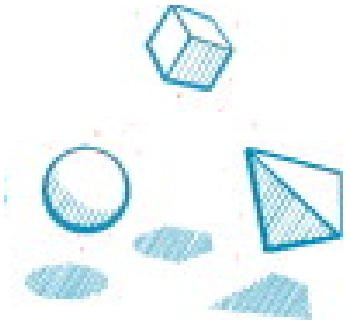
Programmation Orientée Objet - Intro



technocité
creative industries knowledge center

WAVENET
Intelligent IT Solutions

Qu’est ce que la Programmation Orientée Objet?

- La programmation orientée objet ou programmation par objet, est un paradigme* de programmation informatique qui consiste en la définition et l'assemblage de briques logicielles appelées objets.

- Un objet représente un concept, une idée ou toute entité du monde physique, comme une voiture, une personne ou encore une page d'un livre.

- Un programme qui utilise l’approche objet s’appuie sur trois concepts fondamentaux qui seront présentés plus tard en détails:

  - l’encapsulation
  - l’héritage
  - le polymorphisme

*Un **paradigme** est une représentation du monde, une manière de voir les choses.*

La notion d'objet et de classe

- Un objet doit être vu comme une entité cohérente constituée:
  - de **données** qui représentent son **état**
  - de **méthodes** (du code) travaillant sur ses données qui représentent son **comportement**

- Un objet ne peut s'obtenir qu'en **instanciant** celui-ci grâce à une **classe**

- Une classe décrit la structure interne d'un objet à savoir, les données qu'il comprend et les actions qu'il est capable d'assurer sur ses données.

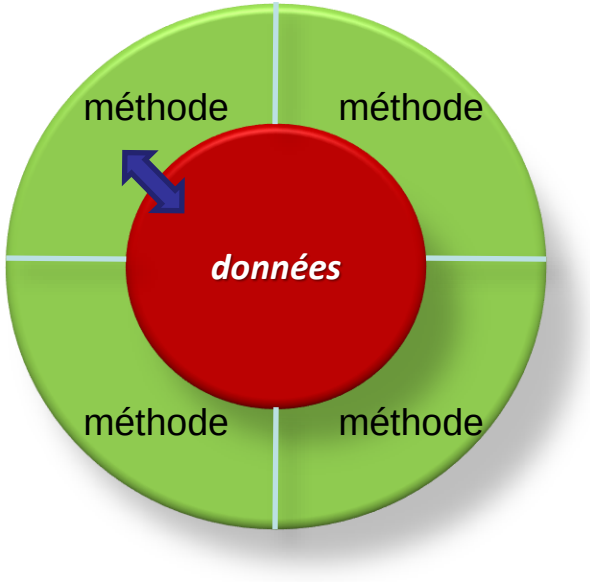
- Une classe peut-être considérée comme un moule, un patron, une usine à partir duquel on fabrique des objets

La notion d'objet et de classe

Classe

La classe joue
le rôle d'une *usine*.
Chaque objet créé
avec cette *usine*
est une instance
de celle-ci

