PHP 简介

PHP 是服务器端脚本语言。

您应当具备的基础知识

在继续学习之前，您需要对以下知识有基本的了解：

- HTML
- CSS

如果您希望首先学习这些项目，请在我们的 首页 访问这些教程。

PHP 是什么?

- PHP（全称：PHP：Hypertext Preprocessor，即"PHP：超文本预处理器"）是一种通用开源脚本语言。
- PHP 脚本在服务器上执行。
- PHP 可免费下载使用。

PHP 对初学者而言简单易学。

PHP 也为专业的程序员提供了许多先进的功能。

PHP 文件是什么?

- PHP 文件可包含文本、HTML、JavaScript代码和 PHP 代码
- PHP 代码在服务器上执行，结果以纯 HTML 形式返回给浏览器
- PHP 文件的默认文件扩展名是 ".php"

PHP 能做什么?

- PHP 可以生成动态页面内容
- PHP 可以创建、打开、读取、写入、关闭服务器上的文件
- PHP 可以收集表单数据
- PHP 可以发送和接收 cookies
- PHP 可以添加、删除、修改您的数据库中的数据
- PHP 可以限制用户访问您的网站上的一些页面
- PHP 可以加密数据

通过 PHP，您不再限于输出 HTML。您可以输出图像、PDF 文件，甚至 Flash 电影。您还可以输出任意的文本，比如 XHTML 和 XML。

为什么使用 PHP?

- PHP 可在不同的平台上运行（Windows、Linux、Unix、Mac OS X 等）
- PHP 与目前几乎所有的正在被使用的服务器相兼容（Apache、IIS 等）
- PHP 提供了广泛的数据库支持
- PHP 是免费的。可从官方的 PHP 资源下载它：www.php.net
- PHP 易于学习。并可高效地运行在服务器端

PHP 安装

您需要做什么?

为了开始使用 PHP，您可以：

- 找一个支持 PHP 和 MySQL 的 Web 主机
- 在您自己的 PC 机上安装 Web 服务器，然后安装 PHP 和 MySQL

使用支持 PHP 的 Web 主机

如果您的服务器支持 PHP，那么您不需要做任何事情。

只要在您的 web 目录中创建 .php 文件即可，服务器将自动为您解析这些文件。

您不需要编译任何软件，或安装额外的工具。

由于 PHP 是免费的，大多数的 Web 主机都提供对 PHP 的支持。

在您自己的 PC 机上建立 PHP

然而，如果您的服务器不支持 PHP，您必须：

- 安装 Web 服务器
- 安装 PHP
- 安装数据库，比如 MySQL

官方 PHP 网站（PHP.net）有 PHP 的安装说明：http://php.net/manual/en/install.php

PHP 服务器组件

对于初学者建议使用集成的服务器组件，它已经包含了 PHP、Apache、Mysql 等服务,免去了开发人员将时间花费在繁琐的配置环境过程。

WampServer

Window 系统可以使用 WampServer，下载地址：http://www.wampserver.com/，支持32位和64位系统，根据自己的系统选择版本。

WampServer 安装也简单，你只需要一直点击 **"Next"** 就可以完成安装了。

XAMPP

XAMPP 支持 Mac OS 和 Window 系统，下载地址：https://www.apachefriends.org/zh_cn/index.html。

IDE (Integrated Development Environment,集成开发环境)

Eclipse for PHP（免费）

Eclipse 是一个开放源代码的、基于Java的可扩展开发平台(如果未安装JDK，则需要先 下载 JDK 安装)。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 附带了一个标准的插件集，包括Java开发工具（Java Development Kit，JDK）。

支持 Window、Linux 和 Mac OS 平台。

Eclipse for PHP 官方下载地址：http://www.eclipse.org/downloads/packages/eclipse-php-developers/heliosr

PhpStorm（收费）

PhpStorm是一个轻量级且便捷的PHP IDE，其旨在提供用户效率，可深刻理解用户的编码，提供智能代码补全，快速导航以及即时错误检查。

PhpStorm 非常适合于PHP开发人员及前端工程师。提供诸于：智能HTML/CSS/JavaScript/PHP编辑、代码质量分析、版本控制集成（SVN、GIT）、调试和测试等功能。

支持 Window、Linux 和 Mac OS 平台。

PhpStorm 官方下载地址：http://www.jetbrains.com/phpstorm/download/

PHP 语法

PHP 脚本在服务器上执行，然后将纯 HTML 结果发送回浏览器。

基本的 PHP 语法

PHP 脚本可以放在文档中的任何位置。

PHP 脚本以 **<?php** 开始，以 **?>** 结束：

```
<?php
// PHP 代码
?>
```

PHP 文件的默认文件扩展名是 ".php"。

PHP 文件通常包含 HTML 标签和一些 PHP 脚本代码。

下面，我们提供了一个简单的 PHP 文件实例，它可以向浏览器输出文本 "Hello World!"：

实例

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

运行实例 »

PHP 中的每个代码行都必须以分号结束。分号是一种分隔符，用于把指令集区分开来。

通过 PHP，有两种在浏览器输出文本的基础指令：**echo** 和 **print**。

PHP 中的注释

实例

```
<!DOCTYPE html>
<html>
<body>

<?php
// 这是 PHP 单行注释

/*
这是
PHP 多行
注释
*/
?>

</body>
</html>
```

运行实例 »

PHP 变量

变量是用于存储信息的"容器"：

实例

```
<?php $x=5; $y=6; $z=$x+$y; echo $z; ?>
```

运行实例 »

与代数类似

```
x=5
y=6
z=x+y
```

在代数中，我们使用字母（如 x），并给它赋值（如 5）。

从上面的表达式 z=x+y，我们可以计算出 z 的值为 11。

在 PHP 中，这些字母被称为变量。

变量是用于存储数据的容器。

PHP 变量

与代数类似，可以给 PHP 变量赋予某个值（x=5）或者表达式（z=x+y）。

变量可以是很短的名称（如 x 和 y）或者更具描述性的名称（如 age、carname、totalvolume）。

PHP 变量规则：

- 变量以 $ 符号开始，后面跟着变量的名称
- 变量名必须以字母或者下划线字符开始
- 变量名只能包含字母数字字符以及下划线（A-z、0-9 和 _ ）
- 变量名不能包含空格
- 变量名是区分大小写的（$y 和 $Y 是两个不同的变量）

