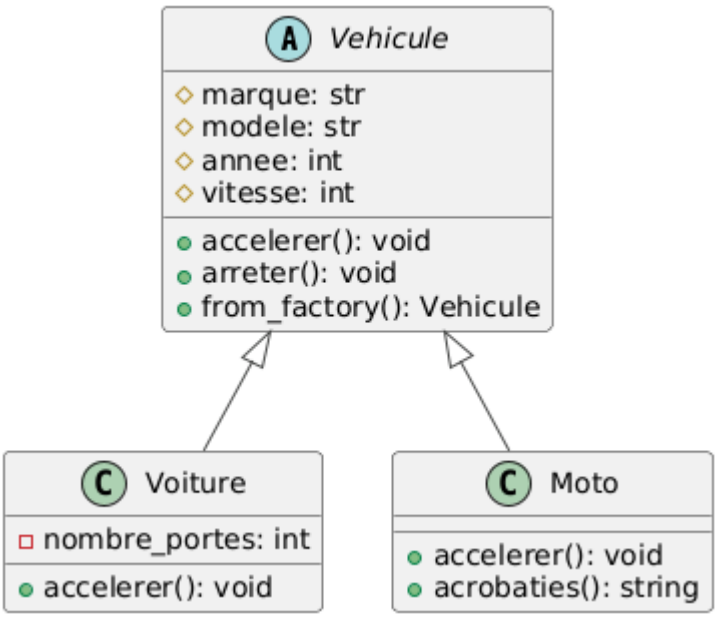
L'objectif de cet exercice est d'appliquer les concepts de la Programmation Orientée Objet en utilisant le design pattern Factory. Vous devrez créer quatre classes : `Vehicule`, `Voiture` et `Moto`.

Représentation UML

- © représente une classe
- Ⓐ représente une classe abstraite
- □ représente un attribut private
- ◊ représente un attribut protected
- ○ représente une méthode public



> [!TIP] Visibilité d'attributs Quand on entend protected ou private, on pense getter et setter

Classe Vehicule

La classe Abstraite `Vehicule` doit avoir les caractéristiques suivantes :

- Attributs :
  - Marque
  - Modèle
  - Année
  - Vitesse (avec une valeur par défaut de 0)
- Méthodes :
  - `accelerer`, ABSTRAITE, permet d'augmenter la vitesse
  - `arreter`, qui défini la vitesse 0

- `from_factory`, qui récupère tous les paramètres (*args; **kwargs) pour créer des classes enfants

> 📌 La méthode abstraite `accelerer` devra être implémentée dans les classes filles (`Voiture` et `Moto`).

Classe Voiture

La classe `Voiture` hérite de la classe `Vehicule` et possède un attribut supplémentaire :

- Attribut :
  - Nombre de portes (`nb_portes`) de type int

Classe Moto

La classe `Moto` hérite également de la classe `Vehicule` et a une méthode spécifique :

- Méthode :
  - `acrobaties`, affiche la chaine de caractère: "Wheel-in!"