PHẦN 5:

PHP VÀ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

1. Giới thiệu
2. Các vấn đề cơ bản hướng đối tượng trong PHP
3. Lớp abstract và lớp interfaces
4. Hàm include và require

SV tự học một số đối tượng có sẵn trong PHP:

- GET, POST, REQUEST
- COOKIE
- SESSION
- SERVER
- FILES

1.Khai báo lớp, đối tượng

- **Cú pháp khai báo lớp**:

```
class <Tên_lớp>{
    // Your code is here
    …
}
```

- **Ví dụ**:

```
class foo {
        const BAR = "Hello World";
}
echo foo::BAR;
```

- **Cú pháp khai báo lớp kế thừa**:

```
class a {
    function test(){ echo "a::test called";}
    function func(){echo "a::func called";}
}
class b extends a {
    function test(){echo "b::test called";}
}
class c extends b {
    function test(){parent::test();}
}
class d extends c {
    function test(){b::test();}
}
```

- **Cú pháp xác định lớp đối tượng**:

```
if ($obj instanceof MyClass) {
            echo "\$obj is an instance of MyClass";
}
```

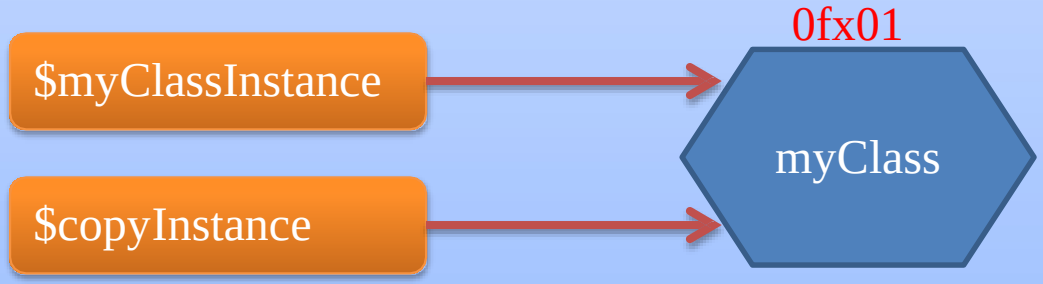
- **Cú pháp tạo đối tượng**:

```
$myClassInstance = new myClass();
```

Lưu ý: các đối tượng trong PHP được sử dụng theo dạng tham chiếu

- **Ví dụ**:

```
$myClassInstance = new myClass();
$copyInstance = $myClassInstance();
// Cả 2 biến $myInstance và $copyInstance cùng trỏ tới một đối tượng thuộc myClass.
```



- **Phương thức và thuộc tính**:

```
class myClass {
    function myFunction() {
        echo "You called myClass::myFunction";
    }
}
// Access methods of class myClass
$obj = new myClass();
$obj -> myFunction();
```

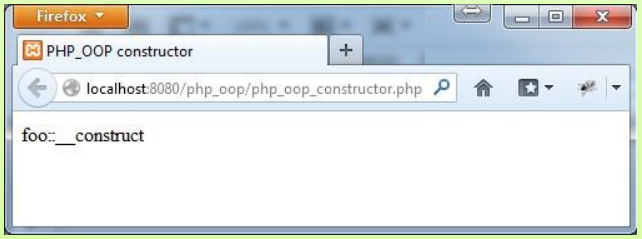
- **Con trỏ $this**:

```
class myClass {
    function myFunction($data) {
        echo "The value is $data";
    }
    function callMyFunction($data) {
        // Call myFunction()
        $this->myFunction($data);
    }
}
$obj = new myClass();
$obj->callMyFunction(123);
```

Constructors

- **Cú pháp hàm khởi tạo**:

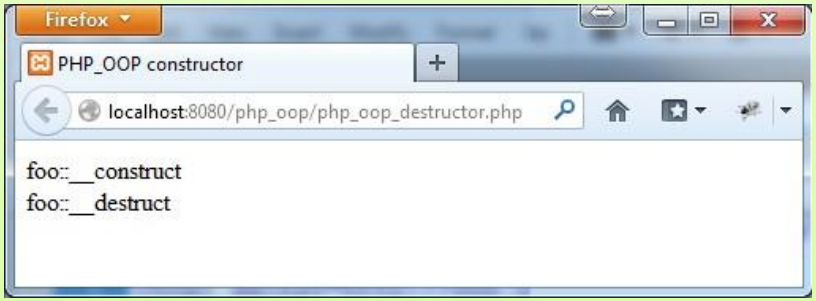
```
class foo {
    function__construct()
    {
        // PHP 5 new  style constructor
        echo__METHOD__;
    }
    function foo()
    {
        // PHP 4 style constructor
    }
}
new foo();
```