Objet

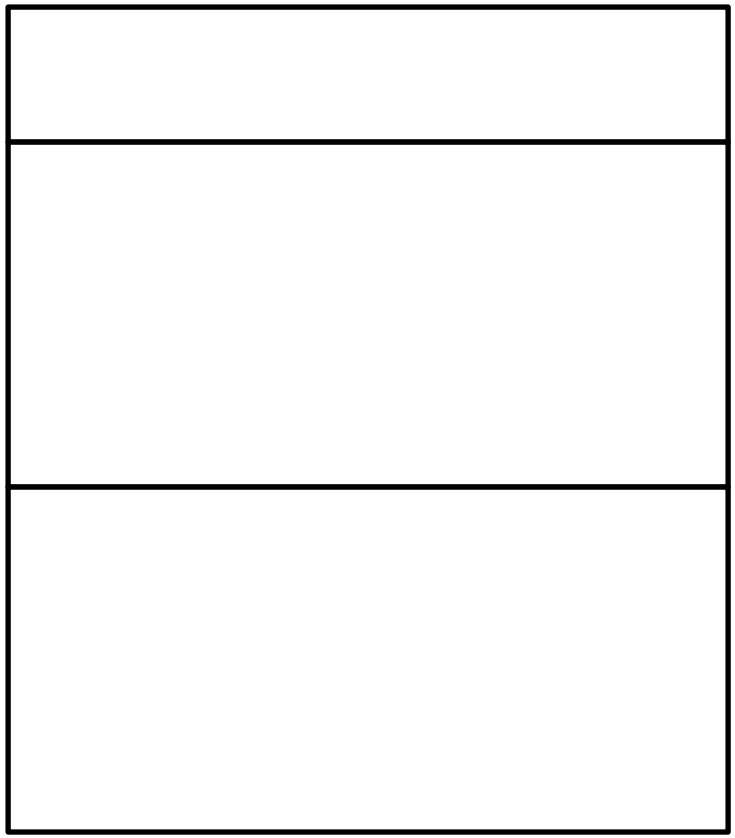


Pensons notre premier objet

- Choisir un objet de la vie de tous les jours
- Définir l'état que peut avoir cet objet (ses données)
- Définir le comportement que peut avoir cet objet (ses méthodes)

(Servez-vous du patron mis à votre disposition à la page suivante)

Pensons notre premier objet



Nom de la classe

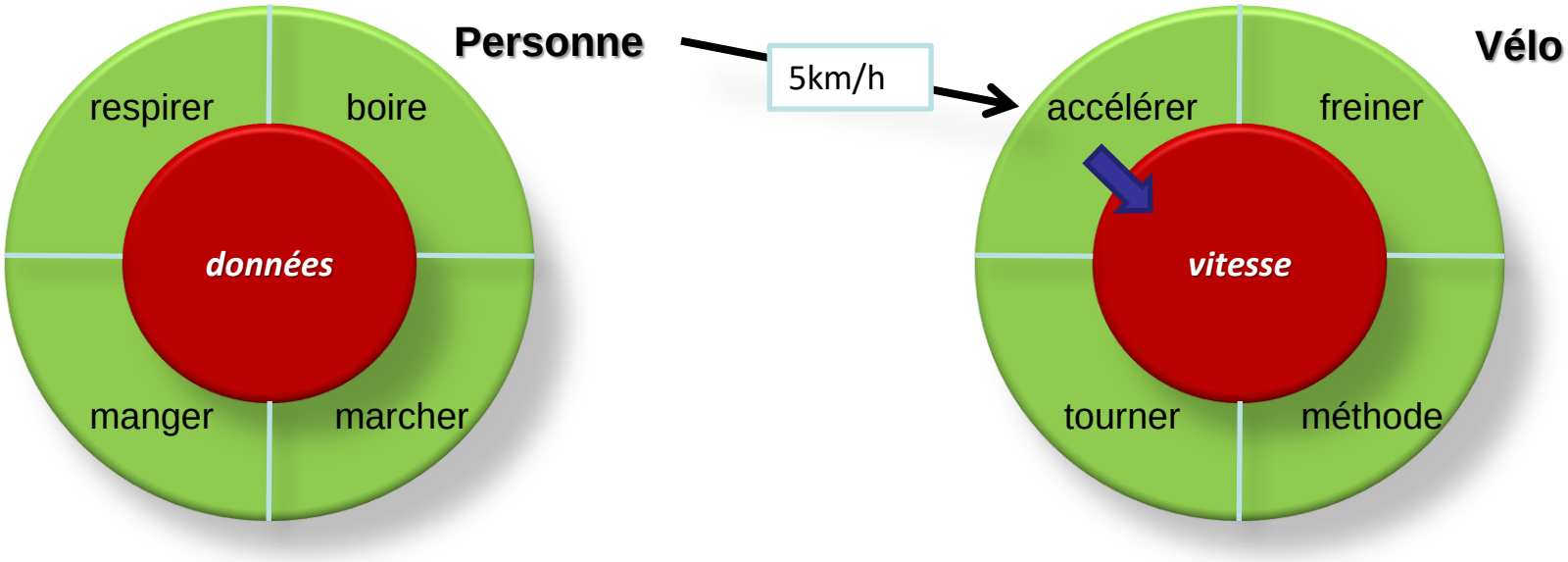
Etat

Comportement

Les objets peuvent communiquer entre eux
La notion de message

- Une application comporte le plus souvent de 1 à n objets
- Les objets interagissent entre eux par le biais de **messages** qu'ils s'envoient.

