LẬP TRÌNH BACK-END WEB 1
| Back-end Web Development 1|

Bùi Thị Phương Thảo - Nguyễn Huy Hoàng
[10-2020]

Chương 2
Lập Trình Hướng Đối Tượng Trong PHP

Nội dung

1. Một số khái niệm cơ bản

2. Thừa kế lớp

Lớp - Class

Để khai báo một lớp, ta dùng từ khóa class, trong đó tên class phải:

- Phải được bắt đầu bằng ký tự hoặc dấu _
- Không được bắt đầu bằng số
- Có thể chứa ký tự A-z, ký số 0-9 hoặc dấu _
- Được Viết Hoa các ký tự đầu tiên của mỗi từ

```php
<?php
class Fruit {
  // Properties
  public $name;
  public $color;
  // Methods
  public function setName($name) {
    $this->name = $name;
  }
  public function getName() {
    return $this->name;
  }
}
?>
```

PHP Access Modifiers

- Public - properties hay methods có thể được truy cập ở bất kỳ nơi đâu, bao gồm cả bên trong và bên ngoài class.
- Private - properties hay methods chỉ có thể được truy cập bên trong nội bộ class.
- Protected - properties hay methods chỉ có thể được truy cập bên trong class đó hoặc trong các class con kế thừa từ class đó.

$this

$this là từ khóa dùng tham chiếu đến chính đối tượng được gọi.

Đối tượng - Objects

- Có thể tạo nhiều đối tượng từ class
- Mỗi đối tượng được tạo sẽ có tất cả thuộc tính và phương thức của class

```php
$apple = new Fruit();
$banana = new Fruit();
$apple->setName('Apple');
$banana->setName('Banana');

echo $apple->getName();
echo "<br>";
echo $banana->getName();
```

$apple and $banana are instances of the class Fruit

Hàm __construct()

Hàm được gọi một cách tự động khi đối tượng được tạo

```php
<?php
class Fruit {
  public $name;
  public $color;

  function __construct($name) {
     $this->name = $name;
  }
  function getName() {
    return $this->name;
  }
}
```

```php
$apple = new Fruit("Apple");
echo $apple->getName();

?>
```

Hàm __destruct()

Hàm được gọi một cách tự động khi đối tượng bị huỷ

```php
<?php
class Fruit {
  public $name;
  public $color;

  function __construct($name) {
    $this->name = $name;
  }
  function __destruct() {
    echo "The fruit is {$this->name}.";
  }
}
```

```php
$apple = new Fruit("Apple");

?>
```

Thừa kế lớp

Một lớp có thể kế thừa các public properties và methods của một lớp khác bằng cách sử dụng từ khóa extends

```php
<?php
class Fruit {
  public $name;
  public $color;
  public function __construct($name, $color) {
      $this->name = $name;
      $this->color = $color;
  }
  public function intro() {
    echo "The fruit is {$this->name} and the color is {$this->color}.";
  }
}
```

```php
// Strawberry is inherited from Fruit
class Strawberry extends Fruit {
  public function message() {
    echo "Am I a fruit or a berry? ";
  }
}
$strawberry = new Strawberry("Strawberry", "red");
$strawberry->message();
$strawberry->intro();

?>
```

Overwrite

Để thay đổi một public property hay method kế thừa từ lớp cha, ta chỉ cần định nghĩa lại property hay method đó trong lớp con

```php
<?php
class Fruit {
  public $name;
  public $color;
  public function __construct($name, $color) {
    $this->name = $name;
    $this->color = $color;
  }
  public function intro() {
    echo "The fruit is {$this->name} and the color is {$this->color}.";
  }
}
```

```php
// Strawberry is inherited from Fruit
class Strawberry extends Fruit {
// Override the intro method in Fruit
  public function intro() {
    echo "Am I a fruit or a berry? ";
  }
}
$strawberry = new Strawberry("Strawberry", "red");
$strawberry->intro();

?>
```

Overwrite

Khi có nhu cầu giữ lại hàm gốc từ class cha, đồng thời thêm vào đó một vài câu lệnh khác, ta dùng parent:: trước tên hàm cha cần giữ lại.

```php
<?php
class Fruit {
  public $name;
  public $color;
  public function __construct($name, $color) {
      $this->name = $name;
      $this->color = $color;
  }
  public function intro() {
    echo "The fruit is {$this->name} and the
color is {$this->color}.";
  }
}
```

```php
// Strawberry is inherited from Fruit
class Strawberry extends Fruit {
// Override the intro method in Fruit
  public function intro() {
// keep the parent's intro method
      parent::intro();
      echo "Am I a fruit or a berry? ";
  }
}
$strawberry = new Strawberry("Strawberry",
"red");
$strawberry->intro();
?>
```

Static

- Khi properties hay methods được khai báo kèm từ khóa static, chúng có thể được truy cập mà không cần khởi tạo instance cho class đó
- Cách gọi static properties hay methods dùng dấu :: (scope resolution operator)

```php
<?php
class Greeting {
  public static function welcome() {
    echo "Hello World!";
  }
}
// Call static method
Greeting::welcome();
?>
```

Static

Một trong những ưu điểm chính của static properties là dữ liệu của chúng được lưu giữ trong suốt quá trình hoạt động

```php
<?php
class MyClass {
  public static $count = 1;
  public static function plusOne() {
     return ++self::$count;
  }
}
MyClass::plusOne();
MyClass::plusOne();
MyClass::plusOne();
?>
```

Thanks for your attention!

FACULTY OF
INFORMATION TECHNOLOGY
THU DUC COLLEGE OF TECHNOLOGY

Phone: (+848) 22 158 642
Email: fit@tdc.edu.vn
Website: fit.tdc.edu.vn
Facebook: facebook.com/tdc.fit
Youtube: youtube.com/fit-tdc