# TP - Bases de Java

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

2026-10-31

## Table des matières

1 Création et instanciation d'une classe ..... 1
2 Les méthodes de base ..... 2
2.1 toString() ..... 2
2.2 equals() ..... 2
3 Définition de classes locales ..... 2
4 Pour finir... ..... 3

## 1 Création et instanciation d'une classe

1. **Créer une classe Java exécutable nommée** `Test` qui appartient au paquetage `tp1`. Ajouter un prefixe de packages unique. La classe `Test` devra afficher « Hello ».
2. **Créer une classe** `Personne` sans constructeur explicite qui décrit une personne ayant un nom, un prénom, un âge et un salaire. Créer les accesseurs et modificateurs correspondants à ces attributs (l'IDE peut le faire automatiquement). Vous vérifierez qu'un salaire ne peut pas être négatif. Instancier une personne `p1` dans la méthode `main` de la classe `Test`, puis mettre à jour ses informations et les afficher (Pierre Truc est âgé de 30 ans et gagne 2000€).
3. **Ajouter des constructeurs dans la classe** `Personne` qui initialisent le nom et le prénom, l'âge et la profession. Les deux derniers sont optionnels. Modifier cette classe pour que le nom et le prénom ne puissent plus être modifiés après l'instanciation.
4. **Ajouter un attribut qui indique l'année de naissance** et les accesseurs associés. Modifier la méthode qui retourne l'âge. L'année courante s'obtient avec `java.util.Calendar.getInstance().get(Calendar.YEAR)`.
5. **Ajouter une méthode** `comparerSalaire()` qui prend en paramètre une personne et retourne −1, 0 ou 1 selon que celle-ci a un salaire supérieur, égal ou inférieur à l'instance courante. Créer une deuxième personne `p2` née la même année que

`p1` mais qui gagne plus. Afficher le résultat de la comparaison des salaires de `p1` et `p2`.

6. **Ajouter une méthode `comparerSalaire()`** qui prend en paramètre deux personnes et qui retourne un nombre négatif, nul ou positif selon que le salaire de la première est inférieur, égal ou supérieur à celui de la seconde.
7. **Attention**, `comparerSalaire(Personne)` et `comparerSalaire(Personne, Personne)` sont-elles des méthodes de classe ou d'instance ?
8. **Ajouter un attribut de classe `totalDesSalaires`** qui indique le total des salaires des personnes et un accesseur. Modifier les méthodes nécessaires pour que le salaire total soit maintenu à jour automatiquement. Est-ce risqué d'écrire simplement une telle méthode ?
9. **Afficher le salaire total** après la création de chaque personne.

# 2 Les méthodes de base

## 2.1 toString()

- Afficher directement l'objet `p1` dans la classe `Test`.
- Ajouter une méthode `String toString()` à la classe `Personne` qui retourne une chaîne de caractères qui décrit la personne.
- Exécuter `Test`.

## 2.2 equals()

- On souhaite considérer que deux personnes sont « égales » dans notre application, si elles sont nées la même année.
- Pour définir une relation d'égalité d'objets en Java, on redéfinit la méthode `boolean equals(Object)`.
- Vérifier que `p1` et `p2` sont égales.

# 3 Définition de classes locales

- Pour représenter le fait qu'une `Personne` a un cerveau, ajouter une définition de la classe `Cerveau` à l'intérieur de la définition de la classe `Personne`.
- Ajouter un attribut `cerveau` à `Personne` et instancier un cerveau à chaque personne lors de sa création.
- Compiler et exécuter `Test`.
- Regarder dans le répertoire build les classes qui ont été compilées.
- Essayer de créer une instance de `Cerveau` directement dans la classe `Test`.

# 4 Pour finir...

Vous allez maintenant créer une classe `Entreprise` qui permet de représenter des entreprises qui comportent au maximum `MAX_EMPLOYES` personnes.

1. **Ajouter les méthodes** pour créer une entreprise, ajouter des employés et afficher le nombre d'employés, un employé précis et tous les employés (en utilisant le `foreach`).
2. **Catégoriser les entreprises** : Les entreprises doivent appartenir à l'une des trois catégories suivantes : `{PUBLIQUE, PME, GRAND GROUPE}`. Modifier la classe `Entreprise` pour représenter cela en utilisant un type `enum` et ajouter trois méthodes `isPUBLIQUE()`, `isPME()`, `isGRAND_GROUPE()` qui retournent un booléen.
3. **Ajouter une méthode** `miseEnForme()` qui passe le nom de toutes les personnes d'une entreprise en majuscule.
4. **Modifier votre classe** `Test`, pour qu'on puisse lui passer en paramètre le nom et le type d'une entreprise, puis la liste des personnes qui la composent.