PHP 语句和 PHP 变量都是区分大小写的。

创建（声明）PHP 变量

PHP 没有声明变量的命令。

变量在您第一次赋值给它的时候被创建：

实例

```
<?php $txt="Hello world!"; $x=5; $y=10.5; ?>
```

运行实例 »

在上面的语句执行中，变量 **txt** 将保存值 **Hello world!**，且变量 **x** 将保存值 **5**。

注释：当您赋一个文本值给变量时，请在文本值两侧加上引号。

PHP 是一门弱类型语言

在上面的实例中，我们注意到，不必向 PHP 声明该变量的数据类型。

PHP 会根据变量的值，自动把变量转换为正确的数据类型。

在强类型的编程语言中，我们必须在使用变量前先声明（定义）变量的类型和名称。

PHP 变量作用域

变量的作用域是脚本中变量可被引用/使用的部分。

PHP 有四种不同的变量作用域：

- local
- global
- static
- parameter

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 »

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，函数将会输出全局部变量 $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。

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 »

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; ?>
```

运行实例 »

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++; } myTest(); myTest(); myTest(); ?>
```

运行实例 »

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。

参数作用域

参数是通过调用代码将值传递给函数的局部变量。

参数是在参数列表中声明的，作为函数声明的一部分：

实例

```
<?php function myTest($x) { echo $x; } myTest(5); ?>
```

我们将在 PHP 函数 章节对它做更详细的讨论。

PHP 字符串变量

字符串变量用于存储并处理文本。

PHP 中的字符串变量

字符串变量用于包含有字符的值。

在创建字符串之后，我们就可以对它进行操作了。您可以直接在函数中使用字符串，或者把它存储在变量中。

在下面的实例中，我们创建一个名为 txt 的字符串变量，并赋值为 "Hello world!" 。然后我们输出 txt 变量的值：

实例

```
<?php
$txt="Hello world!";
echo $txt;
?>
```

运行实例 »

注释：当您赋一个文本值给变量时，请记得给文本值加上单引号或者双引号。

现在，让我们来看看一些常用的操作字符串的函数和运算符。

PHP 并置运算符

在 PHP 中，只有一个字符串运算符。

并置运算符 (.) 用于把两个字符串值连接起来。

下面的实例演示了如何将两个字符串变量连接在一起：

实例

```
<?php
$txt1="Hello world!";
$txt2="What a nice day!";
echo $txt1 . " " . $txt2;
?>
```

上面的代码将输出：Hello world! What a nice day!

提示：在上面的代码中，我们已经使用了两次并置运算符。这是由于我们需要在两个字符串之间插入一个空格。

PHP strlen() 函数

有时知道字符串值的长度是很有用的。

strlen() 函数返回字符串的长度（字符数）。

下面的实例返回字符串 "Hello world!" 的长度：

实例

```
<?php
echo strlen("Hello world!");
?>
```

运行实例 »

上面的代码将输出：12

提示：strlen() 常常用在循环和其他函数中，因为那时确定字符串何时结束是很重要的。（例如，在循环中，我们需要在字符串中的最后一个字符之后结束循环。）

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");
?>
```

运行实例 »

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。

完整的 PHP String 参考手册

如需查看所有字符串函数的完整参考手册，请访问我们的 PHP String 参考手册。

该参考手册提供了每个函数的简要描述和应用实例！

PHP 运算符

本章节我们将讨论 PHP 中不同运算符的应用。

在 PHP 中，赋值运算符 = 用于给变量赋值。

在 PHP 中，算术运算符 + 用于把值加在一起。

PHP 算术运算符

运算符	名称	描述	实例	结果
x + y	加	x 和 y 的和	2 + 2	4
x - y	减	x 和 y 的差	5 - 2	3
x * y	乘	x 和 y 的积	5 * 2	10
x / y	除	x 和 y 的商	15 / 5 5 % 2	3 1
x % y	模（除法的余数）	x 除以 y 的余数	10 % 8 10 % 2	2 0
- x	取反	x 取反	-2	-2
a . b	并置	连接两个字符串	"Hi" . "Ha"	HiHa

以下实例演示了使用不同算术运算符得到的不同结果：

实例

```
<?php $x=10; $y=6; echo ($x + $y); // 输出16 echo '<br>'; // 换行 echo ($x - $y); // 输出4 echo '<br>'; // 换行 echo ($x * $y); // 输出60 echo '<br>'; // 换行 echo ($x / $y); // 输出1.6666666666667 echo '<br>'; // 换行 echo ($x % $y); // 输出4 echo '<br>'; // 换行 echo -$x; ?>
```

尝试一下 »

PHP7+ 版本新增整除运算符 **intdiv()**,使用实例：

实例

```
<?php var_dump(intdiv(10, 3)); ?>
```

以上实例会输出：

```
int(3)
```

PHP 赋值运算符

在 PHP 中，基本的赋值运算符是 "="。它意味着左操作数被设置为右侧表达式的值。也就是说，"$x = 5" 的值是 5。

运算符	等同于	描述
x = y	x = y	左操作数被设置为右侧表达式的值
x += y	x = x + y	加
x -= y	x = x - y	减
x *= y	x = x * y	乘
x /= y	x = x / y	除
x %= y	x = x % y	模（除法的余数）
a .= b	a = a . b	连接两个字符串

以下实例演示了使用不同赋值运算符得到的不同结果：

实例

```
<?php $x=10; echo $x; // 输出10 $y=20; $y += 100; echo $y; // 输出120 $z=50; $z -= 25; echo $z; // 输出25 $i=5; $i *= 6; echo $i; // 输出30 $j=10; $j /= 5; echo $j; // 输出2 $k=15; $k %= 4; echo $k; // 输出3 ?>
```

尝试一下 »

以下实例演示了使用不同字符串运算符得到的不同结果：

实例

