codecademy

Classes et objets en PHP

PHP étend le mot-clé

En PHP, pour définir une classe qui hérite d'une autre, on utilise le mot clé `extends` :

```
class ChildClass extends
ParentClass {
}
```

```
// Dog class inherits from Pet class.
class Dog extends Pet {
  function bark() {
    return "woof";
  }
}
```

La classe nouvellement définie peut accéder aux membres avec `public` et `protected` la visibilité de la classe de base, mais ne peut pas accéder aux `private` membres.
La classe nouvellement définie peut également redéfinir ou remplacer les membres de la classe.

Méthode du constructeur PHP

Une méthode constructeur est l'une des nombreuses méthodes magiques fournies par PHP. Cette méthode est appelée automatiquement lorsqu'un objet est instancié. Une méthode constructeur est définie avec le nom de méthode spécial `__construct` . Le constructeur peut être utilisé pour initialiser les propriétés d'un objet.

```
// constructor with no arguments:
class Person {
  public $favorite_color;
  function __construct() {
    $this->favorite_color = "blue";
  }
}
```

```
// constructor with arguments:
class Person {
  public $favorite_color;
  function __construct($color) {
    $this->favorite_color = $color;
  }
}
```

classe PHP

En PHP, les classes sont définies à l'aide du mot-clé `class`.
Les fonctions définies au sein d'une classe deviennent des méthodes et les variables au sein de la classe sont considérées comme des propriétés.
Il existe trois niveaux de visibilité pour les membres de la classe :

- `public` (par défaut) - accessible depuis l'extérieur de la classe
- `protected` - uniquement accessible au sein de la classe ou de ses descendants
- `private` - accessible uniquement dans la classe de définition

Les membres peuvent être définis comme étant statiques :

- Les membres statiques sont accessibles à l'aide de l'opérateur de résolution de portée (`::`)

Les classes sont instanciées dans des objets à l'aide du mot-clé `new`. Les membres d'un objet sont accessibles à l'aide de l'opérateur d'objet (`->`).

```
// Test class
class Test {
  public $color = "blue";
  protected $shape = "sphere";
  private $quantity = 10;
  public static $number = 42;
  public static function returnHello() {
    return "Hello";
  }
}

// instantiate new object
$object = new Test();

// only color can be accessed from the instance
echo $object->color; # Works
echo $object->shape; # Fails
echo $object->quantity; # Fails
echo $object->number; # Fails

// we use the static class to access number
echo Test::$number;
```