Bài 2
Mảng và hàm trong PHP

Module: **ADVANCED PROGRAMMING WITH PHP 2.0**

Mục tiêu

- Sử dụng được cú pháp PHP để thao tác với phương thức
- Sử dụng được cú pháp PHP để thao tác với mảng

Thảo luận

Mảng

Mảng

- Mảng là cơ chế giúp lưu trữ nhiều giá trị trong cùng một biến
- Các giá trị được lưu trong mảng thường là cùng loại
- Chẳng hạn: danh sách khách hàng, danh sách sản phẩm…
- Các khái niệm:
  - Tên mảng: tuân thủ các quy tắc như tên biến
  - Phần tử mảng: Từng giá trị trong mảng
  - Độ dài mảng: Số lượng phần tử của mảng
  - Chỉ số: Vị trí của các phần tử trong mảng.
    - Chỉ số của phần tử đầu tiên là 0
    - Chỉ số của phần tử cuối cùng là n – 1 (trong đó n là độ dài của mảng)

Khai báo mảng

- Cú pháp:

```
array(); or []
```

- Ví dụ:

```php
<?php
  $cars = array("Volvo", "BMW", "Toyota");
  // or
  $cars = ["Volvo", "BMW", "Toyota”];
?>
```

- Trong đó:
  - **cars** là tên mảng
  - “**Volvo**”, “**BMW**” và “**Toyota**” là các phần tử của mảng

Truy xuất các phần tử của mảng

- Có thể truy xuất các phần tử của mảng theo chỉ số
- Ví dụ:

```php
<?php
 $cars = ["Volvo", "BMW", "Toyota"];
 echo "I like " . $cars[0] . ", " . $cars[1] . " and
" . $cars[2] . ".";
?>
```

- Cũng có thể gán giá trị cho các phần tử của mảng, ví dụ:

```
$cars[0] = "Volvo";
$cars[1] = "BMW";
$cars[2] = "Toyota";
```

Độ dài của mảng

- Có thể sử dụng hàm count() để lấy độ dài của mảng
- Ví dụ:

```php
<?php
  $cars =  ["Volvo", "BMW", "Toyota"];
  echo count($cars); // 3
?>
```

Duyệt mảng sử dụng for

- Có thể sử dụng vòng lặp for để duyệt qua các phần tử của mảng
- Ví dụ:

```php
<?php
$cars = ["Volvo", "BMW", "Toyota"];
$arrLength = count($cars);

for($x = 0; $x < $arrLength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>
```

Mảng liên kết (Associative Array)

- Associative Array là mảng sử dụng các khóa bằng tên thay cho chỉ số thông thường
- Ví dụ:

```
$age = ["Peter"=>"35", "Ben"=>"37", "Joe"=>"43"];
```

- Có thể truy xuất phần tử qua tên khóa, ví dụ:

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

Duyệt Associative Array dùng foreach

- Có thể sử dụng foreach để duyệt qua tất cả các phần tử của một associative array
- Ví dụ:

```php
<?php
$age = ["Peter"=>"35", "Ben"=>"37", "Joe"=>"43"];

foreach($age as $key => $value) {
    echo "Key=" . $key . ", Value=" . $value;
    echo "<br>";
}
?>
```

Mảng nhiều chiều

- Mảng nhiều chiều là mảng có các phần tử là các mảng
- Có thể có mảng 2 chiều, 3 chiều, 4 chiều… hoặc nhiều hơn
- Thông thường chúng ta chỉ làm việc với mảng 2 chiều, vì các mảng nhiều chiều hơn thì khó quản lý hơn
- Mảng nhiều chiều thì cần nhiều chỉ số để có thể truy xuất đến một phần tử, chẳng hạn:
  - Mảng 2 chiều thì cần 2 chỉ số để truy xuất phần tử
  - Mảng 3 chiều thì cần 3 chỉ số để truy xuất phần tử

Khai báo mảng nhiều chiều

- Ví dụ:

```
$cars = [
      ["Volvo",22,18],
      ["BMW",15,13],
      ["Saab",5,2],
      ["Land Rover",17,15]
    ];
```

Duyệt mảng nhiều chiều

- Sử dụng các vòng lặp lồng nhau để duyệt qua tất cả các phần tử của mảng nhiều chiều
- Ví dụ:

```php
<?php
for ($row = 0; $row < 4; $row++) {
   echo "<ul>";
  for ($col = 0; $col < 3; $col++) {
    echo "<li>".$cars[$row][$col]."</li>";
  }
  echo "</ul>";
}
?>
```

- **Lưu ý:** Càng nhiều vòng lặp lồng nhau thì độ phức tạp của thuật toán càng cao

Thảo luận

Hàm

Hàm trong PHP

- PHP cung cấp sẵ hơn 1000 hàm
- Chúng ta có thể định nghĩa thêm các hàm mới
- Hàm:
  - Một khối lệnh thực hiện một nhiệm vụ nhất định
  - Có thể được sử dụng lặp đi lặp lại nhiều lần
  - Không được thực thi ngay tức thì khi nạp trang php
  - Chỉ được thực thi tại lời gọi hàm

