**TEMA 2: Programación Orientada a Objetos**

**1. Programación Orientada a Objetos en PHP**

La inclusión de la orientación a objetos en PHP es uno de los cambios más significativos que introdujo la versión 5. Es habitual utilizar una metodología secuencial, pero esto provoca muchos problemas para el mantenimiento y actualización del código.

El objetivo será utilizar la POO para desarrollar librerías lo más abstractas y flexibles posibles, dejando el desarrollo secuencial para las vistas.

**2. Objetos**

Los objetos son la representación mínima de cualquier resolución a un problema a nivel de software.

* Propiedades/atributos: Características del objeto.
* Métodos/funciones: Acciones que puede realizar.

**3. Clase**

Una clase se crea mediante la palabra reservada class, seguida de su nombre y unas llaves donde se incorporaran sus propiedades y métodos. El nombre de una clase debe empezar por una letra o un guion, pero podrá contener también números.

**4. Problemas al no usar objetos**

* Como PHP no es un lenguaje tipificado, las variables podrían cambiarse a tipos no deseados. Esto no sucedería con la propiedad de un objeto.
* En caso de usar una función, debemos recordar los parámetros y el orden de los mismos. En el método de un objeto podríamos realizar la misma acción sin utilizar parámetros.
* Cuando queramos instanciar otro elemento similar, tendremos que volver a instanciar también cada atributo con una nueva variable.

Además de solucionar estos problemas, con la POO ganamos en abstracción, mantenimiento, seguridad y legibilidad del código.

**5. Include vs Require**

* include: Incluye y evalúa el archivo especificado. Si no lo encuentra, emitirá un warning. Por su parte, include\_once incluirá el fichero una sola vez y devolverá true.
* require: Incluye y evalúa el archivo especificado. Si no lo encuentra, producirá un error fatal. Por su parte, require\_once verificará si el fichero ha sido incluido, y si es así, no lo vuelve a incluir.

**6. Propiedades y métodos**

Las propiedades de una clase es una serie de variables que definen sus características.

Un método será una función que realice una determinada acción. En PHP no es necesario especificar el tipo de los parámetros ni es necesario que una función devuelva con return.

En PHP no es posible sobrecargar un método, así que para incluir variantes con menos parámetros, debemos indicar previamente que los parámetros opcionales tendrán un valor null por defecto.

$this es una pseudovariable que hace referencia a una propiedad o método de la misma clase.

**7. Ámbito de las variables**

En PHP, el ámbito de una variable se considera simple. De esta manera, si cambiamos el valor de una variable dentro de una función, este cambio solo estará presente dentro de la función, manteniendo su valor original fuera de la misma.

**8. Visibilidad de las clases**

La visibilidad permite definir donde y quien puede acceder a un atributo, método o constante de una clase.

* public: Se puede acceder desde cualquier sitio.
* protected: Solo se puede acceder desde la clase o clase heredadas.
* private: Solo se puede acceder desde la clase.

Los getter son métodos que devuelven el valor de una propiedad.

Los setter son métodos que asignan el valor de una propiedad.

Lo habitual es que los atributos de una clase sean privados y que sus métodos sean públicos

**9. Constructor**

Se trata de una función especial para crear la instancia de una clase. Se llama \_\_construct y recibe como parámetros las propiedades del objeto que queremos definir.

**10. Destructor**

Se trata de una función para destruir un objeto en nuestra aplicación. Se llama \_\_destruct.

**11. Herencia**

Con la herencia, una clase hija hereda los métodos y atributos public y protected de la clase padre.

Para que una clase herede de otra, se utiliza la palabra reservada extends.

Con la palabra reservada final, se impide que se herede de esa clase o se pueda sobreescribir un método.

**12. Constantes**

También se pueden crear constantes con la palabra reservada const. Para poder acceder a una constante, utilizaremos la palabra reservada self.

**13. Clases abstractas e interfaces**

Una clase abstracta es aquella en la que no se puede instanciar un objeto, y debe contener al menos un método abstracto. Se declaran con la palabra reservada abstract.

Los Interfaces son clases especiales con métodos que pueden ser implementados por otras clases. Existe polimorfismo cuando una o más clases utilizan la misma interface. Se declaran con la palabra reservada interface.

Diferencias:

* Los Interfaces no pueden tener atributos
* Todos los métodos de una interface deben ser public y abstractos
* Una clase puede heredar de otra e implementar una interface a la vez.

**14. Traits**

PHP solo soporta herencia simple. Los Traits se usan para declarar métodos y métodos abstractos que puedan ser utilizados por muchas clases. Se declaran con la palabra reservada trait.

**15. Métodos y atributos estáticos**

Los métodos y atributos estáticos son aquellos que pueden ser llamados directamente sin crear una instancia del objeto. Se declaran con la palabra static.