utilisation: objet.nomMethode()

maVoiture.poids

maVoiture.demarer()

Méthodes et attributs de classe

Le mot clef « static » :

- Permet de définir des attribut ou des méthodes de classe.
- Liés à la classe et non à une instance en particulier
- Cela peut permettre de gérer toutes les instance d'une même classe

Utilisation: NomClasse.nomMethode()

Voiture.getNbVoiture()

AFLOKKAT
Conseil et Formation

Notion d'encapsulation

Notion de droit d'accès

- *En POO les élément d'un objet ne sont pas forcement accessible librement*
- *Les accesseur permettent à une classe de contrôler la lecture et la modification de ses attributs*
- *les attributs comme les méthodes ont des droit d'accès:*
  - ***Public** : librement accessible par n'importe qui, peut être appelé partout dans le programme (représenté par un **+** dans les diagramme)*
  - ***Protected** : accessible uniquement à partir d'une classe dérivée (nous verrons ultérieurement ce que cela signifie) (représenté par un **#** dans les diagramme)*
  - ***Private** : accessible uniquement à l'intérieur de la classe, ne pourra pas être utilisé ailleurs dans le programme (représenté par un **-** dans les diagramme)*

*Ex: un attribut « private » ne sera accessible que par les méthode **get** et **set** (accesseurs)*

AFLOKKAT
Conseil et Formation

Généralités

Nous retrouverons encore plus clairement qu'en programmation procédurale la séparation entre programme principal et implémentation des classes

Les conventions :

- *Les nom de classe commencent par une majuscule*
- *Les nom d'attribut ou de méthode comment par une minuscule*
- *Les attributs sont généralement private*
- *Les méthodes sont généralement public*

AFLOKKAT
Conseil et Formation

La programmation de tout cela

- *Bon ok, on a des moules, les classes, on peut créer des objets, ces objets on des méthodes et des attributs qui leur permettent d'évoluer et d'interagir.*
- *Mais concrètement ca donne quoi tout ca ?*
- Reprenons tout ce que nous avons vu sur des exemples ca sera plus parlant
- Pour cela nous allons appliquer les recommandation de l'algorithmique.
    1. Conception
    2. Implémentation
    3. test

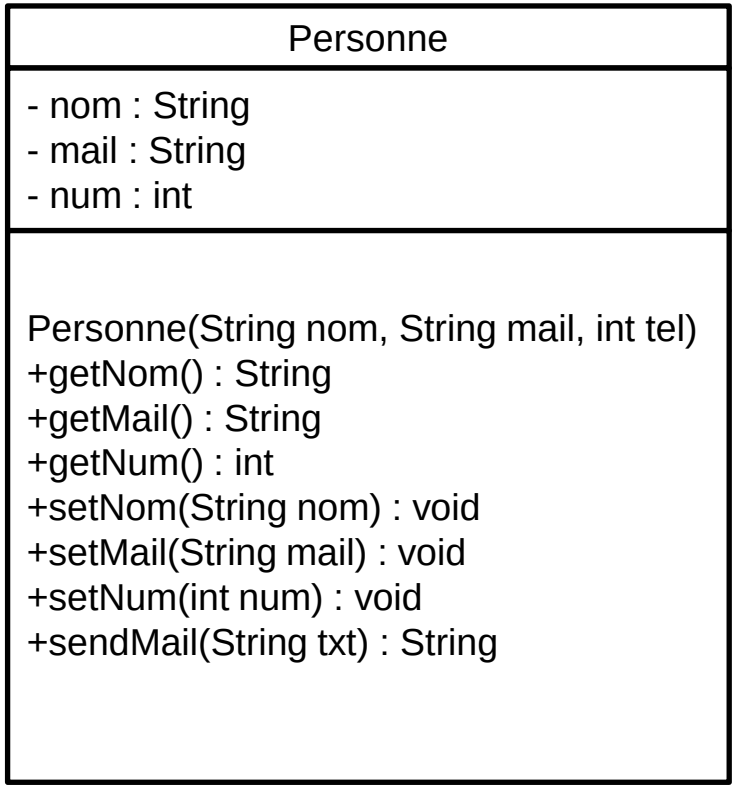
AFLOKKAT
Conseil et Formation

Un premier exemple

- *Posons nous un cas très simple, nous voulons gérer des fiches de contact*
- *Un contact peut avoir un nom, un mail, un numéro de téléphone*
- *Nous voulons des méthodes permettant de gérer les contacts (constructeurs, accesseurs) ainsi que des méthodes permettant de les contactés (nous ne ferons que des affichages dans la console pour symboliser cela)*
- *Nous créerons ensuite 4 personnes :*
  - *Bruno*
  - *Nicolas*
  - *Paul*
  - *Antoine*