```
<?php $a = "Hello"; $b = $a . " world!"; echo $b; // 输出Hello world! $x="Hello"; $x .= " world!"; echo $x; // 输出Hello world! ?>
```

尝试一下 »

PHP 递增/递减运算符

运算符	名称	描述
++ x	预递增	x 加 1，然后返回 x
x ++	后递增	返回 x，然后 x 加 1
-- x	预递减	x 减 1，然后返回 x
x --	后递减	返回 x，然后 x 减 1

以下实例演示了使用递增/递减运算符得到的结果：

实例

```
<?php $x=10; echo ++$x; // 输出11 $y=10; echo $y++; // 输出10 $z=5; echo --$z; // 输出4 $i=5; echo $i--; // 输出5 ?>
```

尝试一下 »

PHP 比较运算符

比较操作符可以让您比较两个值：

运算符	名称	描述	实例
x == y	等于	如果 x 等于 y，则返回 true	5==8 返回 false
x === y	恒等于	如果 x 等于 y，且它们类型相同，则返回 true	5===“5” 返回 false
x != y	不等于	如果 x 不等于 y，则返回 true	5!=8 返回 true
x <> y	不等于	如果 x 不等于 y，则返回 true	5<>8 返回 true
x !== y	不恒等于	如果 x 不等于 y，或它们类型不相同，则返回 true	5!==“5” 返回 true
x > y	大于	如果 x 大于 y，则返回 true	5>8 返回 false
x < y	小于	如果 x 小于 y，则返回 true	5<8 返回 true
x >= y	大于等于	如果 x 大于或者等于 y，则返回 true	5>=8 返回 false
x <= y	小于等于	如果 x 小于或者等于 y，则返回 true	5<=8 返回 true

以下实例演示了使用一些比较运算符得到的不同结果：

实例

```
<?php $x=100; $y="100"; var_dump($x == $y); echo "<br>"; var_dump($x === $y); echo "<br>"; var_dump($x != $y); echo "<br>"; var_dump($x !== $y); echo "<br>"; $a=50; $b=90; var_dump($a > $b); echo "<br>"; var_dump($a < $b); ?>
```

尝试一下 »

PHP 逻辑运算符

运算符	名称	描述	实例
x and y	与	如果 x 和 y 都为 true，则返回 true	x=6 y=3 (x < 10 and y > 1) 返回 true
x or y	或	如果 x 和 y 至少有一个为 true，则返回 true	x=6 y=3 (x==6 or y==5) 返回 true
x xor y	异或	如果 x 和 y 有且仅有一个为 true，则返回 true	x=6 y=3 (x==6 xor y==3) 返回 false
x && y	与	如果 x 和 y 都为 true，则返回 true	x=6 y=3 (x < 10 && y > 1) 返回 true
x || y	或	如果 x 和 y 至少有一个为 true，则返回 true	x=6 y=3 (x==5 || y==5) 返回 false

! x	非	如果 x 不为 true，则返回 true	x=6 y=3 !(x==y) 返回 true

PHP 数组运算符

运算符	名称	描述
x + y	集合	x 和 y 的集合
x == y	相等	如果 x 和 y 具有相同的键/值对，则返回 true
x === y	恒等	如果 x 和 y 具有相同的键/值对，且顺序相同类型相同，则返回 true
x != y	不相等	如果 x 不等于 y，则返回 true
x <> y	不相等	如果 x 不等于 y，则返回 true
x !== y	不恒等	如果 x 不等于 y，则返回 true

以下实例演示了使用一些数组运算符得到的不同结果:

实例

```
<?php $x = array("a" => "red", "b" => "green"); $y = array("c" => "blue", "d" => "yellow"); $z = $x + $y; // $x 和 $y 数组合并 var_dump($z); var_dump($x == $y); var_dump($x === $y); var_dump($x != $y); var_dump($x <> $y); var_dump($x !== $y); ?>
```

尝试一下 »

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，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 ?>
```

PHP If...Else è¯å¥

æ¡ä»¶è¯å¥ç¨äº æ ¹æ®ä¸åæ¡ä»¶æ§è¡ä¸åå¨ä½ã

PHP æ¡ä»¶è¯å¥

å½æ¨ç¼åä»£ç æ¶ï¼æ¨å¸¸å¸¸éè¦ä¸ºä¸åçå¤ææ§è¡ä¸åçå¨ä½ãæ¨å¯ä»¥å¨ä»£ç ä¸ä½¿ç¨æ¡ä»¶è¯å¥æ¥å®æè¿ä¸ªä»»å¡ã

å¨ PHP ä¸ï¼æä»¬å¯ä»¥ä½¿ç¨ä»¥ä¸æ¡ä»¶è¯å¥ï¼

- **if è¯å¥** - å¨æ¡ä»¶æç«æ¶æ§è¡ä»£ç
- **if...else è¯å¥** - å¨æ¡ä»¶æç«æ¶æ§è¡ä¸åä»£ç ï¼æ¡ä»¶ä¸æç«æ¶æ§è¡å¦ä¸åä»£ç
- **if...elseif....else è¯å¥** - å¨è¥å¹²æ¡ä»¶ä¹ä¸æç«æ¶æ§è¡ä¸ä¸ªä»£ç å
- **switch è¯å¥** - å¨è¥å¹²æ¡ä»¶ä¹ä¸æç«æ¶æ§è¡ä¸ä¸ªä»£ç å

PHP - if è¯å¥

if è¯å¥ç¨äº**ä»å½æå®æ¡ä»¶æç«æ¶æ§è¡ä»£ç** ã

è¯æ³

```
if (æ¡ä»¶)
{
 æ¡ä»¶æç«æ¶è¦æ§è¡çä»£ç ;
}
```

å¦æå½åæ¶é´å°äº 20ï¼ä¸é¢çå®ä¾å°è¾åº "Have a good day!"ï¼

å®ä¾

```
<?php $t=date("H"); if ($t<"20") { echo "Have a good day!"; } ?>
```

è¿è¡å®ä¾ Â»

PHP - if...else è¯å¥

å¨æ¡ä»¶æç«æ¶æ§è¡ä¸åä»£ç ï¼æ¡ä»¶ä¸æç«æ¶æ§è¡å¦ä¸åä»£ç ï¼è¯·ä½¿ç¨ if....else è¯å¥ã

è¯æ³

if (*æ¡ä»¶*)
{
æ¡ä»¶æç«æ¶æ§è¡çä»£ç ;
}
else
{
æ¡ä»¶ä¸æç«æ¶æ§è¡çä»£ç ;
}

å¦æå½åæ¶é´å°äº 20ï¼ä¸é¢çå®ä¾å°è¾åº "Have a good day!"ï¼å¦åè¾åº "Have a good night!"ï¼

å®ä¾

```
<?php $t=date("H"); if ($t<"20") { echo "Have a good day!"; } else { echo "Have a good night!"; } ?>
```

è¿è¡å®ä¾ Â»

PHP - if...elseif....else è¯å¥

å¨è¥å¹²æ¡ä»¶ä¹ä¸æç«æ¶æ§è¡ä¸ä¸ªä»£ç åï¼è¯·ä½¿ç¨ if....elseif...else è¯å¥.