Destructors

- **Cú pháp hàm hủy**:

```
class foo {
    function__construct()
    {
        echo__METHOD__. PHP_EOL;
    }
    function__destruct()
    {
        echo__METHOD__;
    }
}
new foo();
```



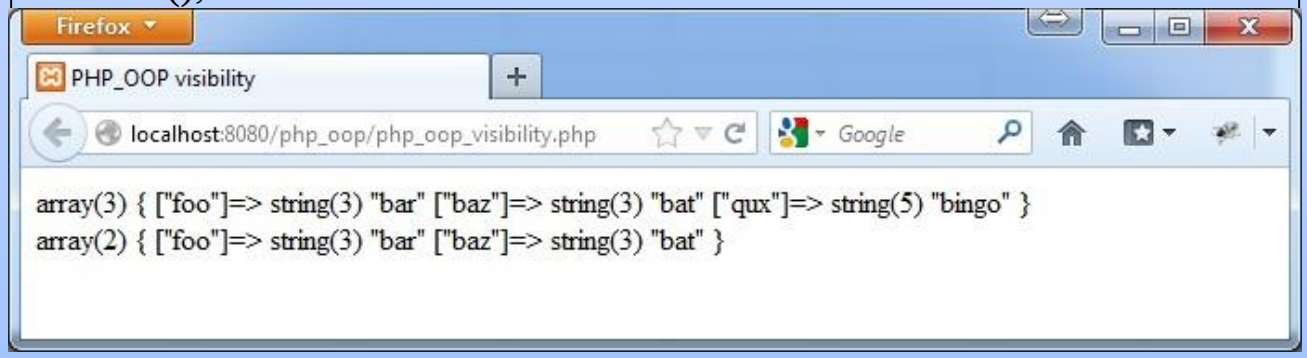
- **Phạm vi truy cập**:

Key	Visibility
public	The resource can be accessed from any scope.
protected	The resource can only be accessed from within the class where it is defined and its descendants
private	The resource can only be accessed from within the class where it is defined.
final	The resource is accessible from any scope, but cannot be overridden in descendant classes.

- **Ví dụ**:

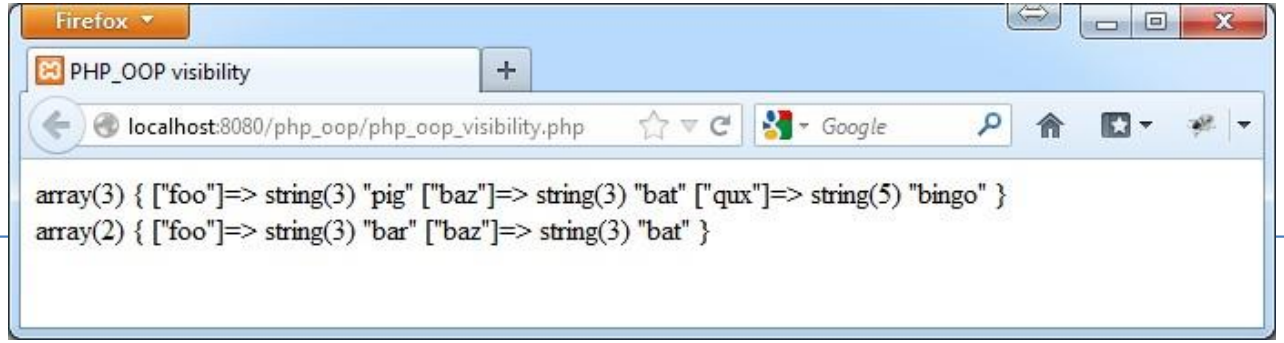
- **Ví dụ 1**: kết quả của đoạn lệnh sau

```
class foo {
    public $foo = 'bar';
    protected $baz = 'bat';
    private $qux = 'bingo';
    function__construct(){
        var_dump(get_object_vars($this));
    }
}
class bar extends foo {
    function __construct(){
        var_dump(get_object_vars($this));
    }
}
new foo();
```



