Durée : 2h, tout document autorisé

Sommaire

MO1 1
Mise en place 1
Création du workspace et import du projet 1
Programmation 2
Utilisation de la classe String 2
Création de la classe Voiture 2
Héritage et interface 3
Restitution du travail 5

MO1

Mise en place

Nous allons commencer par mettre en place notre espace de travail

Vous trouverez tous les documents nécessaires à l'url suivante http://jf.berdjugin.free.fr/examLP

Création du workspace et import du projet

Nous allons créer un workspace sur le partage netBios (z :), importer le projet et le renommer :

1. Créer les répertoires « z:\m01 » et « E:\m01\workspace »
2. Télécharger « exam_project .zip»
3. Importer le projet exam_project (exam_project.zip) avec File->Import->General->Existing projects into workspace -> select archive file.
4. Renommer votre projet avec refactor -> rename en exam_nom où nom est votre nom[1].

[1] Le nom du projet ne doit pas contenir d'espaces ou d'accents.

Programmation

Les trois partis qui suivent sont indépendantes.

Utilisation de la classe String

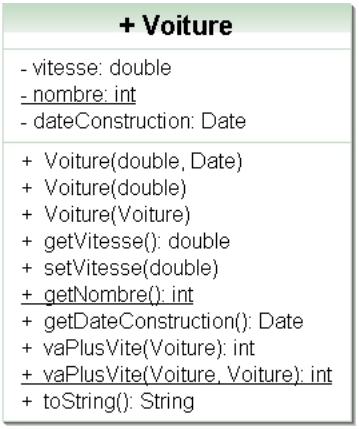
Vous disposez de la classe « TestString » que vous devez compléter le but est de partir de la chaîne de caractères "L'Isle-d'Abeau est une commune de France, à 30 km au sud-est de Lyon dans la plaine du Dauphiné." pour obtenir la chaîne "L'Isle-d'Abeau est une commune du Dauphiné" et effectuer la comparaison.

Voici la démarche proposée :

1. Créer à partir de *s* une chaîne *s1* qui est la sous-chaîne de *s* commençant à « du ».
2. Créer à partir de *s1* une chaîne *s2* qui est la sous-chaîne de *s1* ne contant pas le « . ».
3. Créer à partir de *s* une chaîne *s3* qui est la sous-chaîne de *s* se terminant à « de ».
4. Créer *s4* qui la chaîne résultant de la concaténation de *s3* et *s2*.
5. Comparer *s4* et *res*, afficher « oui » si les chaînes sont égales et afficher « non » sinon.

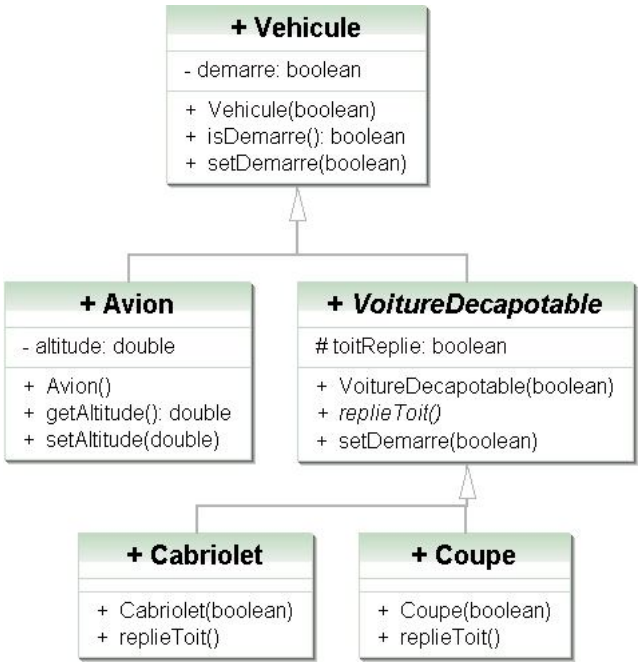
Création de la classe Voiture

La classe « TestVoiture » vous est fournie, en respectant la javadoc compléter la classe « Voiture » pour qu'elle permette au fichier « TestVoiture » de produire le résultat attendu.



Héritage et interface

La classe Véhicule vous est donnée.



Héritage

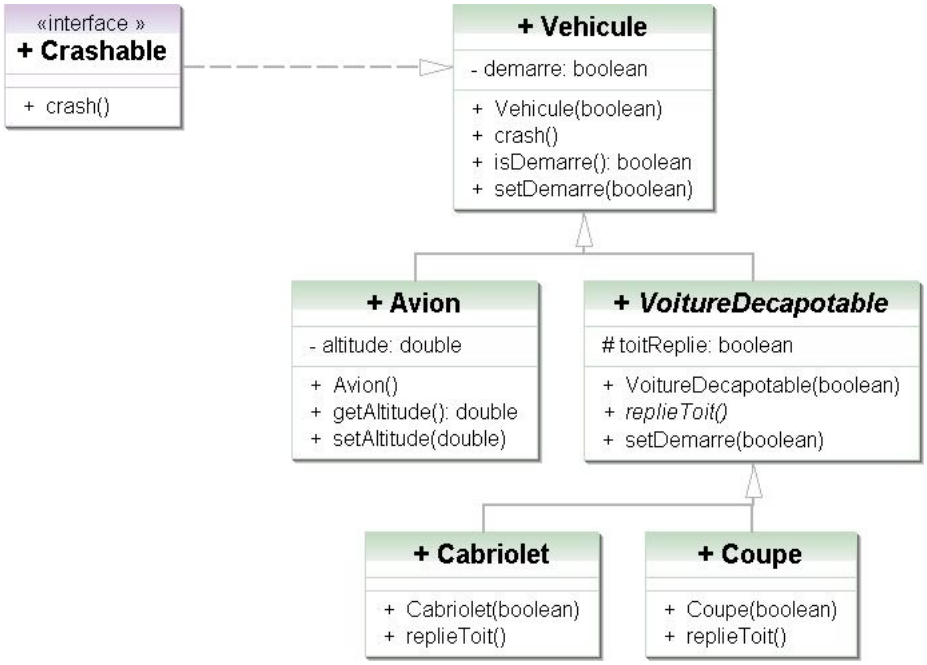
Créer une classe « Avion » qui hérite de « Vehicule » et qui respecte la javadoc.

Classe Abstraite

Nous allons en respectant la javdoc créer une classe « VoitureDecapotable » qui hérite de « Vehicule » et qui a deux classes filles « Cabriolet » et « Coupe ». Dans la classe VoitureDecapotable la méthode « replieToit() » est abstraite. La méthode « **public void** setDemarre(**boolean** demarre) » redéfinie la méthode du même nom de « Vehicule » pour replier la capote avant de démarrer ou d'arrêter.

Interface

En utilisant l'interface « crashable », rendre le « Vehicule » crashable, le message le véhicule est mort doit-être affiché lors du crash. Au final vous devez avoir :



Polymorphisme

Compléter la classe TestPolymorphisme pour replier le doit d'un coupé et d'un cabriolet.

Restitution du travail

Pour restituer votre travail vous devez exporter votre projet puis le déposer par FTP (ftp-exam.src) dans jberdjug/LP.

Pour l'export :

1. Sélectionner le projet (exam_nom)
2. Rafraichir F5
3. Bouton droit-> general -> archive file e:\nom.zip (nom étant votre nom)

Pour le FTP :

1. Déposer
2. Vérifier la taille 92Ko