

TP - Réutilisation et héritage en Java

Emmanuel BRUNO

Université de Toulon, LIS UMR CNRS 7020
emmanuel.bruno@univ-tln.fr

2026-10-14

Table des matières

1 Objectif	1
2 Modélisation par héritage et Instanciation	1
3 Polymorphisme simple	1
4 Classes abstraites	2
5 Administration de classe et polymorphisme	2
6 Héritage multiple et Interface	3

1 Objectif

L'objectif ce TP est de présenter l'héritage, le polymorphisme et l'utilisation des interfaces en Java. Vous devez écrire la javadoc au fur et à mesure de l'avancement et il est fortement conseillé d'avoir la documentation de java sous les yeux : <https://docs.oracle.com/en/java/javase/17/docs/api/index.html>

Nous allons modéliser des plantes, des animaux, des mammifères et des oiseaux mais aussi des chiens, des aigles, des lapins et des hommes.

2 Modélisation par héritage et Instanciation

A Faire

1. Modélisez cette hiérarchie (directement en Java) avec des classes. Ajoutez à chaque classe fille au moins une variable et une méthode nouvelle selon votre imagination (âge, nom, sexe, ...).
2. Ajoutez une classe exécutable Test et créez des instances de chacune des classes. Affichez avec la méthode `System.out.println()` chacun des objets.
3. Redéfinissez la méthode `toString()` (héritée de `Object`, voir dans la documentation) de la super classe pour qu'elle affiche «Je suis un animal» (ou «Je suis une plante»). Exécutez la classe Test. Expliquer le résultat en étudiant la classe `java.lang.System`.
4. Modifiez la méthode précédente pour qu'elle affiche «Je suis un animal et mon identifiant est X».

3 Polymorphisme simple

Avec quels types de références pouvez-vous manipuler les classes précédentes ?

A Faire

Dans la méthode `main` la classe `Test`, créez des tableaux qui permettent de voir certains des animaux créés dans la section précédente comme des collections (de taille fixe) d'animaux ou de mammifères.

Ajouter à chaque classe une méthode `String getInfo()` (qui appellera celle de la superclasse) pour que l'affichage du tableau suivant :

```
{new Animal(12), new Animal(), new
Chien(5, "Medor"), new Homme(),
new Homme(25, "Robert")}
```

soit de la forme :

```
Je suis un animal âgé de 12 an(s).
Je suis un animal.
Je suis un animal âgé de 5 an(s).
Je suis un mammifère. Je suis un
chien de nom Médor.
Je suis un animal. Je suis un
mammifère. Je suis un homme.
Je suis un animal âgé de 25 an(s).
Je suis un mammifère. Je suis un
homme de nom Robert.
```

4 Classes abstraites

A Faire

1. Modifiez les classes pour s'assurer que tous les Animaux possèdent la méthode `String moyenExpression()` qui retourne une phrase du type « Je parle », « J'aboie », ou « Je fais des bulles » (l'aigle trompette, glapit ou glatit ; le lapin clapit).
2. Est-ce que le moyen d'expression peut être défini pour toutes les classes ? Peut-on alors créer une instance de la classe `Animal` ? Quelle est votre conclusion pour s'assurer de cela ? Faites de même pour toutes les classes pour les lesquelles cela vous semble nécessaire.
3. Écrivez une méthode `void afficherCrisAnimaux(Animal[] animaux)` dans la classe `Animal` qui prend en paramètre un tableau d'animaux et qui affiche leur cri. Testez la sur le tableau de la question précédente.

5 Administration de classe et polymorphisme

A Faire

1. Modifiez vos classes pour que l'on garde automatiquement des références (dans un tableau) vers tous les animaux créés (Vous vérifierez aussi que l'on ne crée pas plus de `NB_MAX_ANIMAUX`).
2. Créez une méthode (de classe ou d'instance ?) qui affiche la description complète de tous les animaux créés.

6 Héritage multiple et Interface

On souhaite maintenant considérer que certains animaux (pas forcément tous) sont des carnivores (ils comportent la méthode `void manger` avec comme paramètre un autre animal) ou des herbivores (ils comportent la méthode `void manger` avec comme paramètre une plante) ou les deux. Créez deux interfaces pour décrire ces comportements.

A Faire

1. Modifiez vos classes pour qu'elles implantent ces interfaces :
 1. Les carnivores : quand ils mangent, les chiens affichent « Je mord X », et les aigles affichent « Je déchire X ».
 2. Les herbivores : les vaches affichent « Je broute X », et les lapins « Je grignotte X ».
2. Les hommes sont à la fois carnivores et herbivores, proposez deux solutions pour le représenter :
 1. Dans un premier temps, sans créer de nouvelle interface.
 2. Dans un second temps en créant une nouvelle interface `IOmnivore`.
3. Le polymorphisme est aussi utilisable avec les interfaces : Appliquez la méthode `manger(unePlante)` sur la Vache que vous avez déjà créée en la voyant à travers une référence de type `IHerbivore`. Peut-on appliquer la méthode `moyenExpression()` sur cette référence ?
4. Créez deux tableaux `lesCarnivores` et `lesHerbivores` pour manipuler les animaux existants. Créez un lapin, et faites le manger par tous les carnivores, créez une plante et faites la manger par les herbivores.