Cú pháp khai báo hàm

```
function functionName() {
    code to be executed;
}
```

- Tên hàm phải bắt đầu bằng ký tự chữ hoặc dấu gạch dưới
- Tên hàm không phân biệt chữ hoa và chữ thường
- Nên đặt tên hàm có ý nghĩa

```php
<?php
function writeMessage() {
    echo "Hello world!";
}
?>
```

Tham số của hàm

- Tham số là cơ chế để truyền dữ liệu đầu vào cho hàm
- Tham số giống như là một biến
- Có thể truyền một hoặc nhiều tham số vào hàm
- Các tham số cách nhau bởi dấu phẩy (,)
- Các tham số được đặt trong cặp dấu '(' và ')' ngay sau tên hàm

```
function functionName($agr1, $agr2, $agr3) {
    code to be executed;
}
```

Tham số của hàm: ví dụ

```
function substr ( $string , $start, $length){
   //Code here
}
```

- Hàm substr được sử dụng để lấy một chuỗi con
- Hàm substr có 3 tham số:
  - $string: chuỗi nguồn
  - $start: vị trí bắt đầu của chuỗi con
  - $length: độ dài của chuỗi con

Giá trị mặc định của tham số

- Có thể quy định giá trị mặc định của tham số
- Khi gọi hàm mà không truyền giá trị vào cho tham số đó thì giá trị mặc định sẽ được sử dụng
- Ví dụ:

```
function substr ( $string , $start, $length = null){
   //Code here
}
```

- Nếu gọi hàm mà không truyền giá trị vào cho $length thì mặc định $length sẽ có giá trị là null

Giá trị trả về của hàm

- Từ khóa return được sử dụng để "trả về" một giá trị trong hàm
- Có thể hiểu "giá trị trả về" là "đầu ra" của hàm
- Mỗi hàm chỉ có thể trả về một giá trị
- Nếu muốn trả về nhiều giá trị thì có thể sử dụng mảng hoặc đối tượng
- Ví dụ:

```php
<?php
function sum($firstOperand, $secondOperand) {
    $result = $firstOperand + $secondOperand;
    return $result;
}
```

Nhúng trang PHP

- Có thể nhúng nội dung của một trang PHP vào trong trang PHP khác
- Có một số cơ chế khác nhau:
  - Sử dụng include
  - Sử dụng include_once
  - Sử dụng require
  - Sử dụng require_once
- Nhúng trang PHP rất hữu ích trong những trường hợp muốn tái sử dụng lại các file PHP đã có, chia sẻ mã nguồn giữa các file

include và require

- include và require đều được sử dụng để nhúng trang PHP
- Điểm khác biệt là khi nếu xảy ra lỗi:
  - require sẽ tung ra lỗi nghiêm trọng (E_COMPILE_ERROR) và dừng thực thi
  - include sẽ tung ra cảnh báo (E_WARNING) và tiếp tục thực thi
- Cú pháp:

```
include 'filename';

 // hoặc

require 'filename';
```

- Trong đó ‘filename’ là đường dẫn đến file muốn nhúng

include_once và require_once

- Chức năng của include_once và require_once là giống với include và require
- Điểm khác biệt là include_once và require_once chỉ nhúng trang PHP một lần duy nhất
- include và require có thể nhúng trang PHP nhiều lần nếu được sử dụng nhiều nơi

Xem log của ứng dụng PHP

- Log của ứng dụng PHP có thể lưu trữ ở một vài nơi khác nhau
- Có 2 loại log quan trọng:
  - Error log: Log ghi nhận các lỗi xảy ra
  - Access log: Log ghi nhận các truy cập lên ứng dụng
- Có thể vào file php.ini để xem cấu hình nơi lưu trữ log, chẳng hạn:

  error_log = /var/log/php-scripts.log
- Vị trí lưu trữ log phụ thuộc vào hệ điều hành đang dùng, web server đang dùng và cấu hình của ứng dụng web

Một số vị trí của log

- Một số vị trí thông thường của file log:
  - /var/log/httpd/
  - /var/log/apache2/

- Một số framework có thể cung cấp file log khác. Chẳng hạn Wordpress hoặc Laravel có các file log của riêng mình

Tóm tắt bài học

- Mảng cho phép lưu trữ nhiều giá trị cùng kiểu
- Các khái niệm của mảng: Tên mảng, kiểu dữ liệu, kích thước, phầntử, chỉ số
- Chỉ số của phần tử đầu tiên là 0. Chỉ số của phần tử cuối cùng là length – 1
- Có thể sử dụng vòng lặp for và for-each để duyệt mảng
- Phương thức giúp tái sử dụng mã nguồn. Các thành phần của một phương thức: modifier, kiểu dữ liệu trả về,tên phương thức, danh sách tham số
- Đặt tên cho phương thức là một thao tác rất quan trọng để đảm bảo clean code
- Câu lệnh return được sử dụng để trả về giá trị của một phương thức

Hướng dẫn

- Hướng dẫn làm bài thực hành và bài tập
- Chuẩn bị bài tiếp: *Lớp và đối tượng trong PHP*