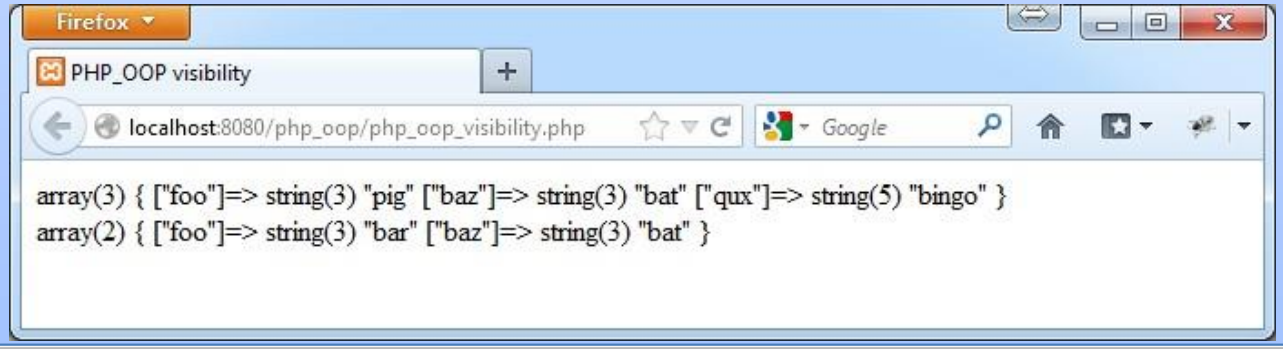
- **Ví dụ 2**: kết quả của đoạn lệnh sau

```
class foo {
    public $foo = 'bar';
    protected $baz = 'bat';
    private $qux = 'bingo';
    function __construct(){
        $this->foo="pig";
        var_dump(get_object_vars($this)); echo "<br>";
    }
}
class bar extends foo {
    function __construct(){
        var_dump(get_object_vars($this)); echo "<br>";
    }
}
new foo();
new bar();
```



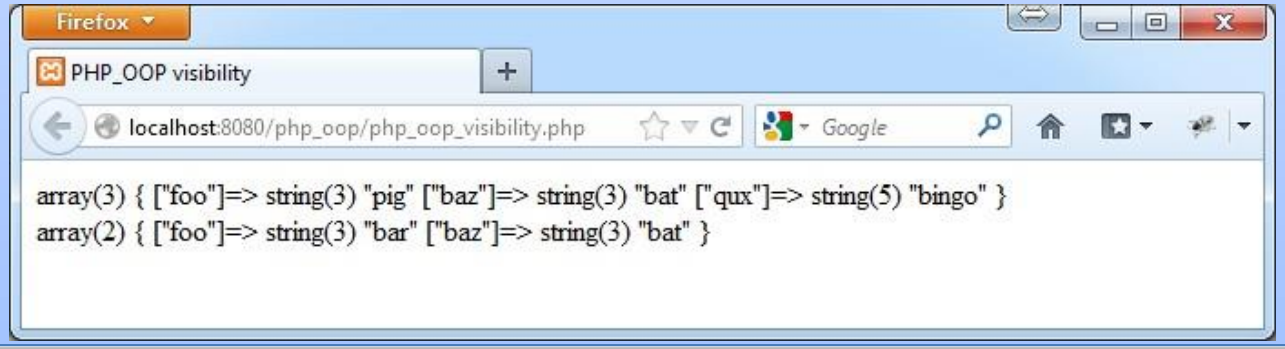
- **Ví dụ 2**: kết quả của đoạn lệnh sau

```
class foo {
    public $foo = 'bar'; protected $baz = 'bat'; private $qux = 'bingo';
    function__construct(){
        $this->foo="pig";
        var_dump(get_object_vars($this));
    }
}
class bar extends foo {
    function __construct(){
        var_dump(get_object_vars($this));
    }
}
new foo();
new bar();
```



