Java Standard Edition

La programmation Orienté Objet en Java.

PARTIE I

- Introduction à la POO
- Introduction à Java SE

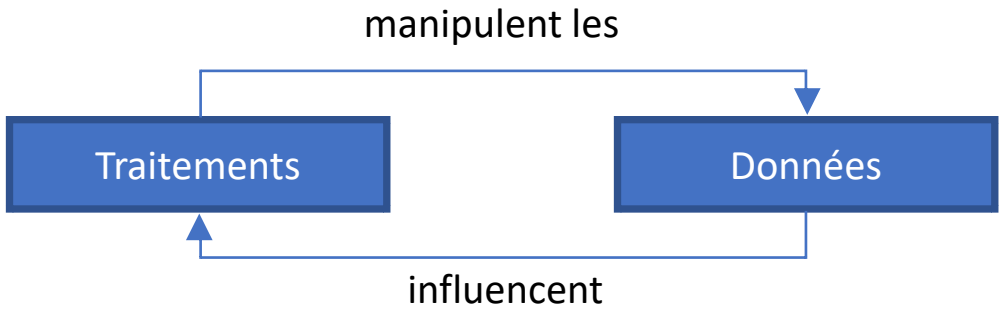
INTRODUCTION A LA POO

Aujourd'hui le courant objet est une réalité incontournable dans le monde des méthodes de conception et de développement des Systèmes d'Information (SI).

L'objectif de cette présentation est d'essayer d'éclairer au mieux ce que l'approche objet peut apporter au monde des SI

L’approche procédurale - rappel

- Programmation procédurale
  - Dans les programmes écrits selon l’approche dite procédurale, les notions de variables, types de **données** et de **traitement** de ces données étaient séparées :



  - On s'intéresse à écrire les étapes séquentielles nécessaires pour résoudre un problème. L'accent est mis sur les étapes pour réaliser une tâche.

L’approche procédurale - rappel

- Programmation procédurale
  - Cette approche fonctionnelle n'est pas adaptée au développement d'applications qui évoluent sans cesse et dont la complexité croit continuellement.
  - Elle met en évidence un évident problème de réutilisation des solutions déjà implémentées.

L’approche procédurale - rappel

- Programmation procédurale - Principe
  - Centrée autour de la définition de routines (procédures ou fonctions).
  - Utilise une approche algorithmique (séquence d’instructions ordonnées pour exécuter des opérations désirées).
  - Structures de données et routines difficilement réutilisables
  - Difficultés de maintenance (corrective et évolutive)

L’approche **objet**

- Le principe
  - On s'intéresse à modéliser le problème par un ensemble d'objets.
  - L'accent est mis sur les objets requis pour résoudre un problème.
  - Le programme est l’ensemble des objets et des interactions entre ces objets
  - En programmation orientée objet, on cherche à ***identifier les objets*** impliqués et leurs ***responsabilités respectives.***

L'approche **objet**

- Le concept de base – l'objet
  - Un **objet** est une représentation (*abstraite*) d'une entité (*concrète*) du monde réel qui se caractérise par :
    - son **état (**ses données**)** et
    - son **comportement (**ses actions**).**
  - Objet du monde réel
    - Une *personne*, une *voiture*, un *point*, une *figure*, une *commande*, un *client*, un *produit*, etc …
  - Objet informatique (Modélisation - Pilotage - Simulation)
    - Entité atomique formée de l'union d'un état et d'un comportement

