esprit
Se former autrement
HONORIS UNITED UNIVERSITIES

POO avec PHP7

Année universitaire
2022-2023

United Nations Educational, Scientific and Cultural Organization
uniTwin
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
EUR-ACE® Délivrée par la Commission des Titres d'Ingénieur
CONFÉRENCE DES GRANDES ÉCOLES
cdio CONCEIVE DESIGN IMPLEMENT OPERATE

Objectifs

- Rappel sur les classes et le POO en PHP7
- Appréhender les notions de l’orientée objet

Plan

- Notion des Classes en POO / PHP 7
- L'instanciation des Objets
- Constructeur/Déconstructeur
- Les propriétés et les droits d'accés
- L'encapsulation
- L'héritage et la redéfinition des méthodes

POO: Notion de classe (1/2)

- Une **classe** est une représentation abstraite d'un **objet**.
- Une classe peut être rendue concrète au moyen d'une **instance de classe**, que l'on appelle **objet.**
- Une classe s'écrit au moyen du mot "**class**" suivi du nom de la classe et d'accolades.

```
Class Personne
{
    public $nom;
     public $prenom;
     function  saisir($nom,$prenom);
     function afficher();
}
```

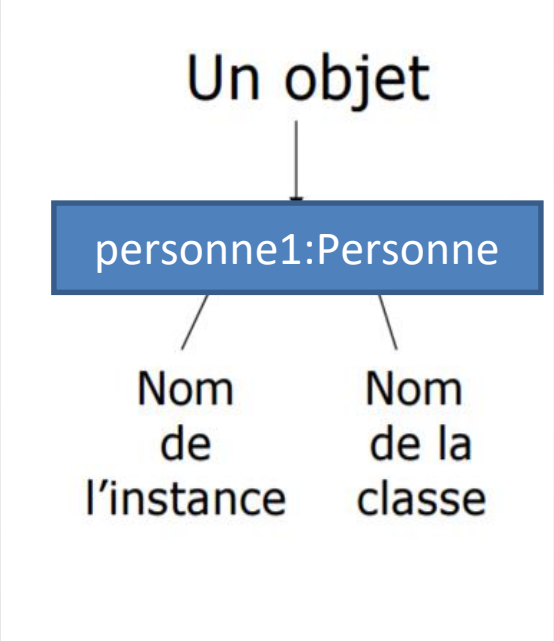
POO: Notion de classe (2/2)

Personne
+nom +prenom
+saisir(var,var) +afficher()

```php
<?php

 class Personne
 {

    Public $nom;
    Public $prenom;

    function saisir($nom,$prenom){
        $this->nom= $nom;
        $this->prenom= $prenom;
    }

    function afficher(){
        echo "Nom:".$this->nom."<br>";
        echo "Prenom:".$this->prenom."<br>";
    }
 }
?>
```

POO: instanciation d'un objet

- Exemple:




```php
//Declaration d'une instance
$personne1 = new Personne();
//Utilisation
$personne1->saisir("Foulen","Ben Foulen");
//Vérification
if($personne1 instanceOf Personne)
    echo "L'objet \$personne est de type Personne";
```

POO: Le constructeur

- Le constructeur est la méthode qui va être appelée à l'instanciation de l'objet.
- Il doit être implémenté dans la classe elle-même de la manière suivante :

```php
public function __construct(string $prenom, string $nom) {
  // on passe par les set
  $this->setPrenom($prenom);
  $this->setNom($nom);
```

- On est passés par les set, qu'on expliquera plus tard avec l'encapsulation,

POO: Le Déstructeur

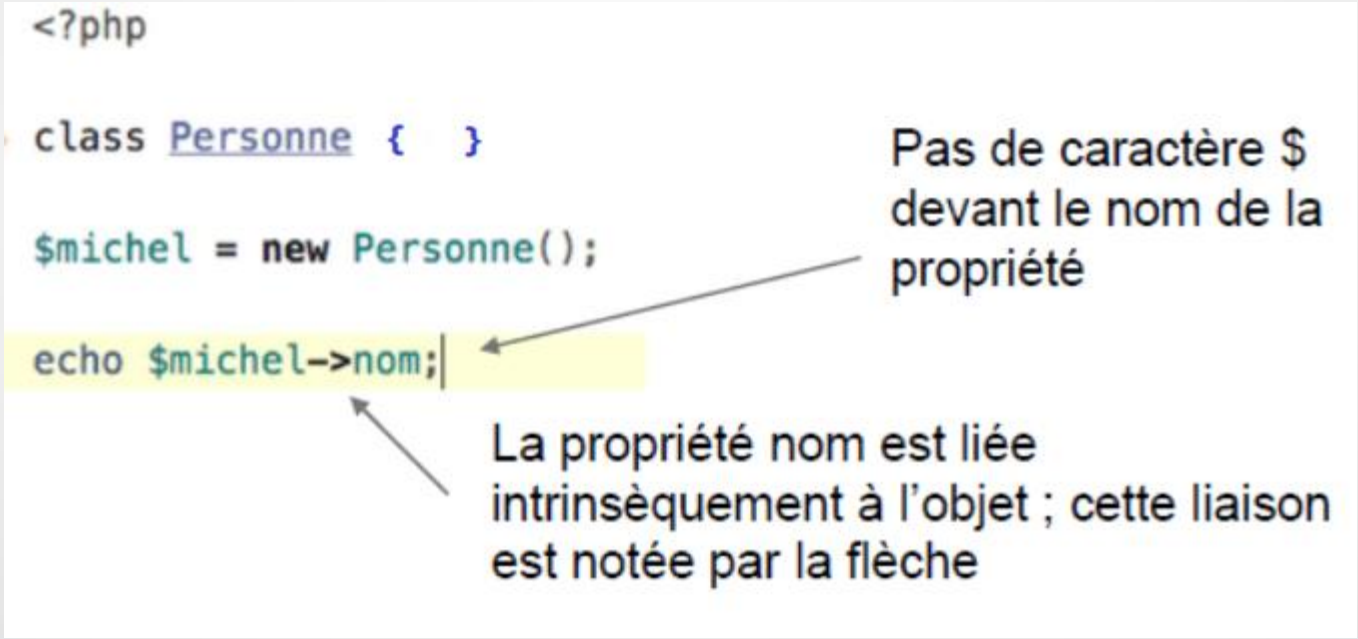
- Un destructeur d'une classe donnée est une méthode exécutée automatiquement à chaque fois qu'une instance de la classe donnée disparaît.
- C'est une méthode sans type de retour
- C'est une méthode sans paramètre, elle ne peut donc pas être surchargée ;
- C'est une méthode en accès public.

```php
public function __destruct(){

  echo "Je suis le destructeur";
}
```

```php
$personne1 = new Personne();
$personne1->afficher();
unset($personne1);
```

