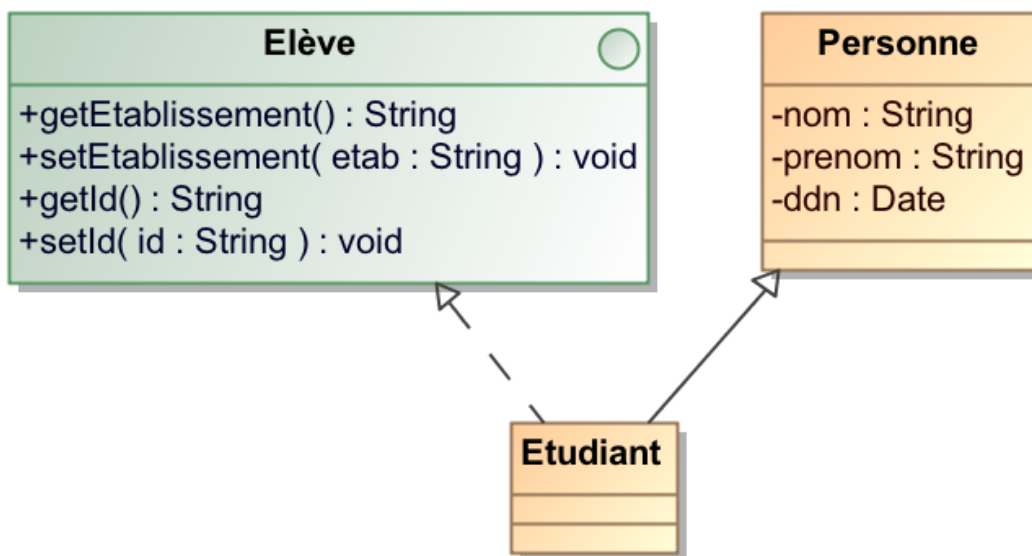


Objets avancés

Collections, Listes et String

Pour cette partie, nous allons prendre en compte les classes suivantes :



Collections

Considérons une `Collection<Personne>`

Quel(s) type(s) d'objet(s) peut contenir cette collection ?

Quels sont les attributs et méthodes disponibles pour les objets de cette collection ?

Donnez le code Java permettant d'afficher les noms des personnes de la collection.

Considérons une `Collection<Elève>`

Quel(s) type(s) d'objet(s) peut contenir cette collection ?

Quels sont les attributs et méthodes disponibles pour les objets de cette collection ?

Donnez le code Java permettant d'afficher toutes les informations disponibles sur les objets contenus dans cette collection.

Dans quelle(s) condition(s) est-il possible d'appeler la méthode `getId()` sur les objets contenus dans une `Collection<Personnes>`

Objets avancés

Listes

Réécrivez les parcours en Java, mais cette fois pour des `Liste<Personne>` et des `Liste<Elève>`.

String

Le travail suivant est inutile, mais a une portée pédagogique ^^

Créez une chaîne de caractères ressemblant à cela :

Etudiant : <prénom> <Nom>(id =<ID>) ; Personne :<prénom> <Nom> ;

Donnez le code qui retourne cette chaîne de caractères en fonction d'une `Collection<Personne>` passée en paramètres.

Faites ensuite une méthode qui prend la chaîne créée en paramètre et produit l'affichage suivant :

Etudiant :

Prénom : <Prénom>

Nom : <Nom>

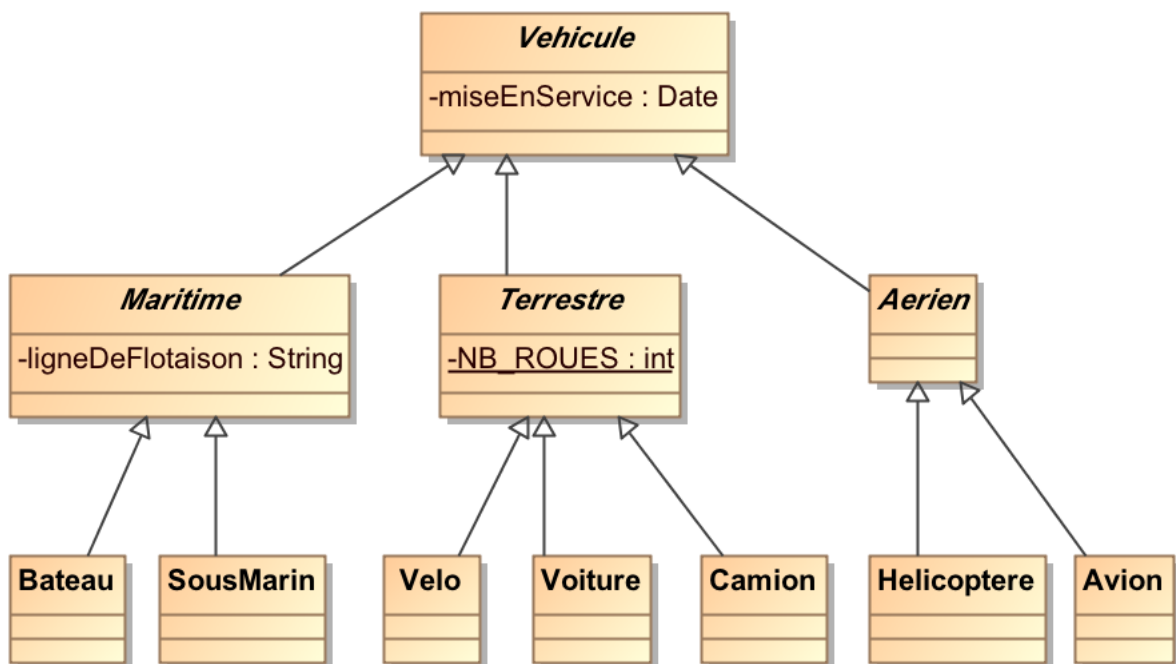
Id : <Id>

Personne :

Prénom : <Prénom>

Nom : <Nom>

Objets avancés



Transport

En vous basant sur le schéma ci-dessus, explicitez les méthodes et les attributs (en précisant si ce sont des attributs/méthodes de classe ou d'instance) des classes Bateau, Camion et Avion.

Complétez les classes afin que l'on puisse spécifier le nom du constructeur pour chaque type de véhicule.

Ajouter les méthodes suivantes à l'endroit le plus approprié :
démarrer(), enAvant(), détruire(), plonger().

Créez une méthode qui affiche la classe de chaque véhicule contenu dans une Collection de véhicules passée en paramètre.

Sans tout refaire, on voudrait qu'il soit possible de peindre tous les véhicules. Mais chaque véhicule doit pouvoir avoir une couleur différente. Attention, la faculté d'être peint doit aussi pouvoir s'appliquer à des bâtiments. Proposez une solution.

Formes

Déclarez une classe abstraite *Forme*

Une forme possède obligatoirement une méthode qui retourne son nom.

Créez des *Formes2D*. Les *Formes2D* ont une méthode de calcul d'aire et de périmètre.

Ajouter une interface *Coloriable*. Déclarez les classes *Rectangle* et *Carré*, puis leurs semblables coloriables.

De la même façon ajoutez les classes *Ellipse* et *Cercle*.

Enfin, créez des *Formes3D*, *Ellipsoïde*, *Cylindre*, *Pavé*, et *Cube* en vous appuyant sur les formes 2D existantes.

Les *Formes3D* ont une méthode supplémentaire donnant le volume.

Faites des tests et des affichages permettant de suivre le déroulement de votre programme.