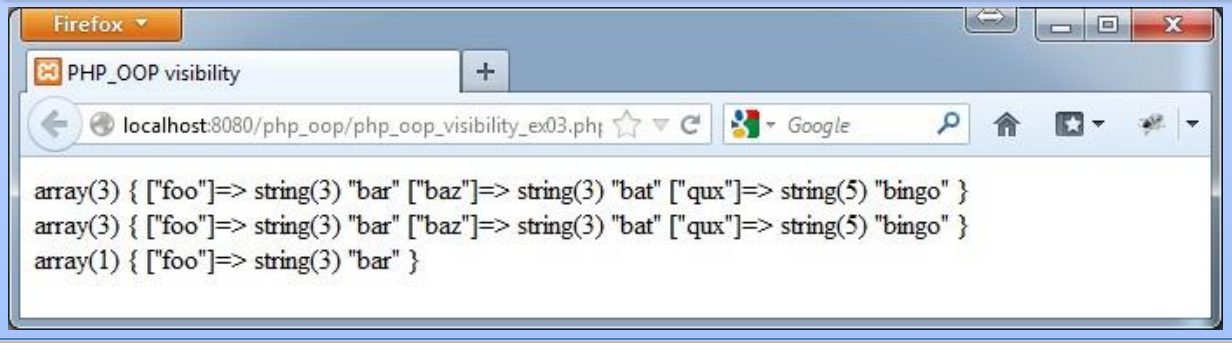
- **Ví dụ 3**: kết quả của đoạn lệnh sau

```
class foo {
    public $foo = 'bar'; protected $baz = 'bat'; private $qux = 'bingo';
    function__construct(){
        $this->foo="pig";
        var_dump(get_object_vars($this));
    }
}
class bar extends foo {
    function __construct(){
        var_dump(get_object_vars($this));
    }
}
new foo();
new bar();
```



- **Ví dụ 3**: kết quả của đoạn lệnh sau

```
class foo {
    public $foo = 'bar'; protected $baz = 'bat'; private $qux = 'bingo';
    function__construct(){
        var_dump(get_object_vars($this)); echo "<br>";
    }
}
class baz {
    function__construct() {
        $foo = new foo();
        var_dump(get_object_vars($foo)); echo "<br>";
    }
}
new foo();
new baz();
```



Constants, Static Methods and Properties

- **Cú pháp khai báo biến và phương thức tĩnh:**

```
class foo {
    static $bar = "bat";
    static public function baz(){
        echo "Hello World";
    }
}
echo foo::$bar."<br>";
foo::baz();
```

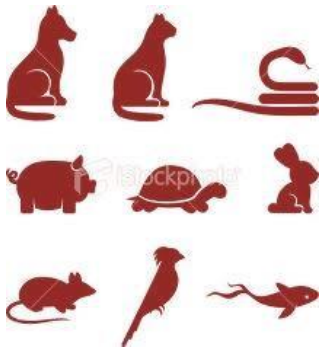
```
// output:
bat
Hello world
```

- **Cú pháp khai báo hằng trong lớp:**

```
class foo {
    const BAR = "Hello World";
}
echo foo::BAR;
```

```
// output:
Hello world
```

Giới thiệu



Animal

Transport

Interface
- ✓ Go
- ✓ Run
- ✓ Fly
- ✓ Swim
- ✓ …

Interface
- ✓ Go
- ✓ …

Chim và máy bay có cùng interface Fly nhưng cách thức hoạt động của Fly là khác nhau hoàn toàn

Interface
- ✓ Run
- ✓ …

b. Lớp trừu tượng

- **Khái niệm**
  - Lớp trừu tượng là một lớp cha cho tất cả các lớp có cùng bản chất. Do đó mỗi lớp dẫn xuất (lớp con) chỉ có thể kế thừa từ một lớp trừu tượng.
  - Lớp trừu tượng không cho phép tạo instance (không thể tạo được các đối tượng thuộc lớp đó).
- **Cú pháp khai báo lớp trừu tượng**

```
abstract class<class_name>{
  [properties … ]
   abstract function func_name(…);
   public function func_name(…);
}
```

b. Lớp trừu tượng

- **Ví dụ**

```
abstract class DataStore_Adapter {
    private $id;
    abstract function insert();
    abstract function update();
    public function save(){
        if (!is_null($this->id)){
            $this->update();
        } else {
            $this->insert();
        }
    }
}
class PDO_DataStore_Adapter extends DataStore_Adapter {
    public__construct($dsn){ // ...    }
    function insert(){    // ...    }
    function update(){ // ...    }
}
```