L’approche **objet**

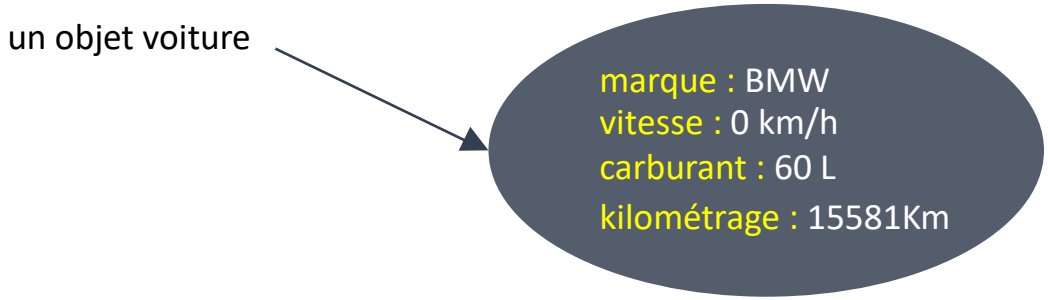
- Le cycle de vie de l’objet informatique est similaire à celui du monde réel.

- En effet, comme les objets du monde réel, les objets informatiques **naissent**, **vivent** et **meurent**.

L’approche **objet**

- Les **caractéristiques** fondamentales d’un objet informatique

1. **Etat** = Ensemble des valeurs de ses attributs à un instant donnée
   - Un attribut est une information qui caractérise l'objet qui le contient
   - Exemple : l’état d’un objet **Voiture** regroupe les valeurs des attributs (vitesse, carburant, marque, kilométrage …)



L’approche **objet**

- Les **caractéristiques** fondamentales d’un objet informatique

2. **Comportement** = Regroupe toutes les compétences d'un objet.
    - Tout ce qu’il sait et peut faire
    - Correspond aux actions de l’objet sur les autres ou sur lui-même.
        - Chaque élément de son comportement est appelé opération ou méthode.
        - Les opérations d'un objet sont déclenchées suite à une stimulation externe, représentée sous la forme d'un message envoyé par un autre objet
        - L'état et le comportement des objets sont liés
        - Exemple : la voiture peut se deplacer >>> *rouler ()*