è¯æ³

if (*æ¡ä»¶*)
{
if æ¡ä»¶æç«æ¶æ§è¡çä»£ç ;
}
elseif (*æ¡ä»¶*)
{
elseif æ¡ä»¶æç«æ¶æ§è¡çä»£ç ;
}
else
{
æ¡ä»¶ä¸æç«æ¶æ§è¡çä»£ç ;
}

å¦æå½åæ¶é´å°äº 10ï¼ä¸é¢çå®ä¾å°è¾åº "Have a good morning!"ï¼å¦æå½åæ¶é´ä¸å°äº 10 ä¸å°äº 20ï¼åè¾åº "Have a good day!"ï¼å¦åè¾åº "Have a good night!"ï¼

å®ä¾

```
<?php $t=date("H"); if ($t<"10") { echo "Have a good morning!"; } elseif ($t<"20") { echo "Have a good day!"; } else { echo "Have a good night!"; } ?>
```

è¿è¡å®ä¾ Â»

PHP - switch è¯å¥

switch è¯å¥å°å¨ä¸ä¸ç« è¿è¡è®²è§£ã

PHP Switch 语句

---

switch 语句用于根据多个不同条件执行不同动作。

PHP Switch 语句

如果您希望**有选择地执行若干代码块之一**，请使用 switch 语句。

语法

```
<?php switch (n) { case label1: 如果 n=label1，此处代码将执行; break; case label2: 如果 n=label2，此处代码将执行; break; default: 如果 n 既不等于 label1 也不等于 label2，此处代码将执行; } ?>
```

工作原理：首先对一个简单的表达式 *n*（通常是变量）进行一次计算。将表达式的值与结构中每个 case 的值进行比较。如果存在匹配，则执行与 case 关联的代码。代码执行后，使用 **break** 来阻止代码跳入下一个 case 中继续执行。**default** 语句用于不存在匹配（即没有 case 为真）时执行。

实例

```
<?php $favcolor="red"; switch ($favcolor) { case "red": echo "你喜欢的颜色是红色!"; break; case "blue": echo "你喜欢的颜色是蓝色!"; break; case "green": echo "你喜欢的颜色是绿色!"; break; default: echo "你喜欢的颜色不是 红, 蓝, 或绿色!"; } ?>
```

运行实例 »

PHP 数组

数组能够在单个变量中存储多个值:

实例

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

运行实例 »

数组是什么?

数组是一个能在单个变量中存储多个值的特殊变量。

如果您有一个项目清单（例如：车名字的清单），将其存储到单个变量中，如下所示：

```
$cars1="Volvo";
$cars2="BMW";
$cars3="Toyota";
```

然而，如果您想要遍历数组并找出特定的一个呢？如果数组的项不只 3 个而是 300 个呢？

解决办法是创建一个数组！

数组可以在单个变量中存储多个值，并且您可以根据键访问其中的值。

在 PHP 中创建数组

在 PHP 中，array() 函数用于创建数组：

```
array();
```

在 PHP 中，有三种类型的数组：

- **数值数组** - 带有数字 ID 键的数组
- **关联数组** - 带有指定的键的数组，每个键关联一个值
- **多维数组** - 包含一个或多个数组的数组

PHP 数值数组

这里有两种创建数值数组的方法：

自动分配 ID 键（ID 键总是从 0 开始）：

```
$cars=array("Volvo","BMW","Toyota");
```

人工分配 ID 键：

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

下面的实例创建一个名为 $cars 的数值数组，并给数组分配三个元素,然后打印一段包含数组值的文本：

实例

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

运行实例 »

获取数组的长度 - count() 函数

count() 函数用于返回数组的长度（元素的数量）：

实例

```
<?php $cars=array("Volvo","BMW","Toyota"); echo count($cars); ?>
```

运行实例 »

遍历数值数组

遍历并打印数值数组中的所有值。您可以使用 for 循环，如下所示：

实例

```
<?php $cars=array("Volvo","BMW","Toyota"); $arrlength=count($cars); for($x=0;$x<$arrlength;$x++) { echo $cars[$x]; echo "<br>"; } ?>
```

运行实例 »

PHP 关联数组

关联数组是使用您分配给数组的指定的键的数组。

这里有两种创建关联数组的方法：

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

or:

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

随后可以在脚本中使用指定的键：

实例

```
<?php $age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43"); echo "Peter is " . $age['Peter'] . " years old."; ?>
```

运行实例 »

遍历关联数组

遍历并打印关联数组中的所有值。您可以使用 foreach 循环，如下所示：

实例

```
<?php $age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43"); foreach($age as $x=>$x_value) { echo "Key=" . $x . ", Value=" . $x_value; echo "<br>"; } ?>
```

运行实例 »

多维数组

多维数组 将在 PHP 高级教程部分做详细介绍。

完整的 PHP Array 参考手册

如需查看所有数组函数的完整参考手册，请访问我们的 PHP Array 参考手册。

该参考手册提供了每个函数的简要描述和应用实例！

PHP 数组排序

数组中的元素可以按字母或数字顺序进行降序或升序排列。

PHP - 数组排序函数

在本章中，我们将一一介绍下列 PHP 数组排序函数：

- sort() - 对数组进行升序排列
- rsort() - 对数组进行降序排列
- asort() - 根据关联数组的值，对数组进行升序排列
- ksort() - 根据关联数组的键，对数组进行升序排列
- arsort() - 根据关联数组的值，对数组进行降序排列
- krsort() - 根据关联数组的键，对数组进行降序排列

