Inheritance and Polymorphism in PHP (OOP)

Learning Objectives

1. By the end of this lesson, you should be able to:

2. Understand the concept and purpose of inheritance in OOP.

3. Understand the relationship between base and derived classes.

4. Demonstrate method overriding to customize inherited behavior.

5. Use the parent:: keyword to enhance inherited methods in derived classes.

6. Explain and implement polymorphism using a common base class.

What is Inheritance?

- Inheritance allows a class to reuse the properties and methods of another class.
- The base class is also called the parent class.
- The derived class is also called the child class.
- Promotes code reuse and modularity.

Roles of Base and Derived Classes

Role	Description
Base Class	The general class (e.g., Person)
Derived Class	The specialized class (e.g., Student)

SYNTAX EXAMPLE

index.php

```php
<?php
class Person {
    public $name;
    public function speak() {
        echo "Hi, I'm $this->name";
    }
}

class Student extends Person {
    public $course;
}

$student = new Student();
$student->name = "Jerico";
$student->speak();

?>
```

What is Method Overriding?

- Method overriding allows a child class to redefine a method from the parent class.
- Used to customize or change inherited behavior.

SYNTAX EXAMPLE

index.php M

index.php

```php
<?php
class Person {
    public function introduce() {
        echo "I am a person.";
    }
}

class Student extends Person {

}

$student = new Student();
$student->introduce();


?>

```

Using parent:: to Extend Behavior

- Use parent::methodName() to call the original method from the parent class.
- Useful when the child needs to extend, not fully replace, the behavior.

SYNTAX EXAMPLE

index.php M

index.php

```php
<?php

class Person {
    public function introduce() {
        echo "I am a person.";
    }
}

class Teacher extends Person {
    public function introduce() {
        parent::introduce();
        echo " And I am a teacher.";
    }
}

$teacher = new Teacher();
$teacher->introduce();

?>
```

What is Polymorphism?

- Polymorphism means one interface, many behaviors.
- Different classes can implement the same method name differently.
- Achieved through method overriding.

Concept of Polymorphism

- All child classes inherit from the same base.
- Each class overrides a method (e.g., introduce()).
- You can treat them as the same type (e.g., Person) in your code.

SYNTAX EXAMPLE

index.php M

index.php

```php
<?php
class Animal {
    public function makeSound() {
        echo "This animal makes a sound.\n";
    }
}

class Dog extends Animal {
    public function makeSound() {
        echo "The dog barks.\n";
    }
}

class Cat extends Animal {
    public function makeSound() {
        echo "The cat meows.\n";
    }
}

$animals = [new Dog(), new Cat(), new Cow()];

foreach ($animals as $animal) {
    $animal->makeSound();
}


?>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Laptop-K1\Documents\Object-Oriented-Programming-Lesson> php index.php
The dog barks.
The cat meows.
PS C:\Users\Laptop-K1\Documents\Object-Oriented-Programming-Lesson>
```

Summary

Concept	Description
Inheritance	Share and reuse code using base and derived classes
Method Overriding	Redefine inherited methods to customize behavior
parent::	Extend inherited method logic using the parent keyword
Polymorphism	Use the same method across different objects with different behavior

Thank you