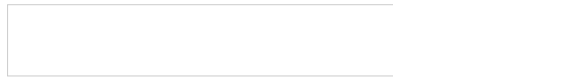


# TUTOWEBDESIGN

Tutoriels web pour la création de sites dédiés à l'internet et au web mobile



## POO, manipuler les objets

La **Classe** est la définition abstraite d'un objet. Elle définit la structure de chaque objet qu'on va pouvoir créer. C'est comme un moule qu'on va utiliser pour produire les objets. Chaque objet va donc posséder la même structure que le 'moule'.

### Créer un objet

Lorsque que la classe est déclarer (le moule est prêt), vous pouvez l'utiliser pour créer des objets. On dit que la **classe** instancie l'**objet**

L'opérateur **new** permet d'instancier un objet.

```
1. <?php
2. class Voiture{
3.     public $marque = "peugeot";
4.     public function demarrage(){
5.         echo "La voiture de marque ".$this->marque." démarre"; // la marque de lui-même
6.     }
7. }
8. $maVoiture = new Voiture(); // un objet
9. ?>
```

Il est possible de créer plusieurs objets avec la même classe.

```
1. <?php
2. class Voiture{
3.     public $marque = "peugeot";
4.     public function demarrage(){
5.         echo "La voiture de marque ".$this->marque." démarre"; // la marque de lui-même
6.     }
7. }
8. $maVoiture = new Voiture();
9. $uneAutreVoiture = new Voiture(); // nouvel objet
10. ?>
```

Les deux objets sont identiques (possèdent les mêmes propriétés et méthodes à la création) mais ce ne sont pas les mêmes (ils peuvent être manipulés différemment).

### Accéder à une propriété

On peut accéder à une propriété de l'objet avec la syntaxe -> à appliquer sur l'objet. La propriété est appelée sans la signe \$.

```
1. <?php
2. class Voiture{
3.     public $marque = "peugeot";
4.     public function demarrage(){
5.         echo "La voiture de marque ".$this->marque." démarre"; // la marque de lui-même
6.     }
7. }
8. $maVoiture = new Voiture();
9. echo $maVoiture->marque;
10. ?>
```

Affichage

peugeot

Vous remarquez qu'on peut accéder à la même propriété au sein de la classe – dans la méthode demarrage() – avec la variable **\$this**.

```
1. $this->marque
```

Le **\$this** fait référence à l'objet lui-même.

### Accéder à une méthode

On peut accéder à une méthode de l'objet avec la syntaxe -> à appliquer sur l'objet. La méthode est appelée sans la signe \$ et avec les parenthèses ().

```
1. <?php
2. class Voiture{
3.     public $marque = "peugeot";
4.     public function demarrage(){
5.         echo "La voiture de marque ".$this->marque." démarre"; // la marque de lui-même
6.     }
7. }
8. $maVoiture = new Voiture();
9. $maVoiture->demarrage(); // appel de la méthode
10. ?>
```

Affichage

La voiture de marque peugeot démarre

### Passage d'une instance en paramètre

Vous pouvez faire passer, en argument d'une méthode de classe, un objet de la **même classe** en mentionnant le nom de la classe, suivi d'un espace, puis du nom de la variable objet (l'instance).

```
1. <?php
```

```

2. class Voiture{
3.     public $marque = "peugeot";
4.     public function demarrage(){
5.         echo "La voiture de marque ".$this->marque." démarre";
6.     }
7.
8.     public function information(Voiture $objvoiture){ // instance de la classe en paramètre
9.         echo "Je suis une voiture de marque ".$this->marque;
10.    }
11. }
12. $maVoiture1 = new Voiture();
13. $maVoiture2 = new Voiture();
14. $maVoiture1->information($maVoiture2);
15. ?>

```

Affichage

```
Je suis une voiture de marque peugeot
```

Il s'agit d'une déclaration de type (ici une instance de la classe), le type doit être ajouté avant le nom du paramètre. Cette déclaration impose que le paramètre soit d'un certain type lors de l'appel de la méthode.

## Constante de débogage

A tout moment, dans une méthode, vous pouvez faire appel aux métaconstantes `__CLASS__` et `__METHOD__`. Elles retourneront les noms de la classe et de la méthode en cours d'exécution.

```

1. <?php
2. class Train {
3.     private $type;
4.     function avance(){
5.         echo 'classe : '.__CLASS__.'  
>methode : '.__METHOD__ ;
6.     }
7. }
8. $train = new Train();
9. $train->avance(); // affiche 'classe : Train - Train::avance'
10. ?>

```

Affichage

```
classe : Train
methode : Train::avance
```

Il existe d'autres constantes magiques que vous pouvez consulter sur [php.net](http://php.net)

## Vérifier le type d'un objet

L'opérateur **instanceof** retourne TRUE si un objet a pour parent une classe donnée ou une sous-classe donnée.

```

1. <?php
2. class Velo {
3. }
4. $velo = new Velo();
5. if ($velo instanceof Velo) {
6.     echo "C'est un vélo";
7. }
8. ?>

```

Affichage

```
C'est un vélo
```

Il est possible de connaître la classe exacte d'un objet et non pas son appartenance à une famille via la fonction **get\_class()**. En fournissant un objet en paramètre, elle renvoie une chaîne de caractères contenant le nom de la classe à laquelle appartient l'objet.

```

1. <?php
2. class Moto {
3.     function name() {
4.         echo "Mon nom est " . get_class($this);
5.     }
6. }
7. $moto = new Moto();
8.
9. // en externe
10. echo "Son nom est " . get_class($moto)."<br>";
11.
12. // en interne
13. $moto->name();
14. ?>

```

Affichage

```
Son nom est Moto
Mon nom est Moto
```