Exemple : l'objet ***personne*** doit changer la vitesse de l'objet ***vélo***.
Il fait donc appel à la méthode ***freiner*** qui va modifier la vitesse du vélo.
L'objet ***personne*** aura dû spécifier l'objet auquel envoyer le message, le nom de la méthode de cet objet et si nécessaire les paramètres à passer à celle-ci.



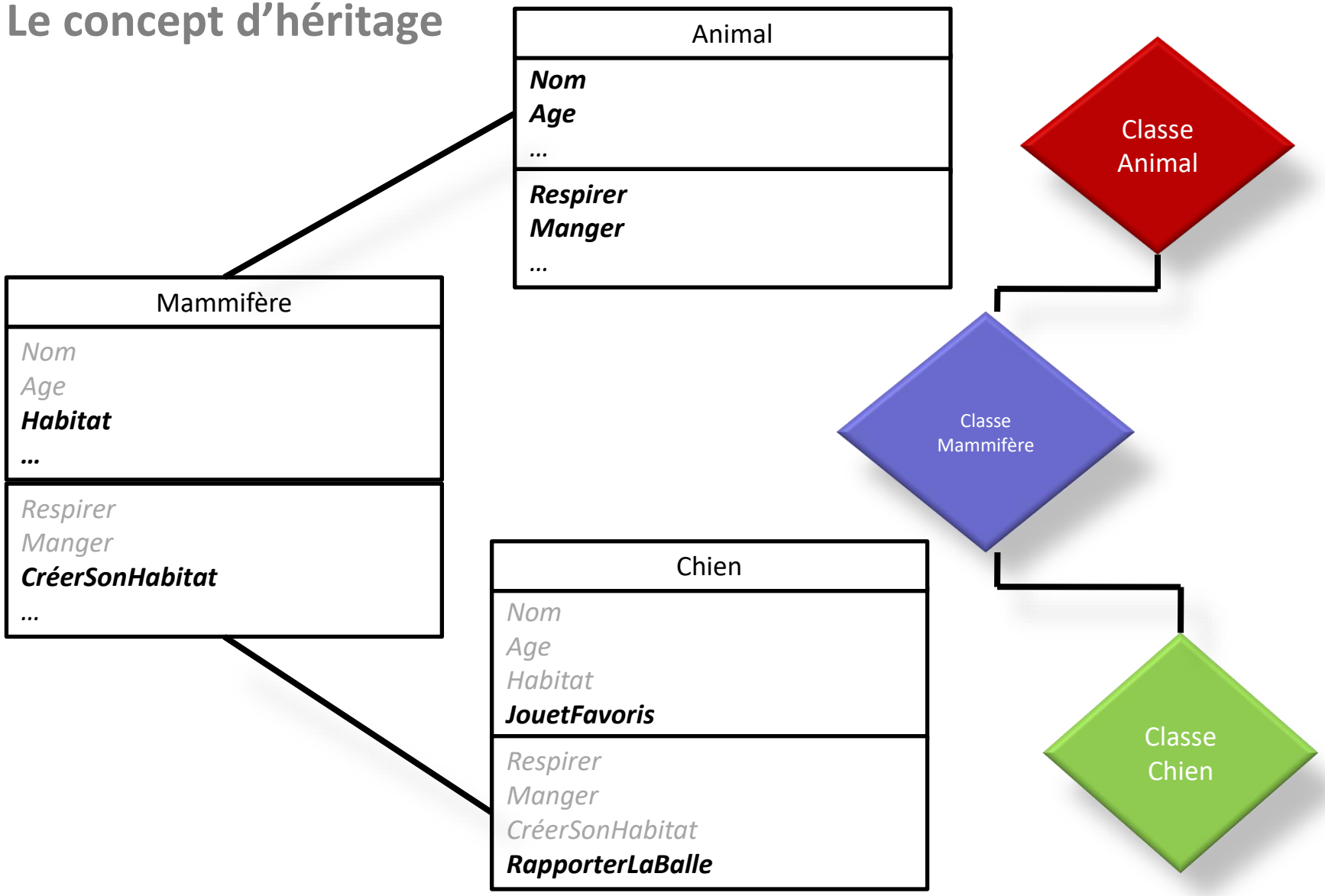
technocité
creative industries knowledge center

