Programación orientada a objetos en PHP

POO

- Modelo de programación que basa su esquema de pensamiento, análisis y diseño en interacciones entre objetos.
- Cada objeto tiene un comportamiento definido y que puede ser relacionado con otros objetos.
  - Colaborativos y cooperativos
  - Reusables
  - Distribuíbles
  - Localizables

Clase

- Una **clase** es la definición formal de un objeto (modelo) en los términos de estructura y comportamiento común.
- Así podemos usar la definición de una clase para crear objetos de ese tipo de clase, esto es, crear objetos que contengan todos los componentes especificados en la clase (instancias).

Objeto

- Un **objeto** es un ente que tiene estado, comportamiento e identidad englobados en una sola unidad.
- La estructura de un objeto deriva del concepto de tipo de dato abstracto.

Abstracción

- Define las caracteristicas esenciales de un objeto.

Encapsulamiento

- Se llama **encapsulamiento** a la conjugación de propiedades y comportamiento de un objeto.
- Esto logra también que se oculte la implementación y variables de ese objeto.

Componentes de una Clase

- Una definición formal de una clase se compone de:
  - Atributos. Estos son variables que almacenan datos referentes al objeto.
  - Métodos. Estos son las operaciones que se pueden realizar sobre objetos de esa clase. De manera simple, estan son las **funciones**.
- Tanto a los campos como los métodos se les considera **miembros**.

Clase en PHP

- Una clase en PHP se define mediante la palabra reservada **class** y enseguida, el identificador de la clase. Las propiedades y el comportamiento se definen dentro del cuerpo de la clase.

```
class NombreClase {
    //Atributos
    ...
    //Métodos
}
```

... Clase en PHP

```
class MiClase {
    $miVariable1;
    $miVariable2;
    ...
    modificadores miMetodo1(arg1, ...) {
    }
}
```

Instancias

- Un objeto es también conocido como una instancia de la clase a la que pertenece. Entonces al crearse la instancia, el objeto contendrá los campos definidos en la clase.
- Los miembros pueden clasificarse como:
  - Miembros de instancia, y
  - Miembros de clase

Miembros de instancia y clase

- Miembros de instancia
  - Cada objeto tendrá su propia copia local de cada variable definida en clase
  - Estas variables existen cuando se genera la instancia
- Miembros de clase
  - Son variables que existen en la clase y solo existe una sola copia para todas las instancias.
  - El valor es compartido y el mismo para todas las instancias.
  - Estas variables existen AÚN que no exista ni una instancia de esa clase.

Miembros de instancia y clase

- Un miembro o metodo de una clase declarado como static va a poder ser accesado sin necesidad de hacer una instancia del objecto en una variable.

... Miembros de instancia y clase

```
class Mensajes {
    public static $msok = 'Mensaje
ok!';

    public static function getMsOk(){
        echo self::$msok;
    }
}
Mensajes::getMsOk();
```

... Miembros de instancia y clase

```
class Person {
    public $name = 'Juanito';
    public $age;

    public static function getName(){
        echo self::$name;
    }

    public function setAge($a){
        $age = $a;
    }
}

Person::getName();
Person::setAge(21);
```

- Esto marca error, ¿Por qué?

```php
class Person {
    public $name;
    public $age;

    public function getName(){
        echo $this->name;
    }

    public function setAge($a){
        $this->$age = $a;
    }

    public function setName($n){
        $this->name = $n;
    }
}

$person1 = new Person;
$person1 -> setName('Juanito');
$person1 -> getName();
```

Acceso a campos y métodos

- Campos y métodos de instancia
  - `$a = new A();`
  - `...`
  - `$a -> i = 5;`
  - `$a -> imprime();`
- Campos y métodos de clase
  - `A::dato = 6;`
  - `A::ejecuta();`

Constructores

- Un constructor es un método especial que no devuelve ningún tipo de dato, que posee el mismo nombre de la clase y que tiene la finalidad de:
  - Crear espacio en memoria para el objeto
  - Inicializar las variables de instancia

... Constructores

```
class Clock {
   public $hour;
   function __construct() {
      $this -> hour = 12;
   }
   function setHour($hour) {
      $this -> hour = $hour;
   }
}
```

... Constructores

- Cuando un objeto es declarado para su uso posterior, es imperativo **construir** el objeto mediante una llamada al constructor.

```
class A {
   function __construct() {
   ...
   }
}
...
$a = new A();
```

```
class B{
    public $b;
    public function B() {
        $this->b = 12;
    }
}

$b = new B();
$c = $b;
$b->b = 15;

echo $c->b; //¿?
```

La variable this

- Esta variable siempre se refiere a la instancia actual.
- Por medio de esta referencia, se pueden acceder a los campos y métodos del objeto en turno.
- Es obligatorio ponerla.

Polimorfismo

- Es la capacidad que da a diferentes objetos, la posibilidad de contar con métodos, propiedades y atributos de igual nombre, sin que los de un objeto interfieran con el de otro.

Sobrecarga de métodos

- Hay ocasiones que resulta útil tener un mismo identificador de método para diferentes método con diferente funcionalidad.
- A esto le llamamos polimorfismo.
- PHP no cuenta con el polimorfismo como tal, pero existen los llamados “métodos mágicos” que pueden facilitar esta tarea.

Herencia

- Es la relación existente entre dos o más clases, donde una es la principal (madre) y otras son secundarias y dependen (heredan) de ellas (clases “hijas”), donde a la vez, los objetos heredan las características de los objetos de los cuales heredan.