---

sort() - 对数组进行升序排列

下面的实例将 $cars 数组中的元素按照字母升序排列：

实例

```
<?php
$cars=array("Volvo","BMW","Toyota");
sort($cars);
?>
```

运行实例 »

下面的实例将 $numbers 数组中的元素按照数字升序排列：

实例

```
<?php
$numbers=array(4,6,2,22,11);
sort($numbers);
?>
```

运行实例 »

---

rsort() - 对数组进行降序排列

下面的实例将 $cars 数组中的元素按照字母降序排列：

实例

```
<?php
$cars=array("Volvo","BMW","Toyota");
rsort($cars);
?>
```

运行实例 »

下面的实例将 $numbers 数组中的元素按照数字降序排列：

实例

```
<?php
$numbers=array(4,6,2,22,11);
rsort($numbers);
?>
```

运行实例 »

---

asort() - 根据数组的值，对数组进行升序排列

下面的实例根据数组的值，对关联数组进行升序排列：

实例

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
asort($age);
```

```
?>
```

运行实例 »

---

ksort() - 根据数组的键，对数组进行升序排列

下面的实例根据数组的键，对关联数组进行升序排列：

实例

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
ksort($age);
?>
```

运行实例 »

---

arsort() - 根据数组的值，对数组进行降序排列

下面的实例根据数组的值，对关联数组进行降序排列：

实例

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
arsort($age);
?>
```

运行实例 »

---

krsort() - 根据数组的键，对数组进行降序排列

下面的实例根据数组的键，对关联数组进行降序排列：

实例

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
krsort($age);
?>
```

运行实例 »

完整的 PHP Array 参考手册

如需查看所有数组函数的完整参考手册，请访问我们的 PHP Array 参考手册。

该参考手册提供了每个函数的简要描述和应用实例！

PHP å¾ªç¯ - While å¾ªçŽ¯

å¾ªçŽ¯æ‰§è¡Œä»£ç æ®µæŒ‡å®šçš„æ¬¡æ•°ï¼Œæˆ–è€…å½“æŒ‡å®šçš„æ¡ä»¶ä¸ºçœŸæ—¶å¾ªçŽ¯æ‰§è¡Œä»£ç å—ã€‚

PHP å¾ªçŽ¯

å¯¹æ‰¹é‡æ•°æ®è¿›è¡Œç›¸åŒçš„æ“ä½œæ—¶ï¼Œä½¿ç”¨å¾ªçŽ¯æ˜¯éžå¸¸æœ‰ç”¨çš„æ–¹æ³•ã€‚

åœ¨ PHP ä¸æä¾›äº†ä»¥ä¸‹å¾ªçŽ¯è¯å¥ï¼š

- **while** - 在指定条件为真时循环指定的代码块
- **do...while** - 先执行一次代码块，然后在指定条件为真时重复循环，
- **for** - 循环指定次数的代码块
- **foreach** - 根据数组中每个元素循环代码块

---

while 循环

while 循环将执行一个代码块，只要表达式为真。

语法

```
while (条件为真){
执行的代码;
}
```

实例

演示一个打印数字的简单 while 循环：

```
<html>
<body>

<?php
$i=1;
while($i<=5)
{
echo "The number is " . $i . "<br>";
$i++;
}
?>

</body>
</html>
```

输出：

```
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

---

do...while 语句

do...while 语句将执行一个代码块一次，然后只要表达式为真时重复循环。

语法

```
do
{
执行的代码;
}
while (条件为真);
```

实例

演示一个打印数字的简单 do...while 循环：

```
<html>
<body>

<?php
$i=1;
do
{
$i++;
echo "The number is " . $i . "<br>";
}
while ($i<=5);
?>

</body>
</html>
```

输出：

```
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
```

for 循环在 foreach 循环中进行了讨论。

PHP 循环 - For 循环

循环执行代码块指定的次数，或者当指定的条件为真时循环执行代码块。

---

for 循环

for 循环用于您预先知道脚本需要运行的次数的情况。

语法

```
for (初始值: 条件: 增量)
{
要执行的代码;
}
```

参数：

- *初始值*：主要是初始化一个变量值，用于设置一个计数器（但可以是任何在循环的开始被执行一次的代码）。
- *条件*：循环执行的限制条件。如果为 TRUE，则循环继续。如果为 FALSE，则循环结束。
- *增量*：主要用于递增计数器（但可以是任何在循环的结束被执行的代码）。

注释：上面的 *初始值* 和 *增量* 参数可为空，或者有多个表达式（用逗号分隔）。

实例

下面的实例定义一个初始值为 i=1 的循环。只要变量 *i* 小于或者等于 5，循环将继续运行。循环每运行一次，变量 *i* 就会递增 1：

```
<html>
<body>

<?php
for ($i=1; $i<=5; $i++)
{
echo "The number is " . $i . "<br>";
}
?>

</body>
</html>
```

输出：

```
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

---

foreach 循环

foreach 循环用于遍历数组。

Syntax

```
foreach ($array as $value)
{
要执行代码;
}
```

每进行一次循环，当前数组元素的值就会被赋值给 $value 变量（数组指针会逐一地移动），在进行下一次循环时，您将看到数组中的下一个值。

实例

下面的实例演示了一个输出给定数组的值的循环：

```
<html>
<body>

<?php
$x=array("one","two","three");
foreach ($x as $value)
{
echo $value . "<br>";
}
?>

</body>
</html>
```

输出：

```
one
two
three
```

PHP 函数

PHP 的真正成力源自于它的函数。

在 PHP 中，提供了超过 1000 个内建的函数。

PHP 内建函数

如需查看所有数组函数的完整参考手册和实例，请访问我们的 PHP 参考手册。

PHP 函数

在本章中，我们将为您讲解如何创建自己的函数。

如要在页面加载时执行脚本，您可以把它放到函数里。

函数是通过调用函数来执行的。

你可以在页面的任何位置调用函数。

创建 PHP 函数

函数是通过调用函数来执行的。

语法

```
function functionName()
{
要执行的代码;
}
```

PHP 函数准则：

- 函数的名称应该提示出它的功能
- 函数名称以字母或下划线开头（不能以数字开头）

实例

一个简单的函数，在其被调用时能输出我的名称：

```
<html>
<body>

<?php
function writeName()
{
echo "Kai Jim Refsnes";
}

echo "My name is ";
writeName();
?>

</body>
</html>
```

