Module PHP

Cours 9

Fouziya Kouddane

SOMMAIRE

- Introduction à la POO
- MVC
- Namespaces
- Exercices

Introduction à la POO

La programmation orientée objet ou POO est une méthode de programmation. Elle est considéré comme plus puissant et plus rapide pour certaines tâches que la manière dite procédurale vu jusqu'à présent.
La POO aide à créer et à gérer facilement des tâches. Voici quelques avantages de la POO:

- Facile à gérer
- Facile à utiliser
- Empêche la répétition
- Rapide et efficace

La POO se base sur 4 points principaux:

- Classe
- Objet
- Propriétés
- Méthodes (ou fonctions)

Introduction à la POO

En POO, une tâche ou un élément est géré par une classe .

Mais alors qu'est-ce qu'une classe?

Une classe est un plan, il s'agit d'un morceau de code décrivant comment gérer un élément ou une tâche.

Par exemple, une classe est comme le plan d'une maison.

Il est possible de construire plusieurs maisons à partir d'un plan, de la même manière, il est possible de créer plusieurs objets à partir d'une classe .

Qu'est-ce qu'un objet?

Un objet est une instance de classe, c'est-à-dire un comportement correspondant à cette classe .

Reprenons notre exemple de plan de maison, nous pouvons créer plus d'un objet à partir d'une classe, comme nous pouvons créer plusieurs maisons à partir d'un plan. Chaque maison peut avoir sa propre couleur, ses types de fenêtres, portes, son équipement domestique et plus encore. De la même manière, les différents objets peuvent avoir des propriétés différentes .

Introduction à la POO

Qu'est ce que les propriétés?

Ceux sont des variables qui caractérisent un objet, ce sont les valeurs associées à l'objet, elles servent à décrire l'apparence de l'objet.

Les propriétés peuvent être ajoutées, modifiées, supprimées. Certains peuvent également être en lecture seule.

La couleur est une propriété de notre maison.

Nous pouvons également effectuer des actions sur notre maison (comme changer la couleur).

Qu'est ce qu'une méthode?

Les méthodes sont des actions effectuées sur des objets. Changer la couleur de ma maison est une méthode effectuée sur mon objet de maison.

Introduction à la POO

Comment créer un objet?

Le mot clé class suivi du nom de la classe est utilisé pour déclarer une classe.

Exemple:

```
<?php
    class House {
        // code
    }
```

Dans cet exemple House est le nom de la classe.

Les classes peuvent contenir des variables qui sont des propriétés .

Dans les classes, avant de déclarer une variable, nous devons ajouter le mot clé pour la visibilité pour définir où la variable est disponible. Ce point sera vu plus tard gardez juste à l'esprit que l'ajout du mot-clé public devant la variable rendra la variable disponible partout.

Introduction à la POO

Exemple

```
<?php
    class House {
        public $couleurPrincipale= 'white';
        public $couleurSecondaire= [salle_de_bain => 'white',
                'chambre => 'light blue',
                'cuisine' => 'beige'];
        public $extra;
     }
```

Comme dans l'exemple ci-dessus, nous pouvons déclarer tout type de variable en tant que propriété. Nous pouvons ajouter des valeurs par défaut pour ces propriétés. (Le white est la valeur par défaut pour $couleurPrincipale). Notez également que la propriété $extra n'a pas de valeur par défaut.

Introduction à la POO

Nous pouvons créer plusieurs objets à partir d'une classe. Ces objets sont appelés instances de la classe. Ce processus est appelé instanciation .
Après la déclaration de classe, nous pouvons créer des instances à partir de celle-ci.

```
$myHouse = new House();
$friendHouse = new House();
```

Maintenant, chaque objet est créé avec les valeurs par défaut de la classe. Pour le vérifier on peut faire un echo à la couleur de chaque objet.

```
echo $myHouse -> primaryColor;
echo $friendHouse -> primaryColor;
```

Les deux feraient écho à la valeur par défaut white .
-> est l' opérateur objet , qui est utilisé pour accéder aux propriétés et aux méthodes d'un objet.
Note: Lors de l'accès aux propriétés et aux méthodes d'un objet via l'opérateur d'objet, le signe $ n'est pas utilisé avec le nom de la propriété ou de la méthode.
On peut aussi modifier les valeurs de propriété par défaut de la classe pour rendre chaque maison unique:

Introduction à la POO

$myHouse -> couleurPrincipale= 'red';
$friendHouse -> couleurPrincipale = 'yellow';

Maintenant, la couleur de ma maison est rouge et la couleur de la maison de friend est jaune.
Des méthodes sont utilisées pour effectuer des actions.
Dans la programmation orientée objet en PHP, les méthodes sont des fonctions à l'intérieur des classes. Leur déclaration et leur comportement sont presque similaires aux fonctions normales, à l'exception de leurs utilisations spéciales à l'intérieur de la classe.

Comment déclarer une méthode?

