LẬP TRÌNH PHP

NỘI DUNG

- 1. Giới thiệu
- 2. Cấu trúc PHP
- 3. Kiểu dữ liệu, hằng và biến
- 4. Các phép toán trong PHP
- 5. Các cấu trúc điều khiển
- 6. Hàm trong PHP
- 7. Mảng (array)

3. Biến

- **Một biến trong PHP là tên của vị trí bộ nhớ chứa dữ liệu**. Biến là bộ nhớ tạm thời được sử dụng để lưu trữ dữ liệu tạm thời.
- Trong PHP, một biến được khai báo sử dụng ký hiệu $ theo sau là tên biến.
- Cú pháp khai báo một biến trong PHP được đưa ra dưới đây:

```
$ten_bien = gia_tri;
```

3. Biến

Quy tắc đặt tên biến trong PHP

- Biến bắt đầu bằng ký hiệu $, theo sau là tên của biến.
- Các biến PHP phải bắt đầu bằng chữ cái hoặc dấu gạch dưới.
- Biến PHP không thể bắt đầu bằng số và ký tự đặc biệt.
- Tên biến chỉ có thể chứa ký tự chữ và số và dấu gạch dưới (Az, 0-9 và _).
- Tên biến phân biệt chữ hoa chữ thường ($age và $AGE là hai biến khác nhau).

```
<?php
        $a = "hello"; //bắt đầu bằng chữ cái (hợp lệ)
        $_b = "hello"; //bắt đầu bằng dấu gạch dưới (hợp lệ)
        echo "$a <br/> $_b";
?>
```

3. Biến

Quy tắc đặt tên biến trong PHP

Biến không hợp lệ

```
<?php
        $4c = "hello"; //bắt đầu bằng chữ số (không hợp lệ)
        $*d = "hello"; //bắt đầu bằng ký tự đặc biệt (không hợp lệ)
        echo "$4c <br/> $*d";
?>
```

Biến trong PHP: Khai báo biến String, integer và float

```
<?php
        $str = "hello string";
        $x = 200;
        $y = 44.6;
        echo "string is: $str <br/>";
        echo "integer is: $x <br/>";
        echo "float is: $y <br/>";
?>
```

3. Biến

Quy tắc đặt tên biến trong PHP

Biến trong PHP: Chữ hoa, chữ thường

```
<?php
        $studentName = "David Q";
        echo "student name 1: " . $studentName."<br>";
        echo "student name 2: " . $StudentName."<br>";
        echo "student name 3: " . $studentname."<br>";
?>
```

Biến trong PHP: Khai báo biến String, integer và float

```
<?php
        $str = "hello string";
        $x = 200;
        $y = 44.6;
        echo "string is: $str <br/>";
        echo "integer is: $x <br/>";
        echo "float is: $y <br/>";
?>
```

3. Biến

- Phạm vi của biến trong PHP
- Trong PHP, các biến có thể được khai báo ở bất kỳ đâu trong kịch bản lệnh.
- Phạm vi của một biến là một phần của kịch bản mà biến có thể được tham chiếu/sử dụng.
- PHP có 3 phạm vi biến khác nhau:
  - ✓ Biến toàn cục (global).
  - ✓ Biến cục bộ (local).
  - ✓ Tĩnh (static).

3. Biến

- **Phạm vi của biến toàn cục**

```php
<?php
$x = 5; // phạm vi toàn cục

function myTest() {
   // sử dụng biến $x bên trong hàm này sẽ bị lỗi
   echo "<p>Biến x bên trong hàm là: $x</p>";
}

echo "<p>Biến x bên ngoài hàm là: $x</p>";

myTest();
?>
```

3. Biến

- **Phạm vi của biến cục bộ**

```
<?php
function myTest() {
    $x = 5; // phạm vi biến cục bộ
    echo "<p>Biến x bên trong hàm là: $x</p>";
}
  // sử dụng biến $x bên ngoài hàm này sẽ bị lỗi
echo "<p>Biến x bên ngoài hàm là: $x</p>";

myTest();
?>
```

❑ 3. Biến

Từ khóa global trong PHP

- Từ khóa **global** được sử dụng để truy cập vào một biến toàn cầu từ bên trong một hàm.
- Để làm điều này, hãy sử dụng từ khóa **global** trước các biến (bên trong hàm):

```php
<?php
$x = 5;
$y = 10;

function myTest() {
  global $x, $y;
  $y = $x + $y;
}

myTest();
echo $y; // kết quả là 15
?>
```