输出：

```
My name is Kai Jim Refsnes
```

PHP 函数 - 添加参数

为了给函数添加更多的功能，我们可以添加参数。参数类似变量。

参数就在函数名称后面有一个括号内指定。

实例 1

下面的实例将输出不同的名字，但姓是相同的：

```
<html>
<body>

<?php
function writeName($fname)
{
echo $fname . " Refsnes.<br>";
}

echo "My name is ";
writeName("Kai Jim");
echo "My sister's name is ";
writeName("Hege");
echo "My brother's name is ";
writeName("Stale");
?>

</body>
</html>
```

输出：

```
My name is Kai Jim Refsnes.
My sister's name is Hege Refsnes.
My brother's name is Stale Refsnes.
```

实例 2

下面的函数有两个参数：

```
<html>
<body>

<?php
function writeName($fname,$punctuation)
{
echo $fname . " Refsnes" . $punctuation . "<br>";
}

echo "My name is ";
writeName("Kai Jim",".");
echo "My sister's name is ";
writeName("Hege","!");
echo "My brother's name is ";
writeName("Ståle","?");
?>

</body>
</html>
```

输出：

```
My name is Kai Jim Refsnes.
My sister's name is Hege Refsnes!
My brother's name is Ståle Refsnes?
```

PHP 函数 - 返回值

如需让函数返回一个值，请使用 return 语句。

实例

```
<html>
<body>

<?php
function add($x,$y)
{
$total=$x+$y;
return $total;
}

echo "1 + 16 = " . add(1,16);
?>

</body>
</html>
```

输出：

```
1 + 16 = 17
```

PHP è¡¨åå¯¹æ·è¾¾¥

PHP ä¸ç $_GET å $_POST åéç¨äºæ£ç´¢è¡¨åä¸çä¿¡æ¯ï¼æ¯å¦ç¨æ·è¾å¥ã

PHP è¡¨åå¤ç

æä¸ç¹å¾éè¦çäºæå¼å¾æ³¨æï¼å½å¤ç HTML è¡¨åæ¶ï¼PHP è½æèªå¨è HTML é¡µé¢ä¸çè¡¨åå´ è³å¨åæ¯ä¾ PHP èæ¬ä½¿ç¨ã

å®ä¾

ä¸é¢çå®ä¾åå«äºä¸ä¸ª HTML è¡¨åï¼å¸¦æä¸¤ä¸ªè¾å¥æ¡åä¸ä¸ªæäº¤æé®ã

form.html æä»¶ä»£ç ï¼

```
<pre> <html> <head> <meta charset="utf-8"> <title>èé¸æç¨(runoob.com)</title> </head> <body> <form action="welcome.php" method="post"> åå: <input type="text" name="fname"> å¹´é¾: <input type="text" name="age"> <input type="submit" value="æäº¤"> </form> </body> </html>
```

å½ç¨æ·å¡«åå®ä¸é¢çè¡¨åå¹¶ç¹å»æäº¤æé®æ¶ï¼è¡¨åæ°æ®ä¼è¢«éå°åä¸º "welcome.php" ç PHP æä»¶ï¼

welcome.php æä»¶ä»£ç ï¼

```
<pre> æ¬¢è¿ <?php echo $_POST["fname"]; ?>!<br> ä½ çå¹´é¾æ¯ <?php echo $_POST["age"]; ?> å²ã
```

éè¿æµè§å¨è®¿é®æ¼ç¤ºå¦ä¸ï¼

æä»¬å°å¨ä¸ä¸ç« èä¸è®²è§£ PHP ä¸ç $_GET å $_POST åéã

PHP è·åä¸æèåçæ°æ®

PHP ä¸æèååé

ä»¥ä¸å®ä¾æä»¬è®¾ç½®äºä¸æèåä¸ä¸ªéé¡¹ï¼è¡¨åä½¿ç¨ GET æ¹å¼è·åæ°æ®ï¼action å±æ§å¼ä¸ºç©ºè¡¨ç¤ºæäº¤å°å½åèæ¬ï¼æä»¬å¯ä»¥éè¿ select ç name å±æ§è·åä¸æèåçå¼ï¼

php_form_select.php æä»¶ä»£ç ï¼

```
<?php $q = isset($_GET['q'])? htmlspecialchars($_GET['q']) : ''; if($q) { if($q =='RUNOOB') { echo 'èé¸æç¨<br>http://www.runoob.com'; } else if($q =='GOOGLE') { echo 'Google æç´¢<br>http://www.google.com'; } else if($q =='TAOBAO') { echo 'æ·å®<br>http://www.taobao.com'; } } else { ?>
<form action="" method="get"> <select name="q"> <option value="">éæ©ä¸ä¸ªç«ç¹:</option> <option value="RUNOOB">Runoob</option> <option value="GOOGLE">Google</option> <option value="TAOBAO">Taobao</option> </select> <input type="submit" value="æäº¤"> </form>
<?php } ?>
```

PHP ä¸æèåå¤é

å¦æä¸æèåæ¯å¤éçï¼ multiple="multiple"ï¼ï¼æä»¬å¯ä»¥éè¿å°è®¾ç½® select **name="q[]"** ä»¥æ°ç»çæ¹å¼è·åï¼ä»¥ä¸ä½¿ç¨ POST æ¹å¼ï¼ä»£ç å¦ä¸æç¤ºï¼

php_form_select_mul.php æä»¶ä»£ç ï¼

```
<?php $q = isset($_POST['q'])? $_POST['q'] : ''; if(is_array($q)) { $sites = array( 'RUNOOB' => 'èé¸æç¨: http://www.runoob.com', 'GOOGLE' => 'Google æç´¢: http://www.google.com', 'TAOBAO' => 'æ·å®: http://www.taobao.com', ); foreach($q as $val) { // PHP_EOL ä¸ºå¸¸éï¼ç¨äºæ¢è¡ echo $sites[$val] . PHP_EOL; } } else { ?>
<form action="" method="post"> <select multiple="multiple" name="q[]"> <option value="">éæ©ä¸ä¸ªç«ç¹:</option> <option value="RUNOOB">Runoob</option> <option value="GOOGLE">Google</option> <option value="TAOBAO">Taobao</option> </select> <input type="submit" value="æäº¤"> </form>
<?php } ?>
```

