

# Métodos mágicos en PHP

Funciones especiales que empiezan por “\_”

# Métodos mágicos

Funciones especiales que comienzan con ``__`` (doble guión bajo).

Permiten **manejar comportamientos específicos de objetos**. Se activan **automáticamente** en ciertas circunstancias, facilitando el manejo dinámico de clases.

Ya conocemos una:

**`__construct()`**

Se ejecuta al instanciar la clase; tenga o no su propio `__construct` especificado.



# Principales métodos mágicos

## 1. `__construct()` y `__destruct()`

Constructor y destructor, se ejecutan al crear o destruir un objeto.

## 2. `__get($prop)` y `__set($prop, $valor)`

Controlan el acceso y la asignación de propiedades inaccesibles o inexistentes.

## 3. `__call($metodo, $args)` y `__callStatic($metodo, $args)`

Manejan llamadas a métodos inexistentes (en contexto normal o estático).



# Principales métodos mágicos

## 4. **\_\_toString()**

Define cómo un objeto se convierte en una cadena de texto.

## 5. **\_\_invoke()**

Permite que un objeto sea llamado como si fuera una función.

## 6. **\_\_clone()**

Duplica un objeto. Permite personalizar el comportamiento al clonar.

## 7. **\_\_sleep()**, **\_\_wakeup()**, **\_\_serialize()** y **\_\_unserialize()**

Se usan en la serialización y deserialización de objetos.



# Listado completo

Listado completo de métodos mágicos: `__construct()`, `__destruct()`, `__call()`, `__callStatic()`, `__get()`, `__set()`, `__isset()`, `__unset()`, `__sleep()`, `__wakeup()`, `__serialize()`, `__unserialize()`, `__toString()`, `__invoke()`, `__set_state()`, `__clone()`, and `__debugInfo()`.

**Los más usados: `__construct()`, `__toString()`, `__clone()`**

Estos métodos siempre se deben declarar como **public** (sino dan Warning).



# Ejemplo de uso: \_\_toString()

```
class Persona {  
    private $nombre;  
    private $edad;  
    public function __construct($nombre, $edad) {  
        $this->nombre = $nombre;  
        $this->edad = $edad;  
    }  
  
    // Método mágico __toString  
    public function __toString() {  
        return "Nombre: ".$this->nombre.", Edad: ".$this->edad."<br>";  
    }  
}
```

```
$persona = new Persona("Juan", 30);  
echo $persona;
```

// Salida: Nombre: Juan, Edad: 30

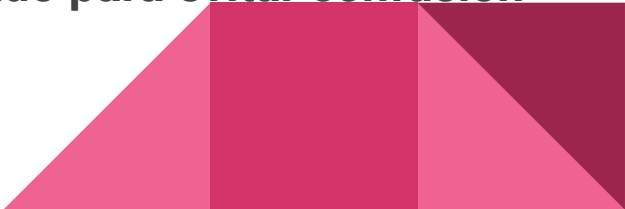
# Ejemplo de uso: \_\_clone()

```
class Libro {  
    public $titulo;  
    public $autor;  
    public function __construct($titulo, $autor) {  
        $this->titulo = $titulo;  
        $this->autor = $autor;  
    }  
  
    // Método mágico __clone  
    public function __clone() {  
        $this->titulo = "Copia de: " . $this->titulo;  
    }  
}  
  
$original = new Libro("Harry Potter", "Ana");  
$copia = clone $original;  
echo "Original: ".$original->titulo."<br>";    // Salida: Harry Potter  
echo "Clonado: ".$copia->titulo."<br>";        // Salida: Copia de: Harry Potter
```

# Relación con P00

- **Encapsulamiento:** Los métodos mágicos como `__get` y `__set` permiten gestionar el acceso controlado a propiedades privadas o protegidas.
- **Polimorfismo:** Métodos como `__call` permiten manejar diferentes comportamientos dinámicos según el contexto.
- **Flexibilidad:** Permiten programar dinámicamente sin modificar directamente las propiedades o métodos de la clase, respetando los principios de diseño orientado a objetos.

Son herramientas potentes, pero **deben usarse con cuidado para evitar confusión** en el mantenimiento del código.





# Método `__equals` en PHP

En PHP **no existe** el método mágico `__equals()` para comprar objetos.

Aun así, se puede hacer de forma manual, empleando, por ejemplo, los comparadores:

- `'=='` : compara las propiedades de los objetos y sus valores
- `'==='` : compara si dos variables **apuntan a la misma instancia** (mismo objeto)

