Programmation **O**rientée **O**bjet en PHP

- Méthodes magiques __get() et __set()
- Manipulation d'objets: clonage et comparaison
- Interactions entre objets : associations
- Opérateur de résolution de portée (::)

Les méthodes magiques: __get() et __set()

__get() est exécutée quand le script tente de lire une propriété déclarée 'private' ou 'protected'.

__set() est exécutée quand le script tente de modifier une propriété déclarée 'private' ou 'protected'.

```php
class MaClasse
{
    private $maVariable1;
    private $maVariable2;

    public function __get($prop) {
        return $this->$prop;
    }

    public function __set($prop, $val) {
        $setProperty = 'set'.ucfirst($prop);
        $this->$setProperty($val);
    }
```

```php
$monObjet = new MaClasse(999, 'azerty');

$monObjet->maVariable1 = 514;
$monObjet->maVariable2 = 'qwerty';
echo "maVariable1 = ".$monObjet->maVariable1."<br>";
echo "maVariable2 = ".$monObjet->maVariable2."<br>";
```

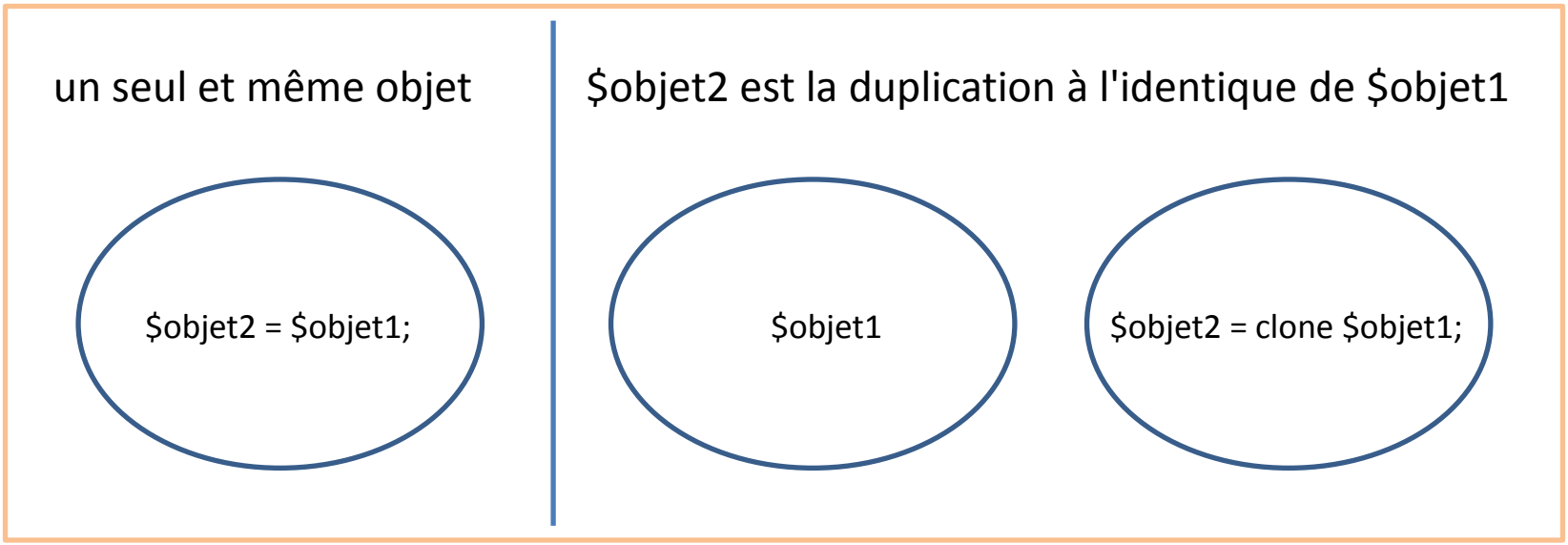
Exercice 6

Reprenez l'exercice 5 en ajoutant les méthodes magiques __get() et __set().

Provoquez l'exécution de ces méthodes pour modifier et afficher les propriétés des objets que vous créez.

Manipulation des objets: clonage (copie) d'un objet

ATTENTION : l'opérateur d'assignation (=) n'a pas le même effet que pour des variables de type alphanumérique.
L'assignation des objets se fait par référence.
Lorsque l'on utilise l'opérateur d'assignation, on n'opère pas de copie de l'objet (contrairement aux nombres et chaînes de caractères) mais on crée un nouveau pointeur sur l'objet.
Pour cloner un objet en dupliquant ses propriétés et ses méthodes, il faut utiliser le mot-clé '**clone**' avec la syntaxe: **$objet2 = clone $objet1;**



Manipulation des objets: clonage (copie) d'un objet

```
$objet2 = clone $objet1;
```


exécution de la méthode magique **__clone()**
si elle a été codée dans la classe

```
public  function  __clone()  {

    // code pour modifier des propriétés de l'objet cloné

}
```

Manipulation des objets: comparaison d'objets

Opérateur **==**

Cet opérateur vérifiera que deux objets ont les mêmes attributs et les mêmes valeurs, et qu'ils sont des instances de la même classe.

Opérateur **===**

Cet opérateur vérifiera si les deux objets font référence à la même instance.
Il vérifiera donc que les deux objets comparés pointent vers la même référence.

Exercice 7

Sur la base de l'exercice 6 :

Ajoutez un méthode magique __clone() pour préfixer la propriété 'nom' par 'clone de '.

