

Programmation Orientée Objet avec Java

TP N°4

Héritage et polymorphisme

Objectif du TP :

- **Utiliser l'héritage**
- **Utiliser le polymorphisme**
- **Développer une première application pour la gestion des comptes bancaires et une deuxième pour le calcul des salaires des employés d'une entreprise.**

Exercice1 : Comptes Bancaires

- Un compte bancaire possède à tout moment une donnée : son solde. Ce solde peut être positif (compte créditeur) ou négatif (compte débiteur).
- Chaque compte est caractérisé par un code incrémenté automatiquement.
- A sa création, un compte bancaire a un solde nul et un code incrémenté.
- Il est aussi possible de créer un compte en précisant son solde initial.
- Utiliser son compte consiste à pouvoir y faire des dépôts et des retraits. Pour ces deux opérations, il faut connaître le montant de l'opération.
- L'utilisateur peut aussi consulter le solde de son compte par la méthode toString().
- Un compte Epargne est un compte bancaire qui possède en plus un champ « TauxInterêt =6» et une méthode calculInterêt() qui permet de mettre à jour le solde en tenant compte des intérêts.
- Un ComptePayant est un compte bancaire pour lequel chaque opération de retrait et de virement est payante et vaut 5 dh

Questions :

- Faire le diagramme des classes
- Définir la classe CompteBancaire .
- Définir la classe CompteEpargne.
- Définir la classe ComptePayant.
- Définir une classe contenant la fonction main() permettant de tester les classes CompteBancaire et CompteEpargne avec les actions suivantes:
 - Créer une instance de la classe CompteBancaire , une autre de la classe CompteEpargne et une instance de la classe ComptePayant
 - Faire appel à la méthode deposer() de chaque instance pour déposer une somme quelconque dans ces comptes.
 - Faire appel à la méthode retirer() de chaque instance pour retirer une somme quelconque de ces comptes.
 - Faire appel à la méthode calculInterêt() du compte Epargne.
 - Afficher le solde des 3 comptes.

Exercice 2 : L'héritage appliqué aux employés d'une entreprise, polymorphisme

L'objectif de cet exercice est de programmer le calcul des salaires hebdomadaires des employés d'une entreprise.

Cette entreprise comporte plusieurs types d'employés :

- Des employés qui sont payés suivant le nombre d'heures qu'ils ont travaillé dans la semaine. Ils sont payés à un certain tarif horaire et leurs heures supplémentaires (au-delà de la charge horaire due (=40)) sont payées 30 % de plus que les heures normales.
- D'autres employés, payés de la même façon, mais leurs heures supplémentaires sont payées 50 % de plus que les heures normales.
- Les commerciaux sont payés avec une somme fixe à laquelle on ajoute 1 % du chiffre d'affaires qu'ils ont fait dans la semaine.

Modélisez cette situation à l'aide de classes qu'il faut placer dans un package entreprise.

- Chaque employé doit avoir un nom. Ce nom, une fois donné ne peut pas être modifié.
- Commencez par écrire une classe `Employe` dont hériteront les autres classes.
- Pour ne pas avoir trop de modificateurs, créez un seul modificateur `setInfosSalaire` pour entrer ou modifier les informations brutes nécessaires au calcul des salaires (nombre d'heures de travail, chiffre d'affaire,...). Ne faites pas du polymorphisme avec cette méthode.
- Les classes doivent comporter au moins 2 constructeurs : un qui ne prend en paramètre que le nom de l'employé et l'autre qui prend en paramètres le nom et toutes les informations pour le calcul du salaire.
- Créez une méthode `getSalaire()` qui sera utilisée pour calculer le salaire de n'importe quel employé en utilisant le polymorphisme.

Créez une classe `Paie` qui comporte une méthode `main()` qui a pour tâches :

Entrer les informations sur des employés des différents types (au moins 3 commerciaux). Utilisez un tableau pour enregistrer Les employés.

Au moins un des employés sera créé avec le constructeur qui n'a que le nom en paramètre, et entrez ensuite les informations pour son salaire avec la méthode `setInfosSalaire`.

Pour au moins un autre employé, utilisez le constructeur pour entrer les informations sur le salaire.

La méthode `main()` affichera le salaire hebdomadaire de chacun des employés dans une boucle "for" qui parcourra le tableau des employés. Utilisez le polymorphisme avec un accesseur pour le salaire. L'affichage doit avoir la forme : "Moha gagne 7800 DHs ".

=====
Correction
=====

=====
Exercice1
=====

```
public class CompteB {
    int code;
    float solde;
    static int nbComptes=0;

    public CompteB(float s){
        solde=s;
        ++nbComptes;
        code=nbComptes;
    }
    public CompteB(){
        this(0);
    }

    public void verser(float mt){
        solde=solde+mt;
    }
    public void retirer(float mt){
        solde-=mt;
    }
    public String toString(){
        return ("code = "+code+ " Solde= "+solde);
    }
}
```

```
public class CompteE extends CompteB {
    float taux=6;
    public CompteE(float s){
        super(s);
    }
    public CompteE(){
        super(0);
    }
    public void calculInterets(){
        solde=solde*(1+taux/100);
    }
    public String toString(){
        return "Compte Epargne "+super.toString()+ " taux=
"+taux;
    }
}
```

```
public class CompteP extends CompteB {
    public CompteP(float s){
        super(s);
    }
}
```

```

public CompteP(){
    super(0);
}
public void verser(float mt){
    super.verser(mt);
    super.retirer(5);
}
public void retirer(float mt){
    super.retirer(mt);
    super.retirer(5);
}
public String toString(){
    return "Compte Payant "+super.toString();
}
}

