POO (Programmation orientée objet)

Introduction générale:

Le langage PHP permet d'utiliser la programmation orientée objet, avec toutefois quelques spécificités par rapport à d'autres langages comme Java ou C#. Nous allons faire un rapide tour d'horizon des possibilités de PHP en matière de POO au travers de l'exemple classique des comptes bancaires.

1. Les principes fondamentaux de la POO:

Il existe plusieurs concepts clés dans la programmation orientée objet que vous devrez comprendre en PHP, en

particulier lorsque vous travaillez avec des classes et des objets :

a. Encapsulation

L'encapsulation en programmation signifie protéger les données d'un objet et y accéder uniquement via des méthodes. En PHP, on utilise trois niveaux de visibilité :

- public : Accessible de partout.
- protected : Accessible dans la classe et ses classes dérivées.
- private : Accessible uniquement dans la classe elle-même.

b. Héritage

L'héritage permet à une classe de hériter des propriétés et méthodes d'une autre classe. Cela permet de créer des relations entre les objets et de réutiliser le code de manière plus efficace.

c. Polymorphisme

Le polymorphisme permet à des objets de classes différentes de répondre de manière spécifique à un même message (appel de méthode). Cela permet de

traiter des objets de classes dérivées comme s'ils étaient du type de la classe de base.

d. Abstraction

L'abstraction consiste à définir des classes et des méthodes de manière à masquer les détails de l'implémentation et à ne laisser visibles que les fonctionnalités essentielles. Cela se fait généralement en utilisant des classes et des méthodes abstraites.

En PHP, pour créer une classe, on utilise le mot-clé `class`, et pour créer un objet à partir de cette classe, on utilise le mot-clé `new`.

2. Les classe

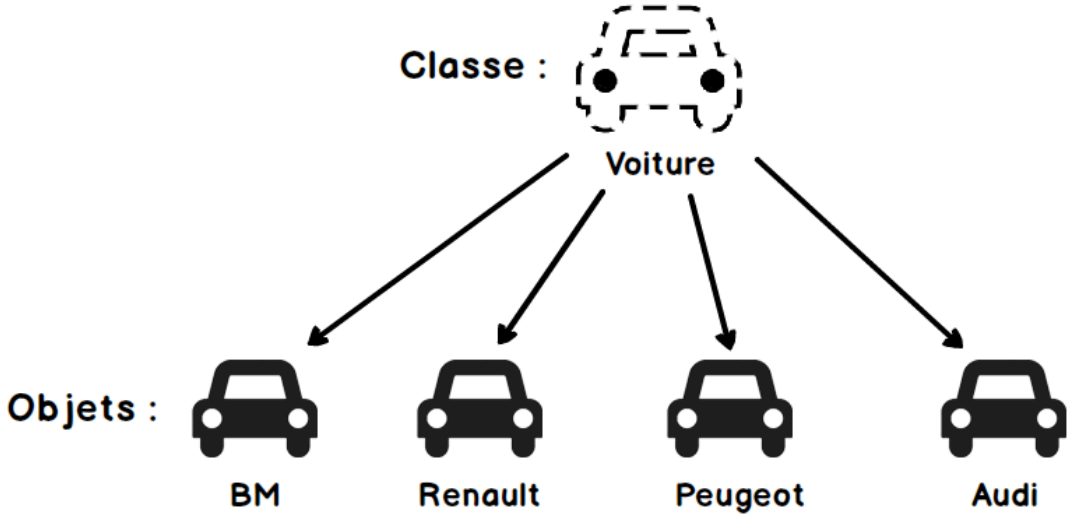
Qu’est ce qu’une classe?

Une classe est un plan ou un prototype défini par l’utilisateur à partir duquel des objets sont créés. Il

représente l’ensemble des propriétés ou méthodes communes à tous les objets d’un type.

Propriétés : Ce sont des variables qui définissent l'état ou les caractéristiques d'un objet.
Méthodes : Ce sont des fonctions qui définissent le comportement des objets.

Par exemple :



3. Les Objets

Un objet est une instance d'une classe. Il est créé à partir de la classe et possède ses propres valeurs pour les propriétés. Les objets peuvent interagir entre eux en utilisant leurs méthodes.