WAVENET
Intelligent IT Solutions

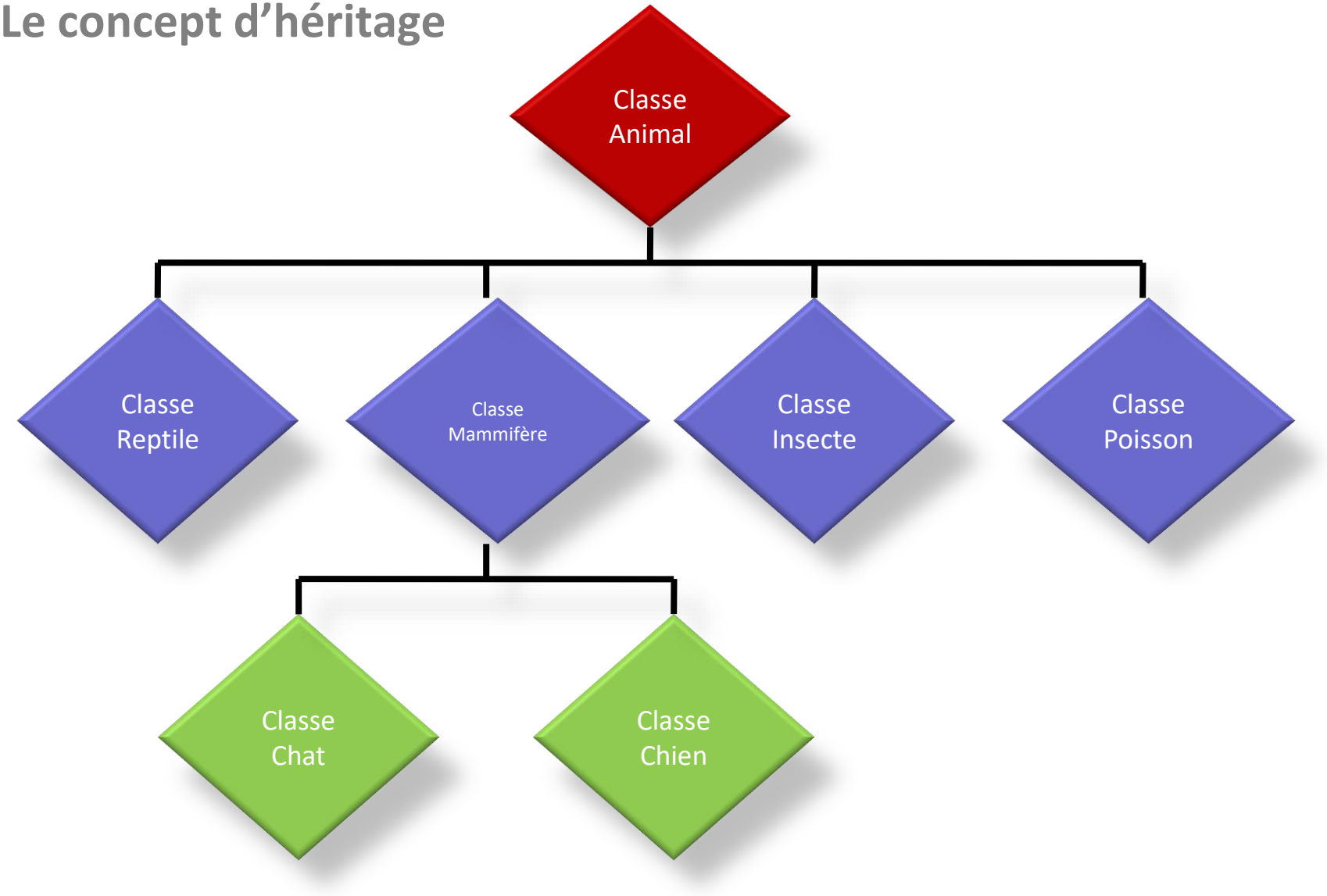
Le concept d'héritage

- L' **héritage** permet de bénéficier du patrimoine de classes existantes, c'est-à dire de pouvoir **étendre** des classes = code réutilisable.
- En orienté objet, il est possible de définir une nouvelle classe à partir d'une autre, ce qui signifie que la **classe descendante** pourra bénéficier des **données** et des **méthodes** de la **classe de base** dont elle hérite.
- Les classes descendantes peuvent contenir de nouvelles données et méthodes permettant ainsi de **spécialiser** la classe de base.
- Les classes descendantes peuvent **compléter** ou **redéfinir** certaines méthodes de la classe de base.
- Grâce à cette technique, il est donc possible de définir des arborescences de classes qui regroupent des classes de plus en plus spécialisées.