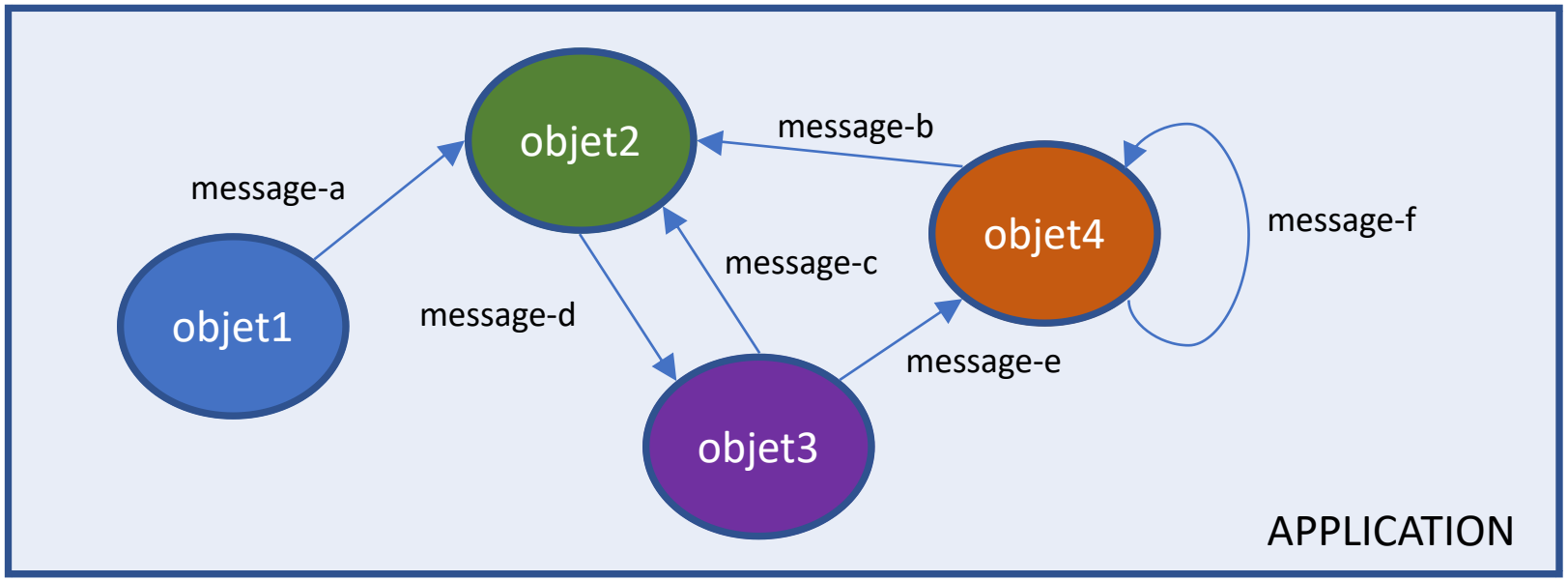
L’approche **objet**

- Les **caractéristiques** fondamentales d’un objet informatique

3. **Identité** = Elément qui caractérise son existence propre.
    - L'identité permet de distinguer tout objet de façon non ambiguë, indépendamment de son état.
    - Permet de distinguer deux objets dont toutes les valeurs d'attributs sont identiques
        - >>> deux personnes de même **nom**, **prénom**, **age** et **sexe** sont néanmoins deux objets distincts.

L’approche **objet**

- La **communication** entre objets – Le concept de **message**
  - Les programmes mettant en œuvre des objets peuvent être vus comme une équipe projet qui travaillent en collaboration afin de réaliser les fonctionnalités d’une l'application.



L’approche **objet**

- La **communication** entre objets – Le concept de **message**
  - L'unité de communication entre objets s'appelle le message.
  - Il existe quatre (4) catégories principales de messages
    - Les **constructeurs** qui créent des objets
    - Les **destructeurs** qui détruisent des objets
    - Les **accesseurs** qui renvoient tout ou partie de l'état de l'objet
    - Les **mutateurs** qui changent tout ou partie de l'état d'un objet

L’approche **objet**

- Les langages orientés objets – **Principe fondamental**

    1. *Permet une approche de la programmation un peu proche du **raisonnement humain** : on crée des modèles d'objets (les **classes**) qui regroupent des propriétés (**attributs**) et des méthodes (**actions**) qui leur sont spécifiques.*

L’approche **objet**

- Les langages orientés objets – **Principe fondamental**

    2. *Offre la possibilité de **factoriser** certaines opérations entre modèles de données similaires et d'avoir une organisation hiérarchique des concepts, du plus général (dans les classes parentes) au plus spécifique (dans les classes filles).*

    *(**Héritage** et **Polymorphisme**).*

L’approche **objet**

- Les langages orientés objets – **Principe fondamental**

    3. *La notion **d'encapsulation**, qui permet de paramétrer la portée, permet plus facilement d'isoler le code technique et de prévenir (dans une certaine mesure) l'appel de certains traitements dans un contexte inapproprié.*

L’approche **objet**

- Les langages orientés objets – **Principe fondamental**

  SIMULA (62),

  SmallTalk (69),

  C++ (82),

  Eiffel (85),

  Java (96),

  C# (2001)

L’approche **objet**

- Les langages orientés objets – **Objectifs**
  - Mise en œuvre de logiciels de meilleure qualité (garantissant une forte fiabilité, extensibilité et réutilisabilité).
  - Améliore la productivité des programmeurs (réduction des risques d’erreurs, respect des jalons de développement, …)
  - Améliore la lisibilité du code source du logiciel
  - Facilite ainsi sa maintenance
  - Réduit la durée des actions de maintenance (évolutive ou corrective)

L’approche **objet**

- La **Classe** – **Concept**
  - Sorte de moule à partir duquel sont générés des objets de même nature (même structure et même comportement).
  - Ces objets ainsi générés sont des instances de la classe de laquelle elles sont issues.
  - Un objet est une instance d’une et d’une seule classe.
  - La classe décrit donc la structure d’un élément

L’approche **objet**

- La **Classe** – **Concept**
  - L’objet représente un exemplaire créé sur le modèle de cette structure.
  - Après sa création, un objet est indépendant des autres objets construits à partir de la même classe.
  - Les classes sont constituées de champs et de méthodes.

