* PHP 能够生成动态页面内容
* PHP 能够创建、打开、读取、写入、删除以及关闭服务器上的文件
* PHP 能够接收表单数据
* PHP 能够发送并取回 cookies
* PHP 能够添加、删除、修改数据库中的数据
* PHP 能够限制用户访问网站中的某些页面
* PHP 能够对数据进行加密
* PHP 运行于各种平台（Windows, Linux, Unix, Mac OS X 等等）
* PHP 兼容几乎所有服务器（Apache, IIS 等等）
* PHP 支持多种数据库
* PHP 是免费的。请从官方 PHP 资源下载：[www.php.net](http://www.php.net/" \t "_blank)
* PHP 易于学习，并可高效地运行在服务器端

如需开始使用 PHP，您可以：

* 使用支持 PHP 和 MySQL 的 web 主机
* 在您的 PC 上安装 web 服务器（Apache, IIS 等等），然后安装 PHP 和 MySQL。
* **PHP 支持三种注释：**

<?php

// 这是单行注释

# 这也是单行注释

/\*

这是多行注释块

它横跨了

多行

\*/

?>

所有用户定义的函数、类和关键词（例如 if、else、echo 等等）都对大小写不敏感。

所有变量都对大小写敏感。

变量以 $ 符号开头，其后是变量的名称

PHP 有三种不同的变量作用域：

* local（局部）
* global（全局）
* static（静态）

## Local 和 Global 作用域

函数之外声明的变量拥有 Global 作用域，只能在函数以外进行访问。

函数内部声明的变量拥有 LOCAL 作用域，只能在函数内部进行访问。

## PHP global 关键词

global 关键词用于访问函数内的全局变量。

要做到这一点，请在（函数内部）变量前面使用 global 关键词：

### 实例

<?php

$x=5;

$y=10;

function myTest() {

global $x,$y;

$y=$x+$y;

}

myTest();

echo $y; // 输出 15

?>

PHP 同时在名为 $GLOBALS[index] 的数组中存储了所有的全局变量。下标存有变量名。这个数组在函数内也可以访问，并能够用于直接更新全局变量。

<?php

$x=5;

$y=10;

function myTest() {

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

}

myTest();

echo $y; // 输出 15

?>

可以用三种格式规定整数：十进制、十六进制（前缀是 0x）或八进制（前缀是 0）

在 PHP 中，必须明确地声明对象。

首先我们必须声明对象的类。对此，我们使用 class 关键词。类是包含属性和方法的结构。

## PHP strlen() 函数

strlen() 函数返回字符串的长度，以字符计。

下例返回字符串 "Hello world!" 的长度：

<?php

echo strlen("Hello world!");

?>

内置函数很少有对象函数？

如需设置常量，请使用 define() 函数 - 它使用三个参数：

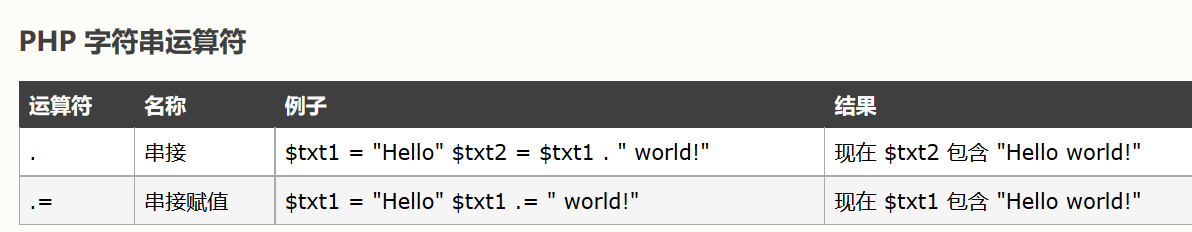
1. 首个参数定义常量的名称
2. 第二个参数定义常量的值
3. 可选的第三个参数规定常量名是否对大小写不敏感。默认是 false。

<?php

define("GREETING", "Welcome to W3School.com.cn!");

echo GREETING;

?>



## PHP echo 和 print 语句

echo 和 print 之间的差异：

* echo - 能够输出一个以上的字符串
* print - 只能输出一个字符串，并始终返回 1

提示：echo 比 print 稍快，因为它不返回任何值。

|  |  |  |  |
| --- | --- | --- | --- |
| === | 全等（完全相同） | $x === $y | 如果 $x 等于 $y，且它们类型相同，则返回 true。 |

|  |  |  |  |
| --- | --- | --- | --- |
| === | 全等 | $x === $y | 如果 $x 和 $y 拥有相同的键/值对，且顺序相同类型相同，则返回 true。 |

