Lab 4. Lập trình hướng đối tượng trong PHP (4 tiết)

I. Yêu cầu

- Sinh viên tìm hiểu các kiến thức liên quan đến lập trình hướng đối tượng trong PHP, xây dựng các bài toán xử lý OOP trên PHP.

- Mỗi người làm trên một dự án khác nhau.

- Khi có yêu cầu, sinh viên nộp bài qua LMS.

II. Lý thuyết

Lập trình hướng đối tượng (gọi tắt là OOP - object-oriented programming) là một kĩ thuật lập trình hỗ trợ công nghệ đối tượng. Nếu như trước kia là các kiểu lập trình hướng thủ tục, hướng modun,.. thì giờ đây thế giới đang ưu về sử dụng hướng đối tượng.Nếu như trước đây chúng ta lập trình với hướng thủ tục thì sẽ chia thành các hàm để xử lý, thì giờ đây khi sử dụng hướng đối tượng thì chúng ta sẽ chia ra thành các đối tượng để xử lý.

Để tìm hiểu thêm về lập trình hướng đối tượng trong PHP, sinh viên xem thêm tại đây:

https://toidicode.com/series/php-huong-doi-tuong

III. Luyện tập

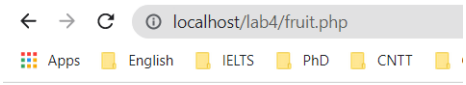
Practical 1: Classes and objects

1. Create a php web page illustrate class, page name is `fruit`.php and output in this page:

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="utf-8">
	<title></title>
</head>
<body>
 <?php
	class Fruit {
	  public $name;
	  public $color;

	  function __construct($name, $color) {
	    $this->name = $name;
	    $this->color = $color;
	  }
	  function get_name() {
	    return $this->name;
	  }
	  function get_color() {
	    return $this->color;
	  }
	}
```

```
        $apple = new Fruit("Apple", "red");
        echo $apple->get_name();
        echo "<br>";
        echo $apple->get_color();
?>
</body>
</html>
```

Output:



Apple
red

2. Create a php web page illustrate class, page name is student.php and call class in index.php page:

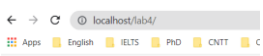
student.php:

```
<?php
class Students {
        public $first_name;
        public $last_name;
        public $address;
        public function __construct($first_name, $last_name, $address){
                $this->first_name = $first_name;
                $this->last_name = $last_name;
                $this->address = $address;
        }
        public function greeting(){
                return "Hello ".$this->first_name."\n";
        }
        public function getAddress(){
                return $this->address."\n";
        }
}
?>
```

index.php:

```
  <?php
    include 'student.php';
    $student = new Students("John", "Doe", "10/33 ABC, Da Lat");
    echo $student->greeting()."<br/>";
    echo $student->getAddress();
  ?>
```

Output:



Hello John
10/33 ABC, Da Lat

Practical 2: Inheritance

1. Inheritance in one page named fruit_inherit.php:

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="utf-8">
	<title></title>
</head>
<body>
<?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}.";
  }
}

// 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();
?>
</body>
</html>
```

Output:

3. Inheritance in separate pages:

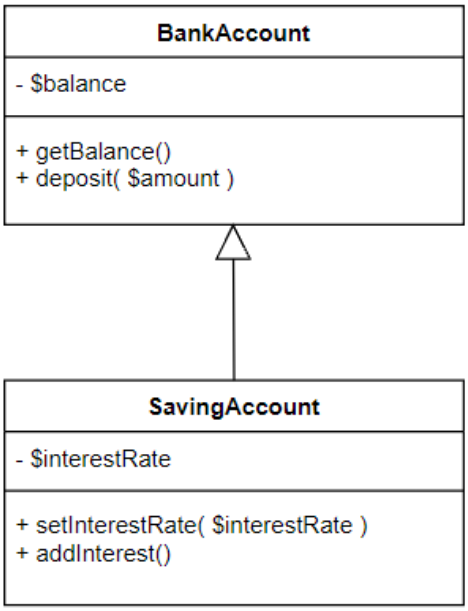
fruit.php
[Students write their own code for class Fruit at here]

strawberry.php
[Students write their own code for class Fruit at here]

index.php

```
<!DOCTYPE html>
<html>
<head>
       <meta charset="utf-8">
       <title></title>
</head>
<body>
  <?php
    include 'fruit.php';
    include 'strawberry.php';
    $strawberry = new Strawberry("Strawberry", "red");
    $strawberry->message();
    $strawberry->intro();
  ?>
</body>
</html>
```

4. The following UML diagram shows the inheritance relationship between the BankAccount and SavingAccount classes:

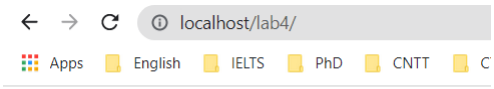


- Suggestions source codes:

```
<?php
class BankAccount
{
    private $balance;
    public function getBalance()
    {
        return $this->balance;
    }
    public function deposit($amount)
    {
        if ($amount > 0) {
            $this->balance += $amount;
        }
        return $this;
    }
}
class SavingAccount extends BankAccount
{
    private $interestRate;

    public function setInterestRate($interestRate)
    {
        $this->interestRate = $interestRate;
    }
    public function addInterest()
    {
        // calculate interest
        $interest = $this->interestRate * $this->getBalance();
        // deposit interest to the balance
        $this->deposit($interest);
    }
}
```

```
$account = new SavingAccount();
$account->deposit(100);
// set interest rate
$account->setInterestRate(0.05);
$account->addInterest();
echo $account->getBalance();
?>
```