```

```

public class TestCompte {

    public static void main(String[] args) {
        CompteB c1=new CompteB(5000);
        CompteE c2=new CompteE(55000);
        CompteP c3=new CompteP();

        c1.verser(6000);
        c1.retirer(4000);
        System.out.println(c1.toString());
        c2.verser(3300);
        c2.retirer(2000);
        System.out.println(c2);
        c2.calculInterets();
        System.out.println(c2);
        c3.verser(2000);
        System.out.println(c3.toString());
    }
}

```

```

=====Exercice 2 =====
public class Employe {

    private String nom;

    protected Employe(String nom) {
        this.nom = nom;
    }

    public String getNom() {
        return nom;
    }

    // On pourrait aussi déclarer cette méthode abstraite

```

```

// (et donc la classe Employe abstraite)
public double getSalaire() {
    return -1.0;
}

public String toString() {
    return getClass().getName() + "; nom=" + nom
        + "; salaire=" + getSalaire();
}

// Inutile de déclarer setInfosSalaire ici car on ne
// peut faire de polymorphisme avec cette méthode
}

```

```

public class Commercial extends Employe {
    // chiffre d'affaires du mois
    private double ca = -1.0,
                fixe = -1.0;
    // pourcentage du CA pour le calcul du salaire
    private static final double pourcentageCa = 1.0;

    public Commercial(String nom, double fixe, double ca) {
        super(nom);
        this.fixe = fixe;
        this.ca = ca;
    }

    public Commercial(String nom) {
        super(nom);
    }

    public void setInfosSalaire(double fixe, double ca) {
        this.fixe = fixe;
        this.ca = ca;
    }

    public void setCA(double ca) {
        if (ca >= 0) {
            this.ca = ca;
        }
    }

    public void setFixe(double fixe) {
        if (fixe >= 0) {
            this.fixe = fixe;
        }
    }

    public double getSalaire() {
        if (ca != -1.0 && fixe != -1.0) {

```

```

        return fixe + (pourcentageCa * ca) / 100;
    }
    else {
        return -1.0;
    }
}

public String toString() {
    return super.toString() + ";fixe = " + fixe + " ca = " +
ca;
}
}

```

```

public class EmployeHoraire extends Employe {

    private double nbHeures = -1.0;
    private double tarifHoraire = -1.0;
    private double pourcentageHeuresSup = -1.0;
    private static final double heuresDues = 39.0;

    public EmployeHoraire(String nom) {
        super(nom);
    }

    public EmployeHoraire(String nom, double nbHeures, double
tarifHoraire,
        double pourcentageHeuresSup) {
        super(nom);
        this.nbHeures = nbHeures;
        this.tarifHoraire = tarifHoraire;
        this.pourcentageHeuresSup = pourcentageHeuresSup;
    }

    public void setInfosSalaire(double nbHeures, double
tarifHoraire,
        double pourcentageHeuresSup) {
        this.nbHeures = nbHeures;
        this.tarifHoraire = tarifHoraire;
        this.pourcentageHeuresSup = pourcentageHeuresSup;
    }

    public double getNbHeures() {
        return nbHeures;
    }

    public double getTarifHoraire() {
        return tarifHoraire;
    }

    public double getPourcentageHeuresSup() {

```

```

    return pourcentageHeuresSup;
}

public double getSalaire() {
    if (nbHeures != -1 && tarifHoraire != -1 &&
pourcentageHeuresSup != -1) {
        if (nbHeures > heuresDues) {
            return tarifHoraire * (heuresDues +
                (nbHeures - heuresDues)
                * (1 + pourcentageHeuresSup / 100));
        }
        else {
            return tarifHoraire * nbHeures;
        }
    }
    else {
        return -1;
    }
}

public String toString() {
    return super.toString() + ";nbHeures=" + nbHeures
        + "; tarif horaire=" + tarifHoraire + "; % HS=" +
pourcentageHeuresSup;
}
}

```

```

public class Paie {
    public static void main(String[] args) {
        Employe[] employes = new Employe[7];
        employes[0] = new Commercial("Toto", 6000, 300000);
        employes[1] = new Commercial("Moha", 6500, 235000);
        employes[2] = new Commercial("Manolo", 5000, 250000);
        employes[3] = new EmployeHoraire("Habti", 41, 200, 30);
        employes[4] = new EmployeHoraire("Farti", 45, 150, 50);
        EmployeHoraire e5 = new EmployeHoraire("Rabati");
        e5.setInfosSalaire(40, 100, 50);
        employes[5] = e5;
        employes[6] = new EmployeHoraire("Tanjaoui");
        double salaire;
        for (int i = 0; i < employes.length; i++) {
            System.out.print(employes[i].getNom());
            salaire = employes[i].getSalaire();
            if (salaire == -1)
                System.out.println(" n'a pas son salaire fixé");
            else
                System.out.println(" gagne " + employes[i].getSalaire() +
" F");
        }
    }
}

```