## PHP foreach 循环

foreach 循环只适用于数组，并用于遍历数组中的每个键/值对。

foreach ($array as $value) {

code to be executed;

}

在 PHP 中，有三种数组类型：

* *索引数组* - 带有数字索引的数组
* *关联数组* - 带有指定键的数组
* *多维数组* - 包含一个或多个数组的数组

有两种创建索引数组的方法：

索引是自动分配的（索引从 0 开始）：

$cars=array("Volvo","BMW","SAAB");

或者也可以手动分配索引：

$cars[0]="Volvo";

$cars[1]="BMW";

$cars[2]="SAAB";

## PHP 关联数组

关联数组是索引不是整数的数组。

关联数组是使用您分配给数组的指定键的数组。

有两种创建关联数组的方法：

$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");

或者：

$age['Peter']="35";

$age['Ben']="37";

$age['Joe']="43";

## 遍历关联数组

如需遍历并输出关联数组的所有值，您可以使用 foreach 循环，就像这样：

$x是键，$x\_value是键对应的值

<?php

$age=array("Bill"=>"35","Steve"=>"37","Peter"=>"43");

foreach($age as $x=>$x\_value) {

echo "Key=" . $x . ", Value=" . $x\_value;

echo "<br>";

}

?>

**超全局变量 在 PHP 4.1.0 中引入，是在全部作用域中始终可用的内置变量。**

这些超全局变量是：

* $GLOBALS
* $\_SERVER
* $\_REQUEST
* $\_POST
* $\_GET
* $\_FILES
* $\_ENV
* $\_COOKIE
* $\_SESSION

## PHP 表单验证

htmlspecialchars() 函数把特殊字符转换为 HTML 实体。这意味着 < 和 > 之类的 HTML 字符会被替换为 &lt; 和 &gt; 。

（通过 PHP trim() 函数）去除用户输入数据中不必要的字符（多余的空格、制表符、换行）

（通过 PHP stripslashes() 函数）删除用户输入数据中的反斜杠（\）

接下来我们创建一个检查函数（相比一遍遍地写代码，这样效率更好）。

我们把函数命名为 test\_input()。

现在，我们能够通过 test\_input() 函数检查每个 $\_POST 变量，脚本是这样的：

<?php

// 定义变量并设置为空值

$name = $email = $gender = $comment = $website = "";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$name = test\_input($\_POST["name"]);

$email = test\_input($\_POST["email"]);

$website = test\_input($\_POST["website"]);

$comment = test\_input($\_POST["comment"]);

$gender = test\_input($\_POST["gender"]);

}

function test\_input($data) {

$data = trim($data);

$data = stripslashes($data);

$data = htmlspecialchars($data);

return $data;

}

?>

**必填输入字段，并创建需要时所用的错误消息。**

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

if (empty($\_POST["name"])) {

$nameErr = "Name is required";

} else {

$name = test\_input($\_POST["name"]);

}

}

接下来是验证输入数据，即“Name 字段是否只包含字母和空格？”，以及“E-mail 字段是否包含有效的电子邮件地址语法？”，并且如果填写了 Website 字段，“这个字段是否包含了有效的 URL？”。

$name = test\_input($\_POST["name"]);

if (!preg\_match("/^[a-zA-Z ]\*$/",$name)) {

$nameErr = "只允许字母和空格！";

}

**本节展示如何在用户提交表单后保留输入字段中的值**

Name: <input type="text" name="name" value="<?php echo $name;?>">

**服务器端包含 (SSI) 用于创建可在多个页面重复使用的函数、页眉、页脚或元素。**

include 和 require 语句是相同的，除了错误处理方面：

require 会生成致命错误（E\_COMPILE\_ERROR）并停止脚本

include 只生成警告（E\_WARNING），并且脚本会继续

假设我们有一个名为 "vars.php" 的文件，其中定义了一些变量：

<?php

$color='银色的';

$car='奔驰轿车';

?>

然后，如果我们引用这个 "vars.php" 文件，就可以在调用文件中使用这些变量：

<html>

<body>

<h1>欢迎访问我的首页！</h1>

<?php

include 'vars.php';

echo "我有一辆" . $color . $car "。";

?>

</body>

</html>