åéæé®è¡¨å

PHP åéæé®è¡¨åä¸ name å±æ§çå¼æ¯ä¸è´çï¼value å¼æ¯ä¸åçï¼ä»£ç å¦ä¸æç¤ºï¼

php_form_radio.php æä»¶ä»£ç ï¼

```
<?php $q = isset($_GET['q'])? htmlspecialchars($_GET['q']) : ''; if($q) { if($q =='RUNOOB') { echo 'èé¸æç¨<br>http://www.runoob.com'; } else if($q =='GOOGLE') { echo 'Google æç´¢<br>http://www.google.com'; } else if($q =='TAOBAO') { echo 'æ·å®<br>http://www.taobao.com'; } } else {
?><form action="" method="get"> <input type="radio" name="q" value="RUNOOB" />Runoob <input type="radio" name="q" value="GOOGLE" />Google <input type="radio" name="q" value="TAOBAO" />Taobao <input type="submit" value="æäº¤"> </form>
<?php } ?>
```

---

checkbox å¤éæ¡

PHP checkbox å¤éæ¡å¯ä»¥éæ©å¤ä¸ªå¼ï¼

php_form_select_checkbox.php æä»¶ä»£ç ï¼

```
<?php $q = isset($_POST['q'])? $_POST['q'] : ''; if(is_array($q)) { $sites = array( 'RUNOOB' => 'èé¸æç¨: http://www.runoob.com', 'GOOGLE' => 'Google æç´¢: http://www.google.com', 'TAOBAO' => 'æ·å®: http://www.taobao.com', ); foreach($q as $val) { // PHP_EOL ä¸ºå¸¸éï¼ç¨äºæ¢è¡ echo $sites[$val] . PHP_EOL; } } else {
?><form action="" method="post"> <input type="checkbox" name="q[]" value="RUNOOB"> Runoob<br> <input type="checkbox" name="q[]" value="GOOGLE"> Google<br> <input type="checkbox" name="q[]" value="TAOBAO"> Taobao<br> <input type="submit" value="æäº¤"> </form>
<?php } ?>
```

è¡¨åéªè¯

åºè¯¥å¨ä»»ä½å¯è½çæ¶åå¯¹ç¨æ·è¾å¥è¿è¡éªè¯ï¼éè¿å®¢æ·ç«¯èæ¬ï¼ãæµè§å¨éªè¯éåº¦æ´å¿«ï¼å¹¶ä¸å¯ä»¥åè½»æå¡å¨çè´è½½ã

å¦ææ¨è¾å¥éè¦å¨æ°æ®åºä¸æå¥ï¼æ¨åºè¯¥èèä½¿ç¨æå¡å¨éªè¯ãå¨æå¡å¨éªè¯è¡¨åçä¸ç§å¥½çæ¹å¼æ¯ï¼æè¡¨åä¼ ç»å®èªå·±ï¼èä¸æ¯è·³è½¬å°ä¸åçé¡µé¢ãè¿æ ·ç¨æ·å°±å¯ä»¥å¨åä¸å¼ è¡¨åé¡µé¢å¾å°éè¯¯ä¿¡æ¯ãç¨æ·ä¹å°±æ´å®¹æåç°éè¯¯äºã

PHP $_GET åé

å¨ PHP ä¸ï¼é¢å®ä¹ç $_GET åéç¨äºæ¶éæ¥èª method="get" çè¡¨åä¸çå¼ã

$_GET åé

é¢å®ä¹ç $_GET åéç¨äºæ¶éæ¥èª method="get" çè¡¨åä¸çå¼ã

ä»å¸¦æ GET æ¹æ³çè¡¨ååéçä¿¡æ¯ï¼å¯¹ä»»ä½äººé½æ¯å¯è§çï¼ä¼æ¾ç¤ºå¨æµè§å¨çå°åæ ï¼ï¼å¹¶ä¸å¯¹åéä¿¡æ¯çéä¹æéå¶ã

å®ä¾

form.html æä»¶ä»£ç å¦ä¸ï¼

```
<html>
<head>
<meta charset="utf-8">
<title>èé¸æç¨(runoob.com)</title>
</head>
<body>

<form action="welcome.php" method="get">
åå: <input type="text" name="fname">
å¹´é¾: <input type="text" name="age">
<input type="submit" value="æäº¤">
</form>

</body>
</html>
```

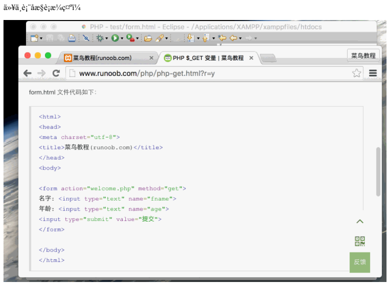
å½ç¨æ·ç¹å» "Submit" æé®æ¶ï¼åéå°æå¡å¨ç URL å¦ä¸æç¤ºï¼

```
http://www.runoob.com/welcome.php?fname=Runoob&amp;age=3
```

"welcome.php" æä»¶ç°å¨å¯ä»¥éè¿ $_GET åéæ¥æ¶éè¡¨åæ°æ®äºï¼è¯·æ³¨æï¼è¡¨ååçåç§°ä¼èªå¨æä¸º $_GET æ°ç»ä¸çé®ï¼ï¼

```
æ¬¢è¿ <?php echo $_GET["fname"]; ?>!<br>
ä½ çå¹´é¾æ¯ <?php echo $_GET["age"]; ?>  å²ã
```