POO: Les propriétés d'un objet

- Une propriété est une variable associée à un objet.
- On peut demander « le nom de cette personne » et non le nom en général.




```php
<?php

class Personne {  }

$michel = new Personne();

echo $michel->nom;
```

POO: Les droits d'accès

- Les méthodes et les variables “ **(+)public**” sont visibles et manipulables par tous les objets, même s'ils sont relatifs à d'autres classes.
- Les méthodes et les variables “**(#)protected**” concernent les objets de la même classe ainsi que ses dérivés, mais pas ceux des classes étrangères.
- Les classes ont des variables et des méthodes internes et qui ne concernent pas l'extérieur. Ces propriétés sont déclarées en tant que “**(-)private**”.

POO: L'encapsulation

- L'encapsulation est la pratique consistant à regrouper des attributs au sein d'une même classe.

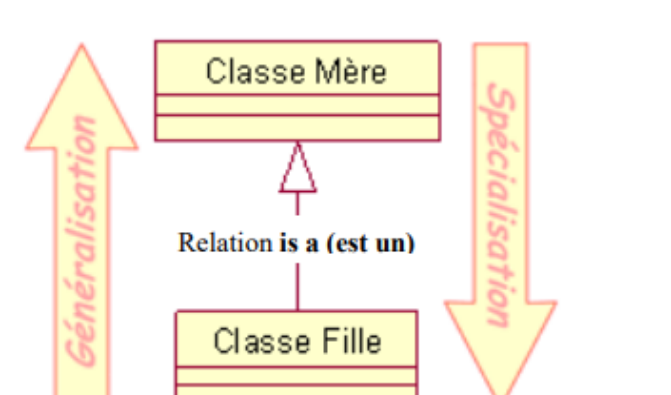
  - Pour améliorer la lisibilité des programmes, les attributs encapsulés sont souvent **privés** (inaccessibles aux autres classes)
  - Les données et méthodes accessibles sont dites **publiques**

POO: L'encapsulation

```php
<?php
class Personne {
// attributs de la classe
  private $prenom;
  private $nom;

// getters and setters
  public function getPrenom(): string {
    return $this->prenom;
  }

  public function getNom(): string {
    return $this->nom;
  }

  public function setPrenom(string $prenom): void {
    $this->prenom = $prenom;
  }

  public function setNom(string $nom): void {
    $this->nom = $nom;
  }
// constructeur
  public function __construct(string $prenom, string $nom) {
    // on passe par les set
    $this->setPrenom($prenom);
    $this->setNom($nom);
    $this->setAge($age);
  }
```

POO: L'héritage

- L'**héritage** consiste à définir différents niveaux d'abstraction permettant ainsi de **factoriser** certains attributs et/ou méthodes communs à plusieurs classes.
- Une classe générale définit alors un ensemble d'attributs et/ou méthodes qui sont partagés par d'autres classes, dont on dira qu'elles héritent de cette classe générale.



POO: La redéfinition

- PHP permet la redéfinition des méthodes, c'est-à-dire la possibilité de redéclarer les mêmes attributs et opérations d'une super classe au sein d'une sous classe. Nous pouvons aussi modifier la signature de la fonction, et son traitement.

```php
class DeuxRoues
{
        //Attribut
        protected $_couleur ;

        //Méthode
        public function getCouleur()
        {
          return $this->_couleur;
        }
}
```

```php
class Bicyclette extends DeuxRoues
{

        //redéfinition de la méthode
        public function getCouleur()
        {
           echo "Nouvelle méthode !";
           return $this->_couleur;
        }
}
```

```php
//Déclaration d'une instance :
$MonVelo = new Bicylette;

$MonVelo->getCouleur();
```

Appel de cette méthode

Références

https://stahe-php7.readthedocs.io/fr/latest/chap-05.html

https://www.php.net/manual/fr/index.php

https://flossmanualsfr.net/initiation-a-php/creer-des-objets/

https://www.w3schools.com/php/php_oop_classes_objects.asp