Php自学整理

2019.8.10

## 局部和全局作用域

在所有函数外部定义的变量，拥有全局作用域。除了函数外，全局变量可以被脚本中的任何部分访问，要在一个函数中访问一个全局变量，需要使用 global 关键字。

在 PHP 函数内部声明的变量是局部变量，仅能在函数内部访问：

## 实例

<?php   
$x=5; // 全局变量   
  
function myTest()   
{   
    $y=10; // 局部变量   
    echo "<p>测试函数内变量:<p>";   
    echo "变量 x 为: $x";   
    echo "<br>";   
    echo "变量 y 为: $y";   
}    
  
myTest();   
  
echo "<p>测试函数外变量:<p>";   
echo "变量 x 为: $x";   
echo "<br>";   
echo "变量 y 为: $y";   
?>

**[运行实例 »](https://www.runoob.com/try/showphp.php?filename=demo_var_local&type=php" \t "https://www.runoob.com/php/_blank)**

运行结果：

测试函数内变量:

变量 x 为:   
变量 y 为: 10

测试函数外变量:

变量 x 为: 5  
变量 y 为:

在以上实例中 myTest() 函数定义了 $x 和 $y 变量。 $x 变量在函数外声明，所以它是全局变量 ， $y 变量在函数内声明所以它是局部变量。

当我们调用myTest()函数并输出两个变量的值, 函数将会输出局部变量 $y 的值，但是不能输出 $x 的值，因为 $x 变量在函数外定义，无法在函数内使用，如果要在一个函数中访问一个全局变量，需要使用 global 关键字。

然后我们在myTest()函数外输出两个变量的值，函数将会输出全局变量 $x 的值，但是不能输出 $y 的值，因为 $y 变量在函数中定义，属于局部变量。

|  |  |
| --- | --- |
| **IMG_256** | 你可以在不同函数中使用相同的变量名称，因为这些函数内定义的变量名是局部变量，只作用于该函数内。 |

## PHP global 关键字

global 关键字用于函数内访问全局变量。

在函数内调用函数外定义的全局变量，我们需要在函数中的变量前加上 global 关键字：

## 实例

<?php

$x=5;

$y=10;

function myTest()

{

global $x,$y;

$y=$x+$y;

}

myTest();

echo $y; // 输出 15

?>

**[运行实例 »](https://www.runoob.com/try/showphp.php?filename=demo_var_global" \t "https://www.runoob.com/php/_blank)**

PHP 将所有全局变量存储在一个名为 $GLOBALS[*index*] 的数组中。 index 保存变量的名称。这个数组可以在函数内部访问，也可以直接用来更新全局变量。

上面的实例可以写成这样：

## 实例

<?php

$x=5;

$y=10;

function myTest()

{

$GLOBALS['y']=$GLOBALS['x']+$GLOBALS['y'];

}

myTest();

echo $y; //输出15

?>

## Static 作用域

当一个函数完成时，它的所有变量通常都会被删除。然而，有时候您希望某个局部变量不要被删除。

要做到这一点，请在您第一次声明变量时使用 **static** 关键字：

## 实例

<?php

function myTest()

{

static $x=0;

echo $x;

$x++;

echo PHP\_EOL; // 换行符

}

myTest();

myTest();

myTest(); ?>

**[运行实例 »](https://www.runoob.com/try/showphp.php?filename=demo_var_static" \t "https://www.runoob.com/php/_blank)**

运行结果 0 1 2

然后，每次调用该函数时，该变量将会保留着函数前一次被调用时的值。

**注释：**该变量仍然是函数的局部变量。

echo 和 print 区别:

* echo - 可以输出一个或多个字符串
* print - 只允许输出一个字符串，返回值总为 1

**提示：**echo 输出的速度比 print 快， echo 没有返回值，print有返回值1。

# PHP 5 常量

常量值被定义后，在脚本的其他任何地方都不能被改变。

## PHP 常量

常量是一个简单值的标识符。该值在脚本中不能改变。

一个常量由英文字母、下划线、和数字组成,但数字不能作为首字母出现。 (常量名不需要加 $ 修饰符)。

**注意：** 常量在整个脚本中都可以使用。

## 设置 PHP 常量

设置常量，使用 define() 函数，函数语法如下：

bool define ( string $name , mixed $value [, bool $case\_insensitive = false ] )

