Object-Oriented Programming (OOP) in PHP

Object-Oriented Programming (OOP) is a programming style that organizes code into objects, which are self-contained units combining data and functions. In PHP, OOP was introduced in version 5 and has become essential for building scalable and maintainable web applications.

# Key OOP Features in PHP

## 1. Classes and Objects

Class: A blueprint for creating objects. It defines properties (variables) and methods (functions).  
Object: An instance of a class.

Example:   
  
class Car {  
 public $color;  
 public $model;  
  
 public function \_\_construct($color, $model) {  
 $this->color = $color;  
 $this->model = $model;  
 }  
  
 public function displayInfo() {  
 echo "A $this->color $this->model.";  
 }  
}  
  
$myCar = new Car("red", "Toyota");  
$myCar->displayInfo(); // Outputs: A red Toyota.

## 2. Encapsulation

Encapsulation: Hiding the internal state of an object and requiring all interaction to be performed through an object's methods. This protects the object's integrity by preventing outside interference and misuse.

## 3. Inheritance

Inheritance: Allows a class (child) to inherit properties and methods from another class (parent), promoting code reuse.

Example:   
  
class ElectricCar extends Car {  
 public $batteryLife;  
  
 public function \_\_construct($color, $model, $batteryLife) {  
 parent::\_\_construct($color, $model);  
 $this->batteryLife = $batteryLife;  
 }  
  
 public function displayBattery() {  
 echo "Battery life: $this->batteryLife hours.";  
 }  
}  
  
$myElectricCar = new ElectricCar("blue", "Tesla", 24);  
$myElectricCar->displayInfo(); // Outputs: A blue Tesla.  
$myElectricCar->displayBattery(); // Outputs: Battery life: 24 hours.

## 4. Polymorphism

Polymorphism: Enables objects of different classes to be treated as objects of a common parent class. It allows methods to do different things based on the object it is acting upon.

Example:   
  
class Animal {  
 public function speak() {  
 echo "Animal makes a sound.";  
 }  
}  
  
class Dog extends Animal {  
 public function speak() {  
 echo "Dog barks.";  
 }  
}  
  
class Cat extends Animal {  
 public function speak() {  
 echo "Cat meows.";  
 }  
}  
  
$animal = new Animal();  
$dog = new Dog();  
$cat = new Cat();  
  
$animal->speak(); // Outputs: Animal makes a sound.  
$dog->speak(); // Outputs: Dog barks.  
$cat->speak(); // Outputs: Cat meows.

# Benefits of OOP in PHP

PHP's adoption of OOP has aligned it with modern programming practices, making it a powerful tool for developers building dynamic and complex web applications.

1. Modularity: Code is organized into classes and objects, making it easier to manage and understand.

2. Reusability: Classes can be reused across different parts of an application or in different projects.

3. Maintainability: Changes in one part of the code can be made independently of others, reducing the risk of bugs.

4. Scalability: OOP principles support the development of large-scale applications by promoting clean and organized code structures.