Le concept d'héritage



Le concept d'héritage



Le concept de polymorphisme

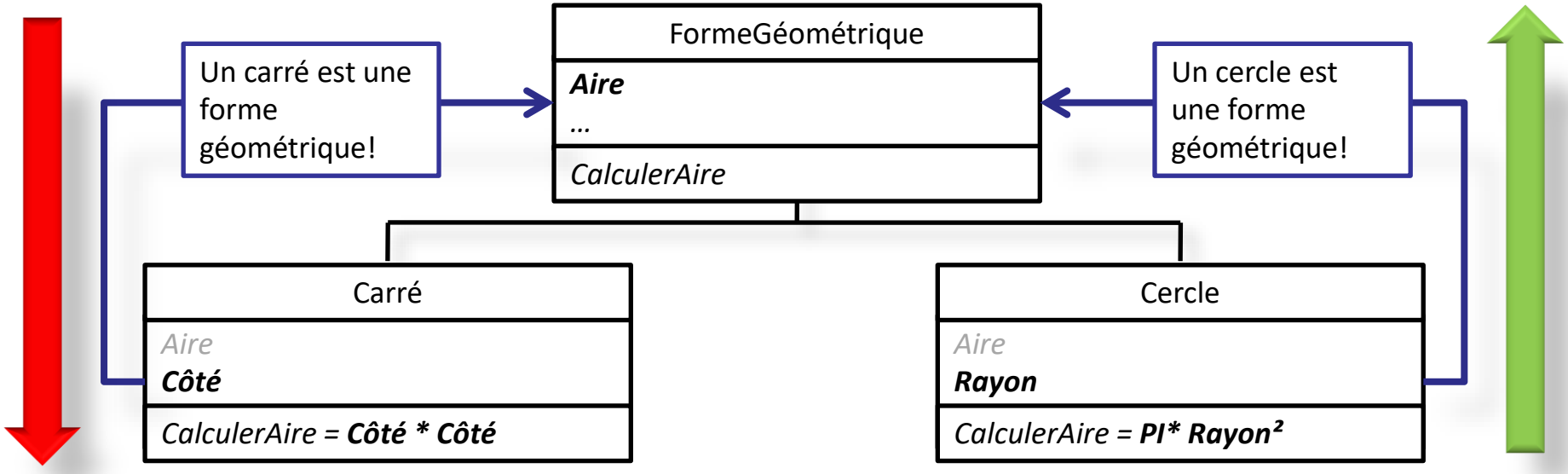
- En informatique, le **polymorphisme** est l'idée d'autoriser le même code à être utilisé avec différents types, ce qui permet des implémentations plus abstraites et générales.

- Cette technique s'utilise conjointement avec l'héritage et représente sans conteste le plus grand apport des langages orientés objet.

- Une forme faible de polymorphisme est la redéfinition d'une méthode dans une classe héritant d'une classe de base. Au même nom de méthode, on va associer des comportements différents selon la classe dans laquelle on se trouve.

- On peut faire de l'héritage et se passer du polymorphisme mais pas l'inverse.

Le concept de polymorphisme



En partant de ce schéma, on peut dire que dans un programme objet, là ou l'on attend un objet de type *FormeGéométrique,* on pourra utiliser un objet de type *Carré* ou *Cercle*…

En résumé

- Tout objet est une instance d'une classe, qui est le moule générique des objets de ce type.
- Les classes définissent les comportements possibles de leur objet.
- L'exécution d'un programme est réalisé par l'échange de messages entre objets.
- Un message est une demande d'action caractérisée par les paramètres nécessaires à la réalisation de cette action.
- Les classes sont organisées en une structure arborescente à racine unique : la hiérarchie d'héritage.