L’approche **objet**

- La **Classe** – **Concept**
  - Les **champs** représentent les **caractéristiques des objets**.
    - Ils sont représentés par des **variables** et il est donc possible de lire leur contenu ou de leur affecter une valeur directement.
  - Les **méthodes** représentent les actions qu’un objet peut effectuer.
    - Elles sont mises en œuvre par la création de **procédures** ou de **fonctions** dans une classe.

L’approche **objet**

- En résumé, en POO, programmer revient donc à :
  - ***Décrire*** *des classes d’objets,*
  - ***Caractériser*** *leur structure et leur comportement, puis à*
  - ***Instancier*** *ces classes pour créer des objets réels et les manipuler.*

L’approche **objet**

[Classe]
Plan de construction

[objet a]
Instance de la classe

[objet b]
Instance de la classe

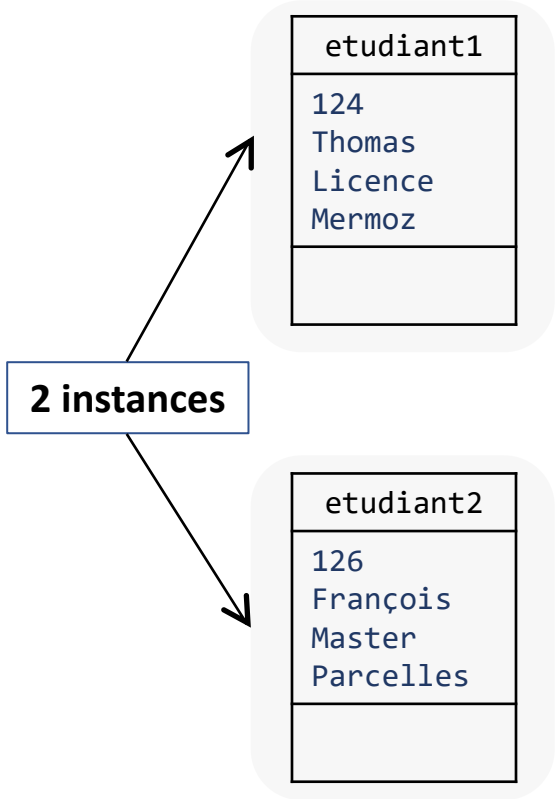
[Objet c]
Instance de la classe

L'approche **objet**

- Considérons la classe **Etudiant** qui représente un étudiant dans une école ou une université. La description de cette classe comportera les éléments suivants :

Nom de la classe	**Etudiant**
Attributs (données)	**Numéro** : int **Nom** : String **Formation** : String **Adresse** : String
Opérations (méthodes)	*inscrire ()* *composer ()* *passer ()* *redoubler ()*

La classe **Etudiant**



Les piliers de la POO

- L'encapsulation
  - L'objet forme un tout
  - L'objet a une nature (type) qui ne peut changer
  - L'objet est garant de son état
- L'héritage
  - L'objet peut être l'évolution d'un autre plus général
- Le polymorphisme
  - Des objets de nature différente peuvent réagir au même message

Les piliers de la POO

- L'encapsulation
  - L'encapsulation consiste à cacher les éléments qui ne sont pas nécessaires pour l'utilisation d'un objet garantissant ainsi que l'objet sera correctement utilisé.
  - Exemple :
    - Pour changer de vitesse, il n'est pas nécessaire de modifier directement la position des différents engrenages.
    - Les constructeurs ont en fait prévu des solutions plus pratiques pour la manipulation de la boîte de vitesse.

