Konsep OOP dalam PHP

Konsep OOP (Object-Oriented Programming) dalam PHP mengacu pada cara menggunakan paradigma pemrograman berbasis objek. Berikut adalah konsep-konsep utama OOP dalam PHP:

1. Class dan Object

Class adalah blueprint atau cetakan untuk menciptakan objek. Itu mendefinisikan struktur dan perilaku objek. Sedangkan **objek** adalah instance dari class yang memiliki properti (variabel) dan metode (fungsi).

Contoh penggunaan class dan pembuatan objek:

```
// Mendefinisikan class
class Car {
    // Properti atau atribut
    public $brand;
    public $model;

    // Metode atau perilaku
    public function displayInfo() {
        return "This is a {$this->brand} {$this->model}.";
    }
}

// Membuat objek dari class Car
$car1 = new Car();
$car1->brand = "Toyota";
$car1->model = "Corolla";

echo $car1->displayInfo(); // Output: This is a Toyota Corolla.
```

2. Encapsulation (Pembungkusan)

Encapsulation menggabungkan data dan perilaku yang beroperasi pada data ke dalam satu unit, yaitu objek. Dalam PHP, encapsulation diterapkan menggunakan visibilitas properti dan metode (`public` , `private` , `protected`).

Contoh penggunaan encapsulation:

```
class Employee {
    private $name;
    private $salary;

    public function __construct($name, $salary) {
        $this->name = $name;
        $this->salary = $salary;
    }

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

    public function getSalary() {
        return $this->salary;
    }
}

$employee1 = new Employee("John Doe", 5000);
echo $employee1->getName(); // Output: John Doe
echo $employee1->getSalary(); // Output: 5000
```

3. Inheritance (Pewarisan)

Inheritance memungkinkan class baru (child class) untuk mengambil atau mewarisi properti dan metode dari class yang sudah ada (parent class). Ini memungkinkan untuk mengorganisir dan menyesuaikan kode dengan lebih efisien.

Contoh penggunaan inheritance:

```
// Parent class
class Animal {
    public function makeSound() {
        return "Some sound";
    }
}
```

```
// Child class inherits from Animal
class Dog extends Animal {
    public function makeSound() {
        return "Bark";
    }
}

$dog = new Dog();
echo $dog->makeSound(); // Output: Bark
```

4. Polymorphism (Polimorfisme)

Polymorphism memungkinkan objek dari class yang berbeda untuk merespons pesan atau metode dengan cara yang unik untuk masing-masing class tersebut. Ini dapat dicapai dengan overriding (mengganti metode dari parent class) atau dengan menggunakan interface dan abstract class.

Contoh penggunaan polymorphism dengan overriding:

```
// Parent class
class Shape {
    public function calculateArea() {
        return "Calculate area of shape";
    }
}

// Child class overrides parent method
class Circle extends Shape {
    public function calculateArea() {
        return "Calculate area of circle";
    }
}

$circle = new Circle();
echo $circle->calculateArea(); // Output: Calculate area of circle
```

5. Abstraction (Abstraksi)

Abstraction mengacu pada ide menyembunyikan detail implementasi dan hanya menampilkan fitur penting dari objek. Dalam PHP, ini sering dicapai dengan menggunakan abstract class dan

interface.

Contoh penggunaan abstraction dengan abstract class:

```
// Abstract class
abstract class Vehicle {
    abstract public function start();
    abstract public function stop();
}

// Child class extends abstract class
class Car extends Vehicle {
    public function start() {
        return "Car starting...";
    }

    public function stop() {
        return "Car stopping...";
    }
}

$car = new Car();
echo $car->start(); // Output: Car starting...
```

Konsep-konsep ini membentuk dasar dari OOP dalam PHP dan membantu dalam mengorganisir kode, meningkatkan keterbacaan, dan mempermudah dalam pengelolaan serta pemeliharaan aplikasi yang kompleks.

Revision #1
Created 13 December 2024 13:59:47 by Admin
Updated 13 December 2024 14:00:24 by Admin