1-1.Création d'un objet à partir d'une classe :

```php
$maVoiture = new Voiture();
$maVoiture->marque = "Peugeot";
$maVoiture->couleur = "Rouge";
$maVoiture->démarrer(); // Affiche "La voiture démarre."
```

Ici, `$maVoiture` est un objet de la classe `Voiture`. L'opérateur `->` est utilisé pour accéder aux propriétés et méthodes de l'objet.

Modificateurs d'accès en PHP

En PHP, il existe trois types de modificateurs d'accès :

- `public` : Indique que la propriété ou la méthode est accessible de n'importe où.

- `private` : Indique que la propriété ou la méthode est accessible uniquement à l'intérieur de la classe elle-même.
- `protected` : Indique que la propriété ou la méthode est accessible à l'intérieur de la classe et dans les classes qui en héritent.

Public : Accessible de partout (même à l'extérieur de la classe)

Protected : Accessible dans la classe et ses sous-classes (mais pas de l'extérieur)

Private : Accessible uniquement à l'intérieur de la classe (pas même dans les sous-classes)

➔ Accès aux propriétés et méthodes avec des getters et setters

Afin de respecter pleinement l'encapsulation, il est courant d'utiliser des getters et setters pour accéder ou modifier les valeurs des propriétés privées ou protégées d'une classe.

Example:

```
class Vehicle {
   private $_color, $_brand;

   public function getColor() { return $this->_color; }
   public function setColor($color) {
    $this->_color = $color;
}

   public function getBrand() { return $this->_brand; }
   public function setBrand($brand) {
    $this->_brand = $brand;
}
}
```

- **Les Getters (Accesseurs)** : méthodes qui permettent de connaître les valeurs des attribut.
- **Les Setters(modificateurs)** : sont des méthodes qui permettent de modifier les attributs

e.Héritage

L'héritage permet à une classe (appelée classe fille ou dérivée) de réutiliser et étendre les fonctionnalités d'une autre classe (appelée classe parent).

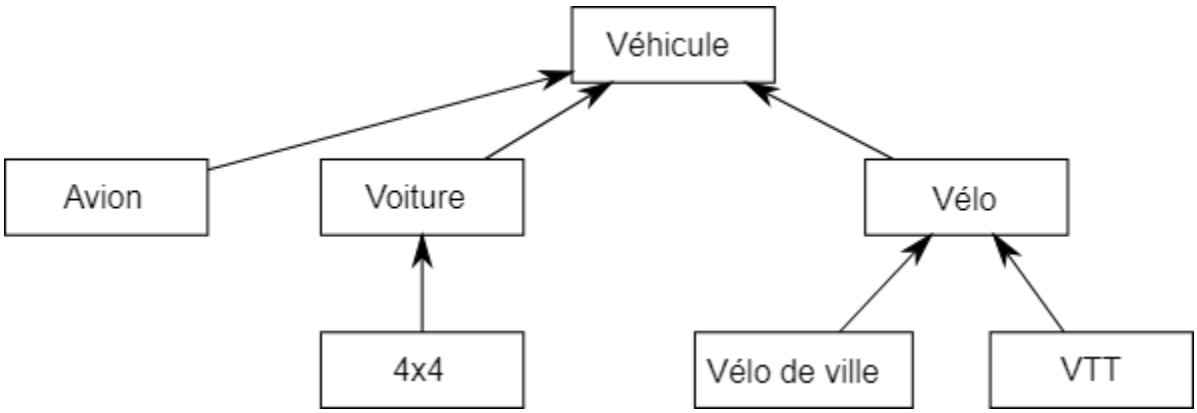
Points clés :

Utilisation du mot-clé `extends` :

- Pour créer une classe fille, il faut utiliser le mot-clé `extends` suivi du nom de la classe parent.

Héritage des propriétés et méthodes :

- Une classe fille hérite automatiquement des **propriétés** et **méthodes publiques et protégées** de la classe parent.
- Les méthodes privées, en revanche, ne sont pas accessibles directement par la classe fille.



f. Polymorphisme

Le **polymorphisme** est la capacité d'une méthode d'avoir des comportements différents en fonction de la classe qui l'implémente. En PHP, cela se fait souvent par la redéfinition des méthodes (méthodes override).

- La classe fille peut redéfinir une méthode de la classe parent avec le même nom.
- L'utilisation des interfaces ou des classes abstraites permet également d'exploiter le polymorphisme.