Output in index.php:



105

Practical 3: Interface

1. Write a program that demonstrate interface in PHP:

interface.php:

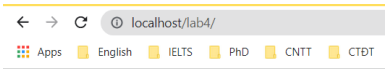
```
<?php
// Interface definition
interface Animal {
  public function makeSound();
}

// Class definitions
class Cat implements Animal {
  public function makeSound() {
    echo " Meow ";
  }
}

class Dog implements Animal {
  public function makeSound() {
    echo " Bark ";
  }
}

class Mouse implements Animal {
  public function makeSound() {
    echo " Squeak ";
  }
}
?>
```

index.php:

```
<!DOCTYPE html>
<html>
<head>
       <meta charset="utf-8">
       <title></title>
</head>
<body>
  <?php
```

```
    include 'interface.php';
    // Create a list of animals
    $cat = new Cat();
    $dog = new Dog();
    $mouse = new Mouse();
    $animals = array($cat, $dog, $mouse);

    // Tell the animals to make a sound
    foreach($animals as $animal) {
      $animal->makeSound();
    }
  ?>
</body>
</html>
```

Output:



Meow Bark Squeak

2. In the following example, we will show you how to use the interface to make the system more flexible and easier to extend.

***loger.php**:*

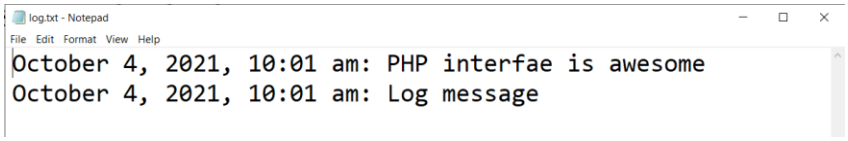
```
<?php

interface Logger
{
	public function log($message);
}

class FileLogger implements Logger
{
	private $handle;
	private $logFile;
	public function __construct($filename, $mode = 'a')
	{
		$this->logFile = $filename;
		// open log file for append
		$this->handle = fopen($filename, $mode)
				or die('Could not open the log file');
	}
	public function log($message)
	{
		$message = date('F j, Y, g:i a') . ': ' . $message . "\n";
		fwrite($this->handle, $message);
	}
	public function __destruct()
	{
		if ($this->handle) {
			fclose($this->handle);
		}
	}
}
```

```
class DatabaseLogger implements Logger
{
	public function log($message)
	{
		echo sprintf("Log %s to the database\n", $message);
	}
}
?>
```

***index.php**:*

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="utf-8">
	<title></title>
</head>
<body>
  <?php
    include 'loger.php';
    // examle 1
    $logger = new FileLogger('./log.txt', 'w');
    $logger->log('PHP interfae is awesome');
    // example 2
    $loggers = [
      new FileLogger('./log.txt'),
      new DatabaseLogger()
    ];
    foreach ($loggers as $logger) {
      $logger->log('Log message');
    }
  ?>
</body>
</html>
```

Output:



IV. Bài tập

Bài tập 1: Xây dựng lớp ***PhanSo*** gồm:

- Thuộc Tính: *tuso*, *mauso*;
- Phương thức:
  - Hàm Khởi Tạo Không Tham Số, Hàm Hủy;
  - Nhập , Xuất;
  - Cong(), Tru(), Nhan(), Chia();
  - Tính Tổng, Hiệu, Tích, Thương 2 phân số obj1 và obj2 rồi in kết quả ra màn hình

Bài tập 2: Xây dựng lớp vận động viên ***VanDongVien*** gồm:

- Thuộc tính: hoten, tuoi, monthidau, cannang, chieucao.
- Phương thức:
  - Thiết lập không tham số;
  - Thiết lập 5 tham số;
  - Nhập; Xuất
  - So sánh (một vận động viên là lớn hơn nếu chiều cao lớn hơn, trong trường hợp chiều cao bằng nhau thì xét cân nặng lớn hơn)

Xây dựng trang baitap2_lab4.php xử lý các nội dung sau:

- Khai báo p là đối tượng lớp VanDongVien (sử dụng hàm thiết lập 5 tham số), hiển thị thông tin của p ra màn hình;
- Nhập vào một mảng gồm n vận động viên;
- Hiển thị danh sách đã nhập ra màn hình;
- Sắp xếp mảng đã nhập theo thứ tự tăng dần, hiển thị danh sách đã sắp ra màn hình.

Bài tập 3:

Xây dựng lớp ***VeMayBay*** gồm:

- Thuộc tính: tenchuyen, ngaybay, giave
- Phương thức: Phương thức tạo lập; Nhập; Xuất; getGiaVe(): trả về giá vé.

Xây dựng lớp ***Nguoi*** gồm:

- Thuộc tính: hoten, gioitinh, tuoi;
- Phương thức: Phương thức tạo lập; Nhập; Xuất;

Xây dựng lớp Hanhkhach (mỗi hành khách được mua nhiều vé) kế thừa lớp Nguoi bổ sung thêm:

- Thuộc tính: VeMayBay ve; soluong;
- Phương thức: Phương thức tạo lập; Nhập; Xuất; tongTien(): trả về Tổng số tiền phải trả của hành khách.

Chương trình chính:

- Nhập vào 1 danh sách n hành khách (n nhập từ bàn phím).
- Hiển thị danh sách hành khách và số tiền phải trả tương ứng của mỗi khách hàng.
- Sắp xếp danh sách hành khách theo chiều giảm dần của Tổng tiền.

--Hết--