❑ 3. Biến

Từ khóa global trong PHP

- PHP cũng lưu trữ tất cả các biến toàn cục trong một mảng gọi là **$GLOBALS [index].** Các chỉ số index giữ tên của biến. Mảng này cũng có thể truy cập từ bên trong các hàm và có thể được sử dụng để cập nhật các biến toàn cục trực tiếp.
- Ví dụ trên có thể được viết lại như sau:

```
<?php
$x = 5;
$y = 10;

function myTest() {
   $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];
}

myTest();
echo $y; // kết quả là 15
?>
```

3. Biến

Từ khóa static trong PHP

- Thông thường, khi một hàm được thực thi hoàn thành, tất cả các biến của nó sẽ bị xóa. Tuy nhiên, đôi khi chúng ta muốn một biến địa phương KHÔNG bị xóa. Chúng ta cần sử dụng nó cho một công việc khác.
- Để thực hiện việc này, hãy sử dụng từ khóa **static** khi bạn khai báo biến:

```
<?php
function myTest() {
  static $x = 0;
  echo $x . "<br>";
  $x++;
}
myTest();
myTest();
myTest();
?>
```

Kết quả:

```
0
1
2
```

Sau đó, mỗi lần hàm được gọi, biến đó sẽ vẫn có thông tin từ lần cuối hàm được gọi. Biến static vẫn là biến địa phương của hàm.

3. Biến

Biến $ và $$ trong PHP

- **$var (đô la đơn) là một biến bình thường** với tên var lưu trữ bất kỳ giá trị như chuỗi, số nguyên, float, v.v.
- **$$var (đô la kép) là một biến tham chiếu** lưu trữ giá trị của biến $ bên trong nó.

```
<?php
$x = "abc";
$$x = 200;                  // tạo biến : $abc=200
echo $x."<br/>";            // output: “abc”
echo $$x."<br/>";           // output: 200
echo $abc;                  // output: 200
?>
```

3. Hằng số

- **Hằng số trong PHP là tên hoặc mã định danh không thể thay đổi trong khi thực thi chương trình**. Các hằng số PHP có thể được định nghĩa theo 2 cách:

  1.Sử dụng hàm **define**()

  2.Sử dụng từ khóa **const**.

- Các hằng số trong PHP tuân theo các quy tắc tương tự như biến PHP. Ví dụ: chỉ có thể bắt đầu bằng chữ cái hoặc dấu gạch dưới.
- Thông thường, hằng số PHP nên được định nghĩa bằng chữ in hoa.

Cú pháp của hàm define() trong PHP.

define(name, value, case-insensitive)

3. Hằng số

Cú pháp của hàm define() trong PHP.

define(name, value, **case**-insensitive)

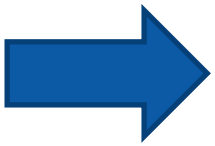
Trong đó:

1.name: chỉ định tên hằng số.

2.value: định nghĩa giá trị không đổi.

3.case-insensitive: không phân biệt chữ hoa chữ thường, giá trị mặc định là **false**. Nó có nghĩa là nó phân biệt chữ hoa chữ thường theo mặc định.

```
<?php
        define("MESSAGE", "Hello PHP");
        echo MESSAGE;
?>
```



Kết quả:

```
Hello PHP
```

3. Hằng số

Cú pháp của hàm define() trong PHP.

define(name, value, **case**-insensitive)

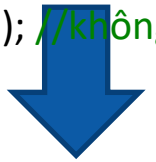
Trong đó:

1.name: chỉ định tên hằng số.

2.value: định nghĩa giá trị không đổi.

3.case-insensitive: không phân biệt chữ hoa chữ thường, giá trị mặc định là **false**. Nó có nghĩa là nó phân biệt chữ hoa chữ thường theo mặc định.

```
<?php
        define("MESSAGE", "Hello PHP", true); //không phân biệt chữ hoa chữ thường
        echo MESSAGE;
        echo message;
?>
```



Kết quả:

```
Hello PHP
Hello PHP
```

3. Hằng số

Cú pháp của hàm define() trong PHP.

define(name, value, **case**-insensitive)

Trong đó:

1.name: chỉ định tên hằng số.

2.value: định nghĩa giá trị không đổi.

3.case-insensitive: không phân biệt chữ hoa chữ thường, giá trị mặc định là **false**. Nó có nghĩa là nó phân biệt chữ hoa chữ thường theo mặc định.

```
<?php
        define("MESSAGE", "Hello PHP", false); // phân biệt chữ hoa chữ thường
        echo MESSAGE;
        echo message;
?>
```

Hello PHP

Fatal error: Uncaught Error: Undefined constant "message" in D:\xampp\htdocs\demo\vd5.php:32
Stack trace: #0 {main} thrown in **D:\xampp\htdocs\demo\vd5.php** on line **32**