c. Lớp interface

- **Khái niệm**
  - Lớp interface được xem như một mặt nạ cho tất cả các lớp cùng cách thức hoạt động nhưng có thể khác nhau về bản chất.
  - Lớp dẫn xuất có thể kế thừa từ nhiều lớp interface để bổ sung đầy đủ cách thức hoạt động của mình (đa kế thừa - Multiple inheritance).
- **Cú pháp khai báo lớp interface**

```
interface class <class_name>{
  …
}
```

c. Lớp interface

- Ví dụ

```
interface DataStore_Adapter {
    public function insert();
    public function update();
    public function save();
    public function newRecord($name = null);
}


  class PDO_DataStore_Adapter implements DataStore_Adapter {
      public function insert(){   // ...    }
                                                }
       public function save(){          // ...    }
       public function newRecord($name = null){ }
    }
```

Include() và requeire()

a. Công dụng

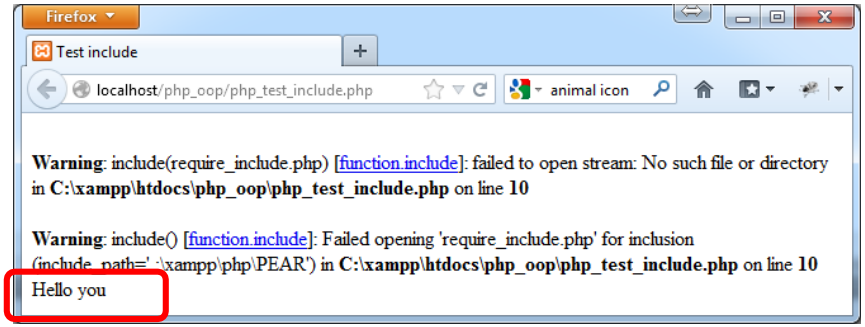
- **Giống nhau**:
  - Chèn file vào file hiện tại, nếu file được chèn có lỗi thì hiện thông báo lỗi
- **Khác nhau**:
  - Khi file được chèn bằng lệnh **require**() có lỗi thì trình biên dịch sẽ dừng lại, không dịch nữa và sẽ xuất hiện thông báo lỗi.
  - Khi file được chèn bằng lệnh **include**() có lỗi thì trình biên dịch vẫn tiếp tục dịch cho đến hết, đồng thời cũng xuất hiện ***warning*** để cảnh báo file đó bị lỗi.

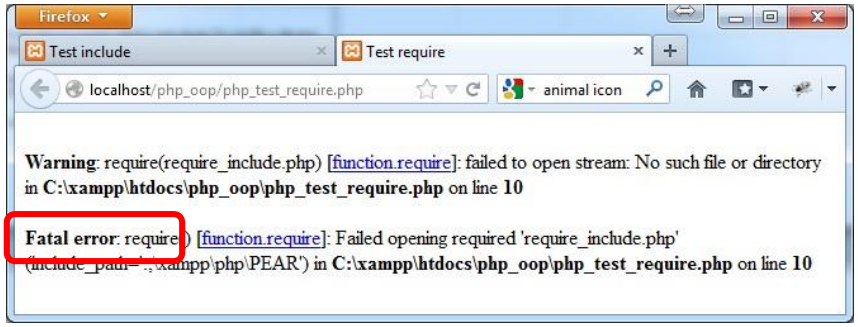
b. Ví dụ

- Giả sử 2 đoạn chương trình sau cùng sử dụng tập tin **require_include.php** không tồn tại như sau:

```
<html>
<head><title>Test include</title></head>
<body>
<?php
    include("require_include.php");
    echo "Hello you";
?>
</body>
</html>
```

```
<html>
<head><title>Test include</title></head>
<body>
<?php
    require("require_include.php");
     echo "Hello you";
?>
</body>
</html>
```





c. Hàm include_once và require_once

- Là 2 hàm biến đổi của hàm include và require nhằm mục đích nếu tập tin đã được chèn trước đó thì không chèn nữa.
- **Ví dụ**: giả sử có tập tin **require_include_once.php** như sau:

```php
<?php
    // Tập tin require_include_once.php
    echo "Hello you <br>";
?>
```

```php
<!– Tập tin test_require_include_once.php -->
<html>
<head><title>Test include_once</title></head>
<body>
<?php
    include_once('require_include_once.php');
    include_once('require_include_once.php');
?>
</body>
</html>
```