Les piliers de la POO

- L'encapsulation
  - Les éléments d'une classe visibles de l'extérieur de la classe sont appelés **l'interface de la classe**.
  - Dans le cas de notre voiture, le levier de vitesse constitue l'interface de la boîte de vitesse.
  - C'est par son intermédiaire que l'on peut agir sans risque sur les mécanismes internes de la boîte de vitesse.

Les piliers de la POO

Accès aux données via l'interface (partie visible de la classe)

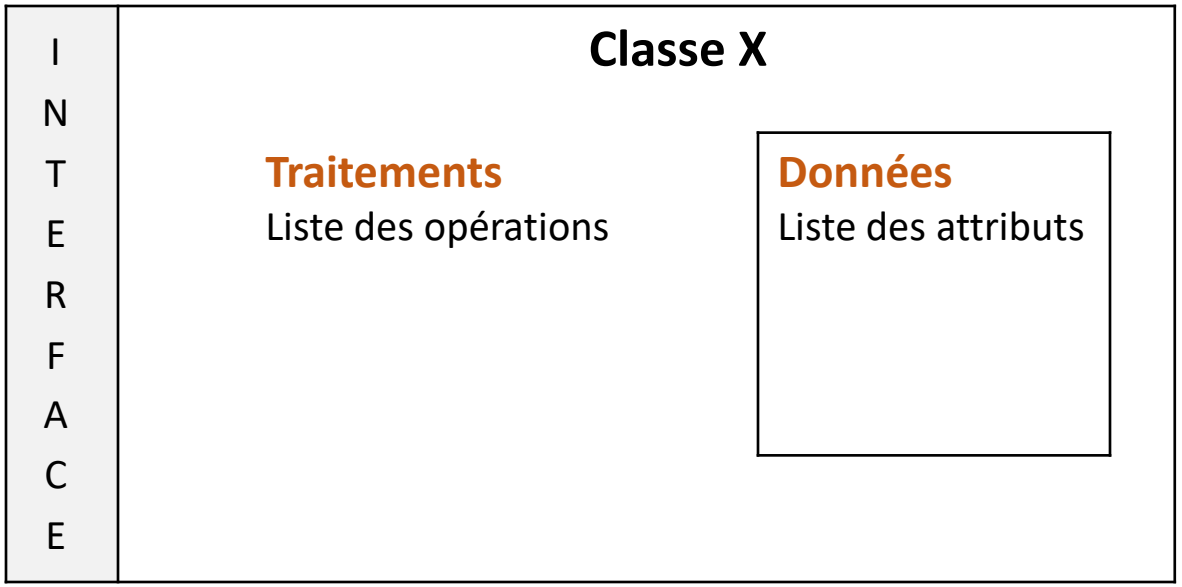


Figure - Schéma du principe de l'encapsulation

Les piliers de la POO

- L'héritage
  - **L'héritage** permet la création d'une nouvelle classe à partir d'une classe existante définissant ainsi une hiérarchie de classes.
  - La classe servant de modèle est appelée **classe de base** ou classe « **mère** ».
  - La classe ainsi créée hérite des caractéristiques de sa classe de base. Elle peut aussi être personnalisée en y ajoutant des caractéristiques supplémentaires.

Les piliers de la POO

- L'héritage
  - Les classes créées à partir d'une classe de base sont appelées **classes dérivées** ou des classes « **filles** ».
  - Exemple :
    - La boîte de vitesse d'une voiture comporte certainement cinq rapports.
    - Les ingénieurs qui ont conçu cette pièce ne sont certainement pas repartis de zéro.
    - Ils ont repris le plan de la génération précédente (quatre rapports) et y ont ajouté des éléments.

Les piliers de la POO

- L'héritage
  - En pratique, la classe de base (**superclasse**) est une classe générique.
  - La spécialisation consiste à créer à partir d'une classe de base, une nouvelle classe également appelée **sous-classe, classe flle ou classe dérivée**.
  - Chaque nouvelle classe ainsi créée est dite spécialisée puisqu'elle comporte en plus des attributs ou opérations (disponibles par héritage), des attributions ou opérations qui lui sont propres.