3. Hằng số

Hằng số trong PHP: Sử dụng từ khóa const

- Từ khóa const định nghĩa các hằng số tại thời gian biên dịch.
- Nó nhanh hơn một chút so với define().
- Nó phân biệt chữ hoa và chữ thường.

```php
<?php
        const MESSAGE="Hello const PHP";
        echo MESSAGE;
?>
```

Kết quả:

```
Hello const PHP
```

3. Kiểu dữ liệu

Các biến trong PHP có thể lưu trữ được các kiểu dữ liệu khác nhau. **Có các kiểu dữ liệu trong PHP như sau**:

- String
- Integer
- Float (số dấu phẩy động - còn được gọi là double)
- Boolean
- Array
- Object
- NULL

❑ 3. Kiểu dữ liệu

PHP String

- Một string là một chuỗi các ký tự, như "Hello world!".
- Một chuỗi có thể là bất kỳ ký tự nào bên trong dấu ngoặc kép. Bạn có thể sử dụng dấu nháy đơn hoặc kép:

```
<?php
        $x = "Hello world!";
        $y = 'Hello world!';

        echo $x;
        echo "<br>";
        echo $y;
?>
```

3. Kiểu dữ liệu

PHP Integer

- Kiểu dữ liệu số nguyên là một số thập phân giữa -2,147,483,648 và 2,147,483,647.
- Quy tắc cho số nguyên:
  - ✓ Số nguyên phải có ít nhất một chữ số.
  - ✓ Số nguyên không được có dấu thập phân.
  - ✓ Một số nguyên có thể dương hoặc âm.

```
<?php
$x = 1000;
var_dump($x);
?>
```

Kết quả:

```
int(1000)
```

Hàm PHP var_dump($var) trả về kiểu dữ liệu và giá trị của biến $var.

3. Kiểu dữ liệu

PHP Float

- Float (số dấu phẩy động) là một số có dấu thập phân

```
<?php
$x = 3.14;
var_dump($x);
?>
```

Kết quả:

```
float(3.14)
```

PHP Boolean

- Boolean đại diện cho hai trạng thái: TRUE hoặc FALSE.

```
<?php
$x = true;
$y = false;
?>
```

3. Kiểu dữ liệu

PHP Array

- Một mảng lưu trữ nhiều giá trị trong một biến duy nhất.

```php
<?php
$cars = array("PHP", "Java", "VBA");
var_dump($cars);
?>
```

Kết quả:

```
array(3) { [0]=> string(3) "PHP" [1]=> string(4) "Java" [2]=> string(3) "VBA" }
```

3. Kiểu dữ liệu

PHP Object

- Một đối tượng (object) là một kiểu dữ liệu lưu trữ dữ liệu và thông tin về cách xử lý dữ liệu đó.
- Trong PHP, một đối tượng phải được khai báo một cách rõ ràng.
- Đầu tiên chúng ta phải khai báo một lớp đối tượng. Đối với điều này, chúng ta sử dụng từ khóa class. Một lớp là một cấu trúc có thể chứa các thuộc tính và các phương thức:

```php
<?php
class Car {
   function Car() {
      $this->model = "Honda";
   }
}

// tạo một đối tượng
$honda = new Car();

// hiển thị thuộc tính của đối tượng
echo $honda->model;
?>
```

Kết quả:

Honda

❑ 3. Kiểu dữ liệu

PHP giá trị NULL

- Null là một kiểu dữ liệu đặc biệt chỉ có thể có một giá trị: NULL.
- Một biến kiểu dữ liệu NULL là một biến không có giá trị nào được gán cho nó.
- Nếu một biến được tạo mà không có giá trị, nó sẽ tự động được gán giá trị NULL.
- Các biến cũng có thể được làm trống bằng cách đặt giá trị thành NULL:

```
<?php
$x = "Hello world!";
$x = null;
var_dump($x);
?>
```

❑ 4. Các phép toán trong PHP

Các **toán tử trong PHP** được sử dụng để thực hiện các phép toán trên các biến và các giá trị.

Toán tử trong PHP được chia thành các nhóm sau:

- ❖ **Toán tử số học**
- ❖ **Toán tử so sánh**
- ❖ **Toán tử logic**
- ❖ **Toán tử gán**
- ❖ **Toán tử điều kiện**

4. Các phép toán trong PHP

Toán tử số học trong PHP

- Bảng dưới liệt kê các toán tử số học được hỗ trợ bởi ngôn ngữ PHP:
- Giả sử biến A giữ giá trị 10, biến B giữ 20 thì:

Toán tử	Miêu tả	Ví dụ
+	Cộng hai toán hạng	A + B kết quả là 30
-	Trừ toán hạng thứ hai từ toán hạng đầu	A - B kết quả là -10
*	Nhân hai toán hạng	A * B kết quả là 200
/	Phép chia	B / A kết quả là 2
%	Phép lấy số dư	B % A kết quả là 0
++	Toán tử tăng, tăng giá trị toán hạng thêm một đơn vị	A++ kết quả là 11
--	Toán tử giảm, giảm giá trị toán hạng đi một đơn vị	A-- kết quả là 9