Créez une classe PersonneBis identique à la classe Personne.

Créez des objets et manipulez ces objets en utilisant les opérateurs d'assignation et de clonage.

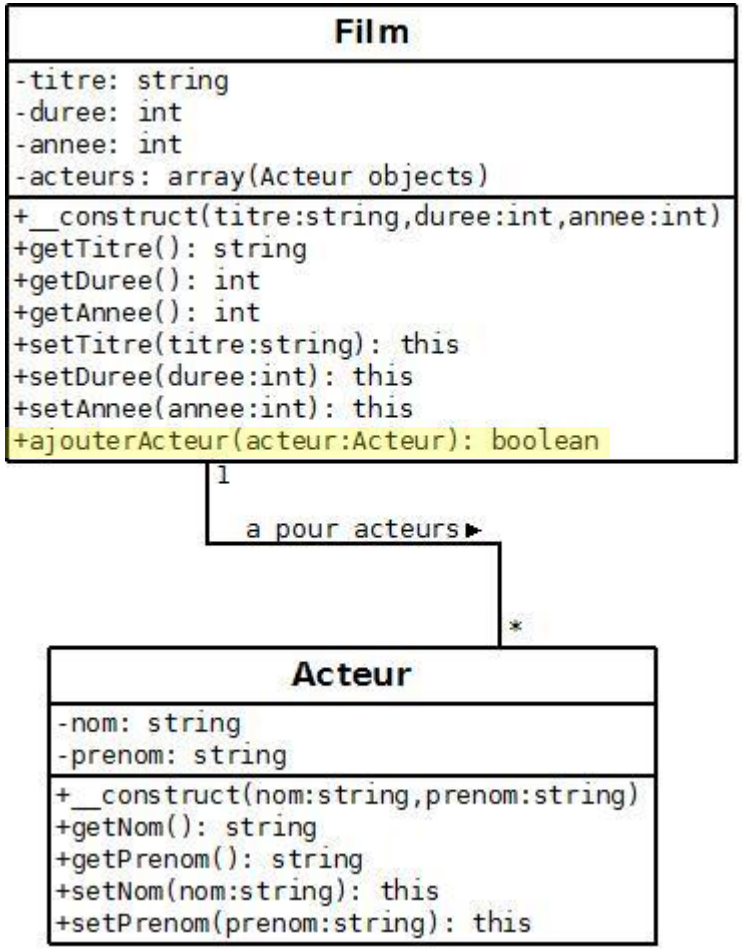
Comparez ces objets pour vérifier leurs caractéristiques:

- même instance ou pas,
- mêmes attributs et valeurs, et instances de la même classe.

Interactions entre objets : les associations

On dit que deux classes sont associées lorsqu'une instance des deux classes est amenée à interagir avec une instance de l'autre classe.

Modélisation UML, en prenant l'exemple de l'association entre une classe Film et une classe Acteur (extrait de la modélisation pour la gestion d'un club de location de DVD) :



L'association est caractérisée par le fait qu'une méthode, ajouterActeur, de la classe Film, entre en relation avec une instance de la classe Acteur.

Le chiffre 1 et l'astérisque sont les cardinalités et indiquent le nombre d'instances qui participent à l'interaction.

L'opérateur de résolution de portée (::)

Opérateur ::

Le "double deux points" est utilisé pour appeler des éléments appartenant à une classe et non à un objet.

C'est à dire que l'on peut accéder à ces éléments sans instancier la classe dans un objet.

L'opérateur de résolution de portée (::)

Cet opérateur permet d'accéder à 3 types d'éléments de la classe :

- **Les constantes de classes**

  permettent d'éviter les codes muets (on ne sait pas à quoi il correspond)

```php
const TYPE_VIDEO    = 0;
const TYPE_PHOTO    = 1;
const TYPE_AUDIO    = 2;
```

  et d'utiliser des variables en étant sûr qu'elles ne sont pas modifiées

```php
const TABLE_NAME = 'media';
```

- **Les variables statiques**

```php
private static $trace = false;
```

- **Les méthodes statiques**

```php
public static function lireLogTrace() {
    // lecture du fichier log de trace si $trace = true
}
```

L'opérateur de résolution de portée (::)

→ **Dans une classe, on utilise le mot-clé self**

```
self::TYPE_VIDEO
```

```
public static function lireLogTrace() {
    if (self::$trace) {
        // lecture du fichier log de trace si $trace = true
    }
}
```

→ **En dehors de la classe, on utilise le nom de la classe**

```
$media = new Media();
$media->setMediaType(Media::TYPE_VIDEO);
```

```
// $trace déclarée par exemple par 'public static $trace = false;'
MaClasse::$trace = true;
```

```
MaClasse::lireLogTrace();
```

Exercice 8

Complétez l'exercice 7 :

Ajoutez deux constantes pour signifier le sexe de la personne, valeur 'F' pour féminin et valeur 'M' pour masculin.
Ajoutez la propriété 'sexe' avec son getter et son setter.

Ajoutez une constante pour définir l'âge de la majorité (18 ans).
Ajoutez une méthode 'estMajeure()' pour connaître si une personne est majeure ou pas.

Ajoutez une classe Trace avec une méthode statique qui enregistre des messages dans un fichier.
Dans la classe Personne, ajoutez une propriété statique et le code nécessaire pour tracer les méthodes 'getPersonne()' et 'estMajeure()', en utilisant la classe Trace.

Créez des objets personnes et testez les méthodes.