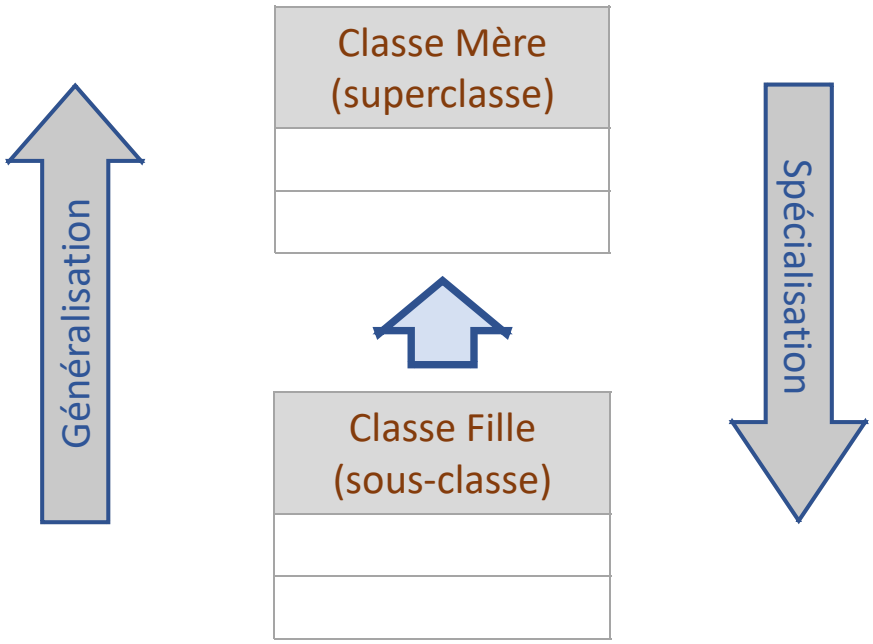
Les piliers de la POO

- L'héritage – Généralisation / Spécialisation
  - La spécialisation de classe se construit en deux temps :
    - d'abord **par héritage** des opérations et attribut d'une superclasse
    - et ensuite **par ajout** d'opérations et/ou attributs spécifiques à la sous-classe.
  - La généralisation représente la démarche inverse puisqu'elle permet de factoriser dans une **superclasse** les opérations et attributs des classes considérées.