4. Các phép toán trong PHP

Toán tử so sánh trong PHP

- Bảng dưới liệt kê các toán tử so sánh được hỗ trợ bởi ngôn ngữ PHP.
- Giả sử biến A giữ giá trị 10, biến B giữ giá trị 20, thì:

Toán tử	Miêu tả	Ví dụ
==	Kiểm tra nếu 2 toán hạng bằng nhau hay không. Nếu bằng thì điều kiện là true.	(A == B) là false.
!=	Kiểm tra 2 toán hạng có giá trị khác nhau hay không. Nếu không bằng thì điều kiện là true.	(A != B) là true.
>	Kiểm tra nếu toán hạng bên trái có giá trị lớn hơn toán hạng bên phải hay không. Nếu lớn hơn thì điều kiện là true.	(A > B) là false.
<	Kiểm tra nếu toán hạng bên trái nhỏ hơn toán hạng bên phải hay không. Nếu nhỏ hơn thì là true.	(A < B) là true.
>=	Kiểm tra nếu toán hạng bên trái có giá trị lớn hơn hoặc bằng giá trị của toán hạng bên phải hay không. Nếu đúng là true.	(A >= B) là false.
<=	Kiểm tra nếu toán hạng bên trái có giá trị nhỏ hơn hoặc bằng toán hạng bên phải hay không. Nếu đúng là true.	(A <= B) là true.

❑ 4. Các phép toán trong PHP

Toán tử logic trong PHP

- Bảng dưới đây chỉ rõ tất cả các toán tử logic được hỗ trợ bởi ngôn ngữ PHP.

Toán tử	Miêu tả	Ví dụ
and	Được gọi là toán tử Logic AND. Nếu cả hai toán hạng là true thì điều kiện trở thành true	(A and B) là true.
or	Được gọi là toán tử Logic OR. Nếu một trong hai toán hạng là đúng thì điều kiện trở thành true	(A or B) là true.
&&	Được gọi là toán tử Logic AND. Nếu cả hai toán hạng là true thì điều kiện trở thành true	(A && B) là true.
||	Được gọi là toán tử Logic OR. Nếu một trong hai toán hạng là đúng thì điều kiện trở thành true	(A || B) là true.
!	Được gọi là toán tử Logic NOT. Sử dụng để đảo ngược trạng thái logic của toán hạng. Nếu điều kiện là true thì toán tử Logic NOT sẽ cho kết quả là false	!(A && B) là false.

❑ 4. Các phép toán trong PHP

Toán tử gán trong PHP

Dưới đây là những toán tử gán được hỗ trợ bởi ngôn ngữ PHP:

Toán tử	Miêu tả	Ví dụ
=	Toán tử gán đơn giản. Gán giá trị toán hạng bên phải cho toán hạng trái	C = A + B sẽ gán giá trị của A + B vào trong C
+=	Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái	C += A là tương đương với C = C + A
-=	Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái	C -= A là tương đương với C = C - A
*=	Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái	C *= A là tương đương với C = C * A
/=	Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng trái	C /= A là tương đương với C = C / A
%=	Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái	C %= A là tương đương với C = C % A

4. Các phép toán trong PHP

Toán tử điều kiện trong PHP

- Có nhiều hơn một toán tử được gọi là toán tử điều kiện. Đầu tiên, nó ước lượng một biểu thức là true hoặc false và sau đó thực thi một trong hai lệnh đã cho tùy thuộc vào kết quả của việc ước lượng. Toán tử điều kiện có cú pháp như sau:

```
<?php
        dieu_kien ? bieu_thuc_1 : bieu_thuc_2;
?>
```

```
<?php
$x = 10;
$y = 20;
echo ($x > $y) ? "x lớn hơn y." : "x nhỏ hơn y.";
?>
```

Kết quả:

```
x nhỏ hơn y.
```

4. Các phép toán trong PHP

Thứ tự ưu tiên của các toán tử trong PHP

Bảng dưới đây thể hiện thứ tự ưu tiên của các toán tử PHP:

Loại	Toán tử	Thứ tự ưu tiên
Unary	! ++ --	Phải sang trái
Tính nhân	* / %	Trái sang phải
Tính cộng	+ -	Trái sang phải
Quan hệ	< <= > >=	Trái sang phải
Tính bằng	== !=	Trái sang phải
Logic AND	&&	Trái sang phải
Logic OR	||	Trái sang phải
Điều kiện	?:	Phải sang trái
Gán	= += -= *= /= %=	Phải sang trái

KẾT THÚC