该函数有三个参数:

* **name：**必选参数，常量名称，即标志符。
* **value：**必选参数，常量的值。
* **case\_insensitive**：可选参数，如果设置为 TRUE，该常量则大小写不敏感。默认是大小写敏感的。

以下实例我们创建一个 **区分大小写的常量**, 常量值为 "欢迎访问 Runoob.com"：

## 实例

<?php

// 区分大小写的常量名

define("GREETING", "欢迎访问 Runoob.com");

echo GREETING; // 输出 "欢迎访问 Runoob.com"

echo '<br>';

echo greeting; // 输出 "greeting"

?>

以下实例我们创建一个 **不区分大小写的常量**, 常量值为 "欢迎访问 Runoob.com"：

## 实例

<?php

// 不区分大小写的常量名

define("GREETING", "欢迎访问 Runoob.com", true);

echo greeting; // 输出 "欢迎访问 Runoob.com"

?>

## 常量是全局的

常量在定义后，默认是全局变量，可以在整个运行的脚本的任何地方使用。

以下实例演示了在函数内使用常量，即便常量定义在函数外也可以正常使用常量。

## 实例

<?php

define("GREETING", "欢迎访问 Runoob.com");

function myTest() { echo GREETING; }

myTest(); // 输出 "欢迎访问 Runoob.com"

?>

## PHP strpos() 函数

strpos() 函数用于在字符串内查找一个字符或一段指定的文本。

如果在字符串中找到匹配，该函数会返回第一个匹配的字符位置。如果未找到匹配，则返回 FALSE。

下面的实例在字符串 "Hello world!" 中查找文本 "world"：

## 实例

<?php   
echo strpos("Hello world!","world");   
?>

上面的代码将输出：6

**提示：**在上面的实例中，字符串 "world" 的位置是 6。之所以是 6 而不是 7 的原因是，字符串中第一个字符的位置是 0，而不是 1。

## 三元运算符

另一个条件运算符是"?:"（或三元）运算符 。

### **语法格式**

(expr1) ? (expr2) : (expr3)

对 expr1 求值为 TRUE 时的值为 expr2，在 expr1 求值为 FALSE 时的值为 expr3。

自 PHP 5.3 起，可以省略三元运算符中间那部分。表达式 expr1 ?: expr3 在 expr1 求值为 TRUE 时返回 expr1，否则返回 expr3。

### **实例**

以下实例中通过判断 $\_GET 请求中含有 user 值，如果有返回 $\_GET['user']，否则返回 nobody：

## 实例

<?php

$test = '菜鸟教程';

// 普通写法

$username = isset($test) ? $test : 'nobody';

echo $username, PHP\_EOL;

// PHP 5.3+ 版本写法

$username = $test ?: 'nobody';

echo $username, PHP\_EOL;

?>

菜鸟教程菜鸟教程

**注意：**PHP\_EOL 是一个换行符，兼容更大平台。

在 PHP7+ 版本多了一个 NULL 合并运算符 **??**，实例如下：

## 实例

<?php

// 如果 $\_GET['user'] 不存在返回 'nobody'，否则返回 $\_GET['user'] 的值

$username = $\_GET['user'] ?? 'nobody';

// 类似的三元运算符

$username = isset($\_GET['user']) ? $\_GET['user'] : 'nobody';

?>

## 组合比较符(PHP7+)

PHP7+ 支持组合比较符（combined comparison operator）也称之为太空船操作符，符号为 **<=>**。组合比较运算符可以轻松实现两个变量的比较，当然不仅限于数值类数据的比较。

语法格式如下：

$c = $a <=> $b;

解析如下：

* 如果**$a > $b**, 则 **$c** 的值为 **1**。
* 如果 **$a == $b**, 则 **$c** 的值为 **0**。
* 如果 **$a < $b**, 则 **$c** 的值为 **-1**。

实例如下：

## 实例

<?php

// 整型

echo 1 <=> 1; // 0

echo 1 <=> 2; // -1

echo 2 <=> 1; // 1 // 浮点型

echo 1.5 <=> 1.5; // 0

echo 1.5 <=> 2.5; // -1

echo 2.5 <=> 1.5; // 1 // 字符串

echo "a" <=> "a"; // 0

echo "a" <=> "b"; // -1

echo "b" <=> "a"; // 1

?>