Les piliers de la POO

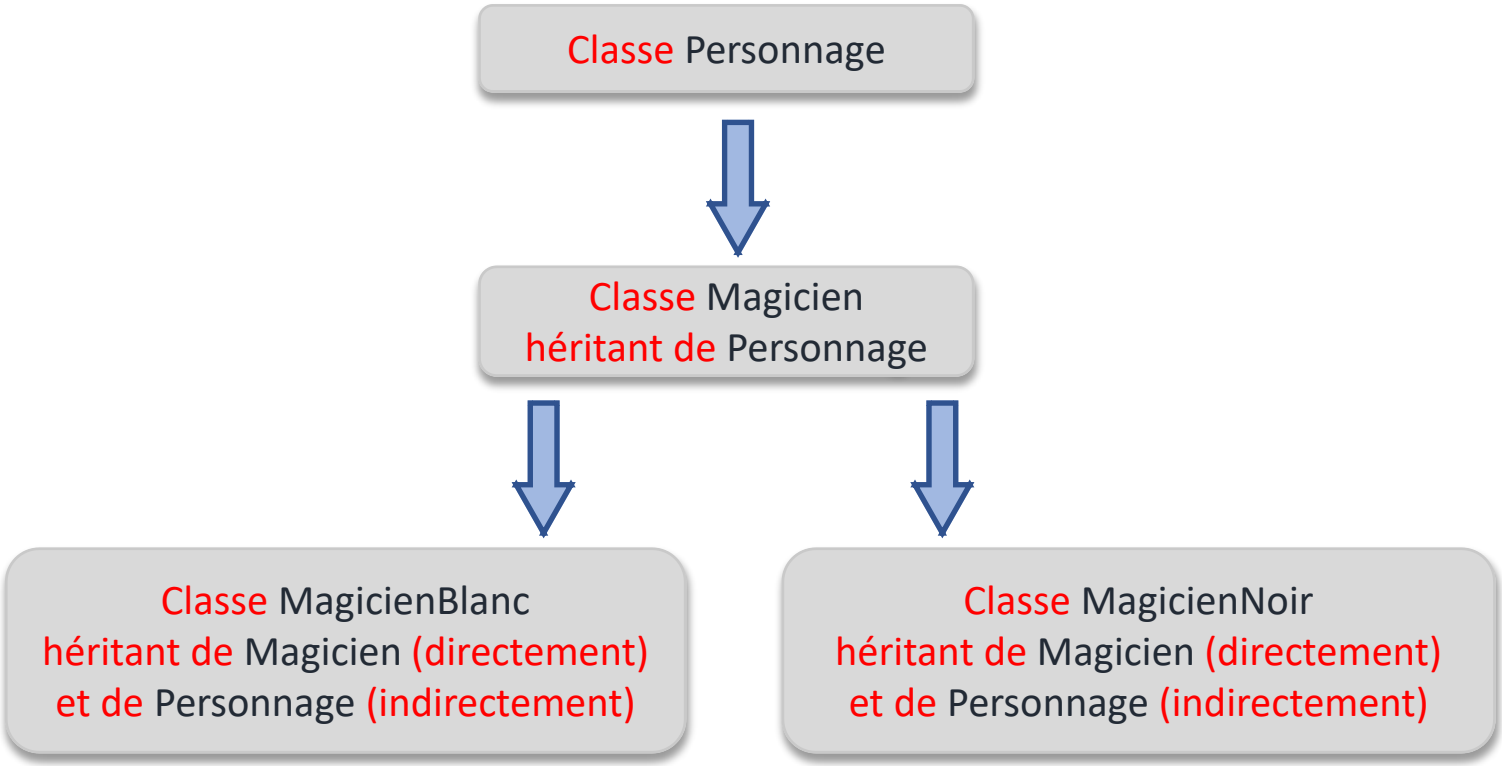
- L'héritage – Généralisation / Spécialisation

En pratique, la classe de base (**superclasse**) est une classe générique.



Les piliers de la POO

- L'héritage – Exemple



Les piliers de la POO

- Le polymorphisme
  - issu du grec
  - signifiant « *qui peut prendre plusieurs formes* »
  - Capacité donnée à une même opération de s'exécuter différemment suivant le contexte de là où elle se trouve.
  - Une opération définie dans une superclasse peut s'exécuter différemment selon la sous-classe où elle est héritée.

Les piliers de la POO

- Le polymorphisme
  - Exemple :
    - Si vous savez changer de vitesse sur une voiture Peugeot, vous savez également comment le faire sur une voiture Renault et pourtant les deux types de boîte de vitesse ne sont pas conçus de la même façon.

Les piliers de la POO

- Le polymorphisme
  - Deux autres concepts sont également associés au polymorphisme : la surcharge et le masquage de méthodes.
    - La surcharge est utilisée pour concevoir dans une classe des méthodes ayant le même nom mais ayant un nombre de paramètres différent ou des types de paramètres différents.
    - Le masquage est utilisé lorsque dans une classe dérivée, on souhaite modifier le fonctionnement d'une méthode dont on a hérité. Le nombre et le type des paramètres restant identiques à ceux définis dans la classe de base.

CONCLUSION

Nous avons mis en évidence les quelques principes de base de la programmation objet. Nous allons voir dans la suite du cours la mise en œuvre de ces principes avec le langage Java.