BPOO - Sujet TD 8

Dut/Info-S2/M2103

Table des matières

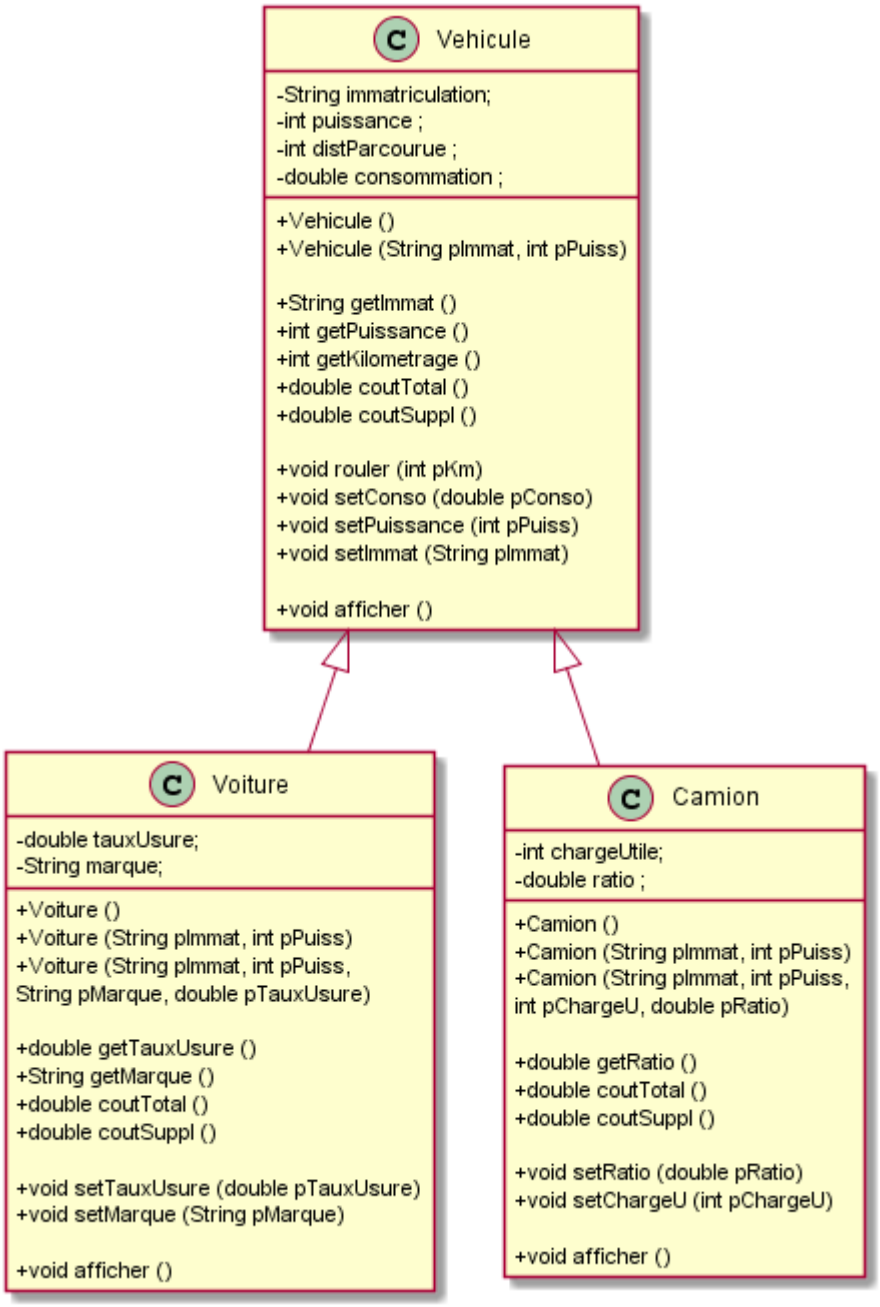
PreReq	Cours 4 : Principes de l'héritage et Héritage en Java. TP7 : Héritage de figures.
ObjTD	**Comprendre l'héritage des méthodes - Classes abstraites.**
Durée	**1 séances** de 1.5h.

1. Domaine d'étude

On va s'intéresser aux mécanismes d'héritage mis en oeuvre dans la hiérarchie de classes suivante.

Le code de la classe Vehicule vous est donné.

Diagramme UML des classes mises en oeuvre (sans multiplicités)



---

2. Constructeurs

Question : Écrire les constructeurs des classes Voiture et Camion. On écrira un "minimum" de code.

Les valeurs par défaut seront "--" pour les chaînes de caractères, 0 pour les numériques.

---

3. Méthodes coutTotal() et coutSuppl()

Les spécialistes donnent les spécification suivantes :

1. Classe Vehicule
    a. Le coût standard d'un Vehicule est de 15% de la distance parcourue fois la consommation.

2. Classe Voiture
    a. Le coût standard d'une Voiture est l'addition de :
        - coût standard d'un véhicule,
        - un forfait de coût de 0.85 fois la distance parcourue,
        - le coût supplémentaire particulier aux voitures.
    b. Le coût supplémentaire particulier aux voitures est la distance parcourue multipliée par le taux d'usure (en pourcentage).
3. Classe Camion
    a. Le coût standard d'un Camion est l'addition de :
        - coût standard d'un véhicule,
        - un forfait de 1.35 fois la distance parcourue,
        - le coût supplémentaire particulier aux camions.
    b. Le coût supplémentaire particulier aux camions est la distance parcourue multipliée par le ratio (en pourcentage) et la charge utile.

On souhaite faire fonctionner le code suivant :

```
Vehicule tabv[] = new Vehicule [4];

tabv[0] = new Voiture ("2222 VV 22", 5);
tabv[1] = new Camion ("2222 CC 22", 12);
tabv[2] = new Voiture ("3333 VV 33", 6, "Opel", 10);
tabv[3] = new Camion ("3333 CC 33", 14, 1000, 2);

for (int i=0 ; i<4 ; i++) {
        tabv[i].afficher();
        System.out.println (tabv[i].coutTotal());
        System.out.println (tabv[i].coutSuppl());
}
```

Questions :

1. Écrire les méthodes coutTotal() et coutSuppl() dans les 3 classes.
2. Nous n'avons pas de définition pour le coût supplémentaire particulier aux véhicules. Il n'existe pas. Doit-on déclarer la méthode coutSuppl() dans la classe Vehicule ? Comment l'écrire ?

4. Classes abstraites

On peut constater plusieurs éléments :

1. Point de vue client (garagiste, …) : un objet Vehicule n’existe pas ⇒ seules existent vraiment une voiture ou un camion.
2. Point de vue client (garagiste, …) : on parle des véhicules en général pour signifier un ensemble de voitures et camions (le parc du garage, …).
3. La classe Vehicule :
   a. Point de vue fonctionnel : généralise des attributs et méthodes communs à tous les véhicules (voitures et camions).
   b. Point de vue ensembliste : permet de **désigner** "indifféremment" des voitures et des camions.

Question :

1. Déclarer la classe Vehicule abstraite.
2. Modifier la déclaration de coutSuppl() dans Vehicule en conséquence.

Dernière mise à jour 2014-04-05 18:30:38 CEST