

# Java : classes interne et locale

**Achref El Mouelhi**

Docteur de l'université d'Aix-Marseille  
Chercheur en programmation par contrainte (IA)  
Ingénieur en génie logiciel

[elmouelhi.achref@gmail.com](mailto:elmouelhi.achref@gmail.com)



- 1 Classe interne
- 2 Classe locale

# Java

## Classe interne ?

- Une classe définie à l'intérieur d'une autre
- Pouvant accéder aux attributs et méthodes de la classe englobante

# Java

Considérons la classe Personne suivante

```
package org.eclipse.model;

public class Personne {

    private String nom;
    private String prenom;
    private int age;

    public Personne(String nom, String prenom, int age) {
        this.nom = nom;
        this.prenom = prenom;
        this.age = age;
    }

    // + getters et setters
}
```

# Java

## Hypothèse

Si on décide d'ajouter les deux propriétés suivantes dans la classe Personne

- `majoriteSexuelle` : acceptant les valeurs `majeur` et `mineur`
- `periode` : acceptant les valeurs `bébé`, `enfant`, `adolescent`, et `adulte`



# Java

## Hypothèse

Si on décide d'ajouter les deux propriétés suivantes dans la classe Personne

- majoriteSexuelle : acceptant les valeurs majeur et mineur
- periode : acceptant les valeurs bébé, enfant, adolescent, et adulte

## Remarques

- Les deux propriétés dépendent de l'attribut age
- On peut donc les regrouper ensemble dans une nouvelle classe
- Cette nouvelle classe doit être placée dans Personne pour qu'elle ait accès à l'attribut age

## Exemple de code pour la classe Categorie qu'on va définir dans Personne

```
public class Categorie {  
    private String majoriteSexuelle;  
    private String periode;  
    public Categorie() {  
        if (age >= 18) {  
            majoriteSexuelle = "majeur";  
            periode = "adulte";  
        } else {  
            majoriteSexuelle = "mineur";  
            if (age < 2)  
                periode = "bébé";  
            else if (age < 12)  
                periode = "enfant";  
            else  
                periode = "adolescent";  
        }  
    }  
    @Override  
    public String toString() {  
        return "Categorie [majoriteSexuelle = " + majoriteSexuelle + ",  
               periode = " + periode + "]";  
    }  
}
```

## Modifions la classe Personne pour intégrer la classe Categorie

```
public class Personne {  
  
    private String nom;  
    private String prenom;  
    private int age;  
    private Categorie categorie;  
  
    public Personne(String nom, String prenom, int age) {  
        this.nom = nom;  
        this.prenom = prenom;  
        this.age = age;  
        categorie = new Categorie();  
    }  
  
    // + getters et setters  
  
    @Override  
    public String toString() {  
        return "Personne [nom = " + nom + ", prenom = " + prenom + ", age =  
            " + age + ", categorie = " + categorie + "]";  
    }  
  
    // + code de la classe Categorie  
}
```

# Java

## Pour tester

```
package org.eclipse.test;

public class Main {
    public static void main(String [] args) {
        Personne personne = new Personne("el mouelhi", "achref
            ", 34);
        System.out.println(personne);
    }
}
```



# Java

## Pour tester

```
package org.eclipse.test;

public class Main {
    public static void main(String [] args) {
        Personne personne = new Personne("el mouelhi", "achref
            ", 34);
        System.out.println(personne);
    }
}
```

## Le résultat sera

```
Personne [nom = el mouelhi, prenom = achref, age=34,
categorie = Categorie [majoriteSexuelle = majeur,
periode = adulte]]
```

# Java

## Remarques

À la compilation deux fichiers, relatifs à la classe Personne, ont été générés (d'extension .class)

- Le premier : Personne
- Le deuxième : Personne\$Categorie



# Java

## Remarques

À la compilation deux fichiers, relatifs à la classe Personne, ont été générés (d'extension .class)

- Le premier : Personne
- Le deuxième : Personne\$Categorie

## Attention

En changeant l'age, les valeurs de periode et majoriteSexuelle ne seront pas mises à jour.

# Java

## Remarques

À la compilation deux fichiers, relatifs à la classe Personne, ont été générés (d'extension .class)

- Le premier : Personne
- Le deuxième : Personne\$Categorie

## Attention

En changeant l'age, les valeurs de periode et majoriteSexuelle ne seront pas mises à jour.

## Solution

Utiliser les classes locales.

## Classe locale ?

- Une classe définie dans une méthode d'une autre
- Pouvant accéder aux attributs et méthodes de la classe englobante
- Pouvant aussi accéder aux paramètres de la méthode où elle est définie

## Java

Considérez la classe Personne suivante

```
package org.eclipse.model;

public class Personne {

    private String nom;
    private String prenom;
    private int age;

    public Personne(String nom, String prenom, int age) {
        this.nom = nom;
        this.prenom = prenom;
        this.age = age;
    }

    // + getters et setters
}
```

# Java

## Hypothèse

Nous allons garder les attributs de la classe `Categorie`

- `majoriteSexuelle` : acceptant les mêmes valeurs `majeur` et `mineur`
- `periode` : acceptant aussi les mêmes valeurs `bébé`, `enfant`, `adolescent`, et `adulte`



# Java

## Hypothèse

Nous allons garder les attributs de la classe `Categorie`

- `majoriteSexuelle` : acceptant les mêmes valeurs `majeur` et `mineur`
- `periode` : acceptant aussi les mêmes valeurs `bébé`, `enfant`, `adolescent`, et `adulte`

## Remarques

- La classe `Categorie` sera plutôt définie dans une méthode (`afficherDetails()`) de la classe `Personne`
- Elle sera aussi instanciée dans cette même méthode

# Java

La méthode `afficherDetails()` qu'on va ajouter dans `Personne`

```
public void afficherDetails() {
    class Categorie {
        private String majoriteSexuelle;
        private String periode;

        public Categorie() {
            if (age >= 18) {
                majoriteSexuelle = "majeur";
                periode = "adulte";
            } else {
                majoriteSexuelle = "mineur";
                if (age < 2)
                    periode = "bébé";
                else if (age < 12)
                    periode = "enfant";
                else
                    periode = "adolescent";
            }
        }

        @Override
        public String toString() {
            return "majoriteSexuelle=" + majoriteSexuelle + ", periode=" + periode;
        }
    }

    Categorie categorie = new Categorie();
    System.out.println("nom= " + nom + ", prenom = " + prenom + ", " + categorie);
}
```

# Java

## Pour tester

```
package org.eclipse.test;

public class Main {
    public static void main(String [] args) {
        Personne personne = new Personne("el mouelhi","achref",34);
        personne.afficherDetails();
        personne.setAge(10);
        personne.afficherDetails();
    }
}
```



# Java

## Pour tester

```
package org.eclipse.test;

public class Main {
    public static void main(String [] args) {
        Personne personne = new Personne("el mouelhi", "achref", 34);
        personne.afficherDetails();
        personne.setAge(10);
        personne.afficherDetails();
    }
}
```

Le résultat sera

```
nom = el mouelhi, prenom = achref, majoriteSexuelle = majeur,
periode = adulte
nom = el mouelhi, prenom = achref, majoriteSexuelle = mineur,
periode = enfant
```

## Remarques

À la compilation deux fichiers, relatifs à la classe Personne, ont été générés (d'extension .class)

- Le premier : Personne
- Le deuxième : Personne\$1Categorie ⇒ 1 étant l'indice de la classe locale (les classes locales seront numérotées dans l'ordre et la première aura l'indice 1)