MỤC LỤC

Namespace - PHP : Cách dùng Namespace trong PHP 5.3 trở lên

A- Giới thiệu

Bắt đầu từ phiên bản PHP 5.3 trở lên. PHP đã có thêm 1 chức năng nữa là namespace. Namespace được sử dụng để định danh 1 lớp một cách cụ thể hóa. Trước đây, nếu chúng ta xây dựng 1 lớp ABC, và vô tình nạp 1 file từ thư viện khác cũng có lớp ABC. Vậy lúc này hệ thống sẽ xử lý như thế nào ?. Sử dụng thư viện ABC của mình hay của file khác ?.

Ở phiên bản zend framework 1.x. Nhận thấy sự bất tiện này, nên zend đã xây dựng cho mình 1 phương pháp đọc định danh dựa trên 1 lớp. Ví dụ như: new Zend_Controller_Action. Có nghĩa là nó sẽ tìm từ zend/controller/action.php

Nhận thấy sự bất cập này, PHP đã bổ sung chức năng namespace để tạo định danh cho các lớp 1 cách rõ ràng hơn. Ví dụ lớp ABC của chúng ta thuộc namespace home và lớp ABC bên file khác thuộc định danh other. Khi đó, việc sử dụng và tương tác giữa chúng hoàn toàn có thể kiểm soát 1 cách dễ dàng.

B- Sử dụng namespace như thế nào:

B.1 Sử dụng namespace đơn giản

Để sử dụng namespace, chúng ta định nghĩa bằng cách dùng từ khóa namespace để tạo vùng định danh.

Ví dụ:

```
<?php
namespace Codeto;
?>
```

Namespace có thể sử dụng để định danh cho 1 lớp, 1 hằng hoặc 1 hàm. Và khi sử dụng thì namespace phải **đứng trên mọi thẻ**. Nghĩa là **trên namespace không được có khoảng trắng hoặc HTML**.

Ví dụ:

Tạo file codeto.php với nội dung:

```php
<?php
namespace Codeto;
class demo{
    public function __construct(){
        echo "Hello Codeto<br />";
    }
    public function say(){
        echo "Toi la Son den tu Viet Nam.";
    }
}
const NAME="Ha Anh Son";
?>
```

Tạo file hello.php với nội dung:

```php
<?php
require("codeto.php");
$b=new demo();
$b->say();
?>
```

Nếu để như thế này khi chạy các bạn sẽ bị lỗi :

```
Fatal error: Class 'demo' not found in C:\xampp\htdocs\hello.php on line
3
```

Như vậy là các bạn thấy hệ thống không thể tìm thấy class **'demo'** đâu cả vì toàn bộ nội dung của file **codeto.php** đã được bảo vệ do sử dụng namespace **Codeto**.vậy là dù chúng ta có gọi tới file đó thì cũng không thể truy cập vào lớp hoặc phương thức trong file này được.Vậy để sử dụng được lớp và phương thức này trong file hello.php ta sẽ viết lại nội dung như sau:

```php
<?php
require("codeto.php");
$b=new Codeto\demo;
$b->say();
echo Codeto\NAME;
?>
```

Như bạn thấy, chúng ta đã sử dụng namespace ở file hello.php thật đơn giản phải không nào. Dùng cho lớp demo và dùng cho hằng ***NAME*** mà ta đã định nghĩa ở file codeto.php.

Lưu ý : ở ví dụ trên nếu hiện lỗi :

```
Fatal error: Namespace declaration statement has to be the very first
statement in the script
```

Thì phải kiểm tra lại xem file **hello.php** có chứa dấu cách hay không, vì trước namespace không được có khoảng trắng, và đặt encoding trong Notedpad++ là UTF8 without BOM.

B.2: Tạo namespaces theo cấu trúc phân cấp

Ngoài cách định nghĩa namespace đơn giản như trên thì các bạn hoàn toàn có thể định nghĩa không gian tên này theo dạng phân cấp.