Déclarons une méthode dans une classe nommée classe Exemple pour faire un echo à une Simple chaîne de caractères.

```
<?php
        class Exemple {
            public function echo($string) {
                    echo $string;
            }
        }
```

Introduction à la POO

Comment appeler une méthode?

```
$exemple = new Exemple();
$exemple -> echo('Hello World');
```

Résultat: Hello World

Tout d'abord, nous créons un objet ($ exemple) à partir de la classe exemple
Ensuite, nous appelons la méthode echo avec -> (opérateur objet) et (), les parenthèses contiennent les arguments.

Ce qu'il faut comprendre ici c'est que nous appelons des méthodes sur des objets, non pas sur des classes .

Introduction à la POO

Modification d'une valeur de propriété à l'aide de méthodes

```
<?php
    class House {
        public $couleurPrincipale= 'white';
        public function changeColor($color) {
            $this -> couleurPrincipale = $color;
        }
     }
     // création de l'objet house à partir de la classe
    $myHouse = new House();

    echo $myHouse -> couleurPrincipale;

    // changer la couleur de la maison
     $myHouse -> changeColor('black');
    echo $myHouse -> couleurPrincipale;
```

Rappel: $ this fait référence à l' objet courant .

MVC

Qu'est ce que c'est?

Le MVC est ce que l'on appel un Design Pattern (modèle de conception), MVC pour Model View Controller.

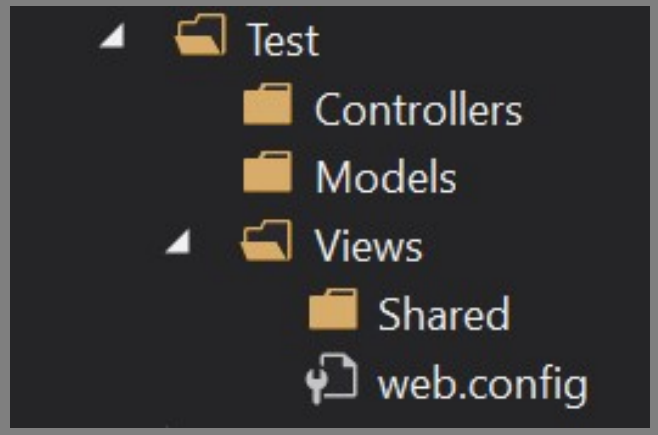
Le principe du MVC est de scinder les différentes partie du code sous forme de dossiers distincts comme suit:

- Le Modèle représente les informations (les données) de l'application.
- La vue correspond à des éléments de l'interface utilisateur tels que du texte, des éléments de case à cocher, etc.
- Le contrôleur gère la communication des données et les règles définies pour manipuler les données vers et depuis le modèle.

  Lorsque le client demande à afficher une page, le contrôleur appellera le ou les modèles de données nécessaires à la vue qui sera retourné à l'affichage côté client.

MVC

L'implémentation du mvc se compose du fichier *index.php* et de plusieurs fichiers placés dans les répertoires *model, view* et *controller* :



NAMESPACES

Les namespaces permettent d'avoir plusieurs classes avec le même nom. En PHP, il est n'est pas possible d'avoir des classes du même nom, c'est pour cela que l'on utilise les namespaces qui permettent de simplifier le processus et le rendre plus organisé .

Le concept de définition et d'appel des namespaces est similaire à la structure de fichier normale de nos systèmes d'exploitation.

Considérez les espaces de noms comme des dossiers et les fichiers comme des classes.

Le dossier / lib / math peut contenir plusieurs fichiers, nombres.php, calculus.php et ainsi de suite. Mais, pas de fichiers multiples avec le même nom.

A l' intérieur du dossier de la physique, si vous utilisez numbers.php directement , il fera référence à /lib/physics/numbers.php

Et donc les namespaces fonctionnent de la même manière.

NAMESPACES

Namespaces global

Lorsque vous déclarez une classe (ou une fonction ou une constante) sans espace de noms, elle peut être appelée "un élément dans le namespace global" .

Comment définir les namespaces?

Le mot-clé namespace suivi du nom du namespace est utilisé pour le déclarer.

Exemple:

```
<?php
    namespace MyApp;
?>
```

Le nom du namespace doit être un identifiant PHP valide. La déclaration du namespace doit être la première chose dans un fichier PHP. (à l'exception du mot-clé declare qui sera vu plus tard).

NAMESPACES

Créons un fichier nommé Math.php et ajoutons le code suivant.

Math.php

```
<?php
    namespace Math;
    function add($a, $b) {
        return $a + $b;
    }
    const PI = 3.14;
    class Geometry {
        static function getCircleArea($radius) {
            return PI * $radius ** 2;
        }
    }
```

NAMESPACES

Maintenant créons un autre fichier appelé usage.php et accédons aux éléments du namespace Math.

```
<?php
    include_once 'Math.php';
    echo Math\add(2,3); // 5
    echo Math\PI;  // 3.14
?>
```

Le mot clé use peut être utilisé pour importer une classe dans la portée actuelle.
Il s'utilise comme ceci:

```
<?php
    use Math\Circle;
    $circle = new Circle(10);
?>
```

Notez qu'il est possible d'importer une classe mais aussi un namespace avec use.