AFLOKKAT
Conseil et Formation

Un premier exemple



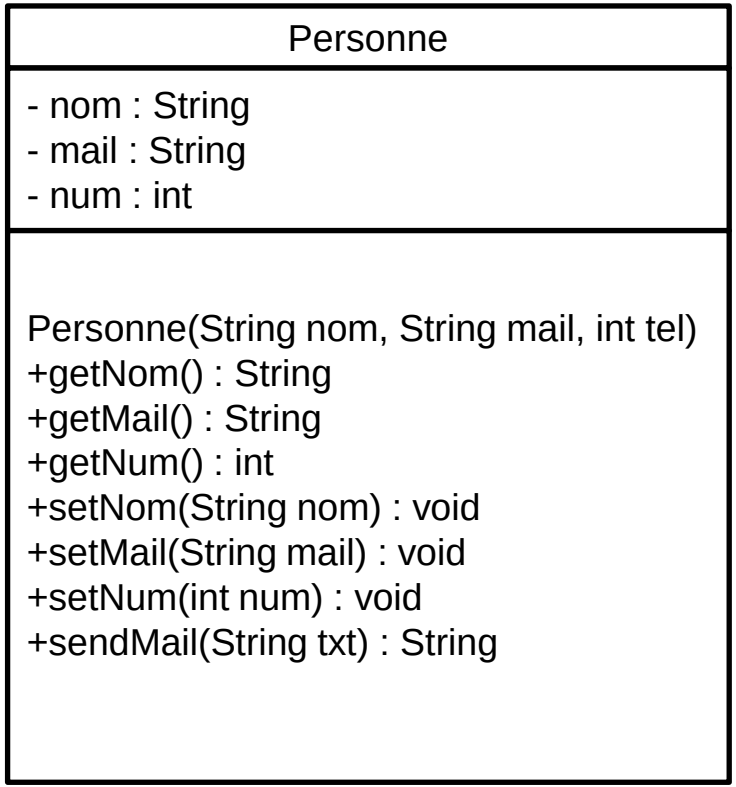
- Dans le programme principal nous allons devoir créer nos personnes et interagir avec.
- Bien nous savons comment nous allons construire notre programme
- Passons à l'implémentation en java

AFLOKKAT
Conseil et Formation

Java

- Entre la compilation et l'interprétation
  - Compilé en pseudo code ou bytecode qui sera interprété par la machine virtuelle java (JRE)
- Portable, il suffit d'avoir la machine virtuelle java
  - slogan de Sun pour Java : " WORA (Write Once, Run Anywhere"
- Fortement type et typage statique
  - Toutes les variables doivent etre **typées**, **aucune conversion** de type **automatique**
- Créé et maintenu par Sun, utilisé par des millions d'applications (android, web, utilisaire etc.)

Un premier exemple

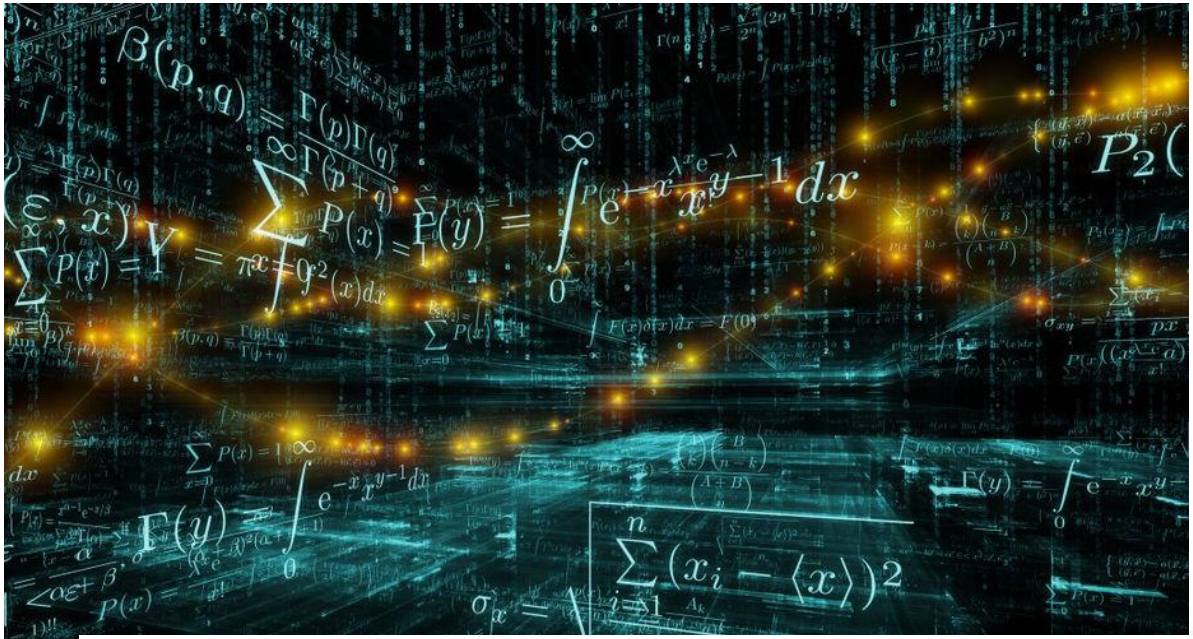


- Passons sur l'IDE pour se lacer dans le code Java

AFLOKKAT
Conseil et Formation

03 L'héritage

AFLOKKAT
Conseil et Formation



04

Conclusion

AFLOKKAT
Conseil et Formation

Conclusion

Nous avons vu les notions de base de l'algorithmique et mis en pratique des exemples en utilisant python.

- Des **variables** qui sont le cœur même de nos algorithmes
- Des **conditions** qui permettent de différencier des situations
- Des **boucles** qui permettent de répéter des instructions
- Des **fonctions** et procédures qui permettent d'appeler des morceaux d'algorithmes

Nous allons maintenant pouvoir nous servir de tout cela pour faire tout ce que nous voulons

Testons ce que nous avons appris en essayant de faire des petit jeux en python

Bonnes pratiques en python (cours de python)

Quelques notes:
Quand vous avez des erreurs, lisez bien les retours, la réponse est souvent dedans.
C'est bien de réutiliser des choses déjà faites, mais seulement si on les comprends

AFLOKKAT
Conseil et Formation

MERCI DE VOTRE ATTENTION !

Quelques icônes pour aider

AFLOKKAT
Conseil et Formation