Ví dụ :

Tạo file codeto.php với nội dung:

```php
<?php
namespace Application\Codeto;
//chú ý : ở ví dụ phần B1 thì namespace chỉ là Codeto
class demo{
    public function __construct(){
        echo "Hello Codeto Application<br />";
    }
    public function say(){
        echo "Toi la Son den tu Viet Nam.";
    }
}
const NAME="Ha Anh Son";
?>
```

Tạo file hello.php với nội dung:

```php
<?php
require("codeto.php");
$b=new Application\Codeto\demo;
$b->say();
echo Application\Codeto\NAME;
?>
```

B.3: Nạp, gọi thêm một namespace từ bên ngoài :

PHP 5.3 hỗ trợ từ khóa use. Với từ khóa này chúng ta hoàn toàn có thể triệu gọi 1 namespace từ bên ngoài vào dễ dàng.

Ví dụ:
Tạo file codeto.php với nội dung:

```
<?php
namespace Codeto;
class demo{
    public function __construct(){
        echo "Hello Codeto<br />";
    }
    public function say(){
        echo "Toi la Son den tu Viet Nam.";
    }
}
const NAME="Ha Anh Son";
?>
```

Tạo tiếp file hello.php với nội dung:

```
<?php
namespace Codeto_new;
require("codeto.php");
use Codeto;
class demo_new{
    public function say_hello(){
        $data = new Codeto\demo;
        $data->say();
        echo Codeto\NAME;
    }
}
$test1= new demo_new();
$test1->say_hello();
?>
```

Ở ví dụ trên chúng ta sẽ chạy được kết quả như ý muốn,

như các bạn thấy trong file **hello.php** có thể sử dụng namespace của của file codeto.php thông qua từ khóa use,và trong **phương thức say_hello()** chúng ta có thể khởi tạo lớp **demo** và sử dụng các phương thức có sẵn cách dễ dàng.

B.4 Sử dụng định danh cho Namespace

Nếu tên định danh quá dài do cách đặt namespace phân cấp, chúng ta có thể xem xét dùng từ khóa as để định danh chúng.

Xét ví dụ sau:Tạo file codeto.php với nội dung:

```php
<?php
namespace Application\Controller\Codeto;
//note : namespace quá dài
class demo{
   public function __construct(){
      echo "Hello Codeto Application<br />";
   }
   public function say(){
      echo "Toi la Son den tu Viet Nam.";
   }
}
const NAME="Ha Anh Son";
?>
```

Tạo tiếp file hello.php với nội dung:

```php
<?php
require("codeto.php");
use Application\Controller\Codeto as AppCodeto;
$data= new AppCodeto\demo();
$data->say();
?>
```

Như, chúng ta thấy. Vì tên định danh quá dài, nên tôi đặt nó ngắn gọn là AppCodeto. Và để sử dụng nó, ta hoàn toàn có thể thao tác bình thường như dùng 1 namespace độc lập.

B.5 Sử dụng từ khóa mặc định của namespace:

Các phiên bản PHP từ 5.3 trở lên cung cấp cho ta 1 từ khóa __NAMESPACE__ Và với từ khóa này, chúng ta hoàn toàn có thể biết được vùng namespace hiện tại của mình là gì.

```
<?php
namespace Codeto;
class demo{
    public function __construct(){
        echo "Hello Codeto Application<br />";
    }
}
echo __NAMESPACE__;
?>
```

Kết quả sẽ hiển thị tên namespace là Codeto.

KẾT LUẬN:

Qua bài này mình đã giới thiệu cho các bạn về khái niệm cũng như cách sử dụng Namespace trong PHP 5.3 trở lên,nó cũng rất dễ sử dụng.Nếu các bạn muốn nghiên cứu tới các PHP framework mới thì cần nắm vững về cách sử dụng namespace này.