ä»¥ä¸è¡¨åæ§è¡æ¼ç¤ºï¼



ä½æ¶ä½¿ç¨ method="get"ï¼

å¨ HTML è¡¨åä¸ä½¿ç¨ method="get" æ¶ï¼ææçåéåå¼é½æ¾ç¤ºå¨ URL ä¸ã

æ³¨éï¼æä»¥å¨åéå¯ç æå¶ä»ææä¿¡æ¯æ¶ï¼ä¸åºè¯¥ä½¿ç¨è¿ä¸ªæ¹æ³ï¼

ç¶èï¼æ£å ä¸ºåéæ¾ç¤ºå¨ URL ä¸ï¼å æ¤å¯ä»¥å¨æ¶èå¤¹ä¸æ¶èè¯¥é¡µé¢ãå¨æäºæåµä¸ï¼è¿æ¯å¾æç¨çã

æ³¨éï¼HTTP GET æ¹æ³ä¸éåå¤§åçåéå¼ãå®çå¼æ¯ä¸è½è¶è¿ 2000 ä¸ªåç¬¦çã

PHP $_POST 变量

在 PHP 中，预定义的 $_POST 变量用于收集来自 method="post" 的表单中的值。

$_POST 变量

预定义的 $_POST 变量用于收集来自 method="post" 的表单中的值。

从带有 POST 方法的表单发送的信息，对任何人都是不可见的（不会显示在浏览器的地址栏），并且对发送信息的量也没有限制。

注释：然而，默认情况下，POST 方法的发送信息的量最大值为 8 MB（可通过设置 php.ini 文件中的 post_max_size 进行更改）。

实例

form.html 文件代码如下：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<form action="welcome.php" method="post">
名字: <input type="text" name="fname">
年龄: <input type="text" name="age">
<input type="submit" value="提交">
</form>

</body>
</html>
```

当用户点击 "提交" 按钮时，URL 类似如下所示：

```
http://www.runoob.com/welcome.php
```

"welcome.php" 文件现在可以通过 $_POST 变量来收集表单数据了（请注意，表单域的名称会自动成为 $_POST 数组中的键）：

```
欢迎 <?php echo $_POST["fname"]; ?><br>
你的年龄是 <?php echo $_POST["age"]; ?>  岁。
```

通过浏览器访问演示如下：

何时使用 method="post"?

从带有 POST 方法的表单发送的信息，对任何人都是不可见的，并且对发送信息的量也没有限制。

然而，由于变量不显示在 URL 中，所以无法把页面加入书签。

PHP $_REQUEST 变量

预定义的 $_REQUEST 变量包含了 $_GET、$_POST 和 $_COOKIE 的内容。

$_REQUEST 变量可用来收集通过 GET 和 POST 方法发送的表单数据。

实例

你可以将 "welcome.php" 文件修改为如下代码，它可以接受 $_GET、$_POST等数据。

```
欢迎 <?php echo $_REQUEST["fname"]; ?><br>
你的年龄是 <?php echo $_REQUEST["age"]; ?>  岁。
```

PHP 多维数组

一个数组中的值可以是另一个数组，另一个数组的值也可以是一个数组。依照这种方式，我们可以创建二维或者三维数组：

实例

```
<?php
// 二维数组:
$cars = array
(
    array("Volvo",100,96),
    array("BMW",60,59),
    array("Toyota",110,100)
);
?>
```

运行实例 »

PHP - 多维数组

多维数组是包含一个或多个数组的数组。

在多维数组中，主数组中的每一个元素也可以是一个数组，子数组中的每一个元素也可以是一个数组。

实例

在这个实例中，我们创建了一个自动分配 ID 键的多维数组：

实例

```
<?php
$sites = array
(
    "runoob"=>array
    (
        "菜鸟教程",
        "http://www.runoob.com"
    ),
    "google"=>array
    (
        "Google 搜索",
        "http://www.google.com"
    ),
    "taobao"=>array
    (
        "淘宝",
        "http://www.taobao.com"
    )
);
print("<pre>"); // 格式化输出数组
print_r($sites);
print("</pre>");
?>
```

上面的数组将输出如下：

```
Array
(
    [runoob] => Array
        (
            [0] => 菜鸟教程
            [1] => http://www.runoob.com
        )

    [google] => Array
        (
            [0] => Google 搜索
            [1] => http://www.google.com
        )

    [taobao] => Array
        (
            [0] => 淘宝
            [1] => http://www.taobao.com
        )

)
```

实例 2

让我们试着显示上面数组中的某个值：

```
echo $sites['runoob'][0] . '地址为：' . $sites['runoob'][1];
```

上面的代码将输出：

```
菜鸟教程地址为： http://www.runoob.com
```

PHP date() 函数

PHP date() 函数用于格式化时间/日期。

PHP date() 函数

PHP date() 函数可把时间戳格式化为可读性更好的日期和时间。

时间戳是一个字符序列，表示一定的事件发生的日期/时间。

语法

```
string date ( string $format [, int $timestamp ] )
```

参数	描述
format	必需。规定时间戳的格式。
timestamp	可选。规定时间戳。默认是当前的日期和时间。

PHP Date() - 格式化日期

date() 函数的第一个必需参数 *format* 规定了如何格式化日期/时间。

这里列出了一些可用的字符：

- d - 代表月中的天 (01 - 31)
- m - 代表月 (01 - 12)
- Y - 代表年 (四位数)

如需了解 *format* 参数中可用的所有字符列表，请查阅我们的 PHP Date 参考手册：date() 函数。

可以在字母之间插入其他字符，比如 "/"、"." 或者 "-"，这样就可以增加附加格式了：

```
<?php
echo date("Y/m/d") . "<br>";
echo date("Y.m.d") . "<br>";
echo date("Y-m-d");
?>
```

上面代码的输出如下所示：

```
2016/10/21
2016.10.21
2016-10-21
```

格式字串可以识别以下 **format** 参数的字符串

format 字符	说明	返回值例子
日	---	---
d	月份中的第几天，有前导零的 2 位数字	*01* 到 *31*
D	星期中的第几天，文本表示，3 个字母	*Mon* 到 *Sun*
j	月份中的第几天，没有前导零	*1* 到 *31*
l（"L"的小写字母）	星期几，完整的文本格式	*Sunday* 到 *Saturday*
N	ISO-8601 格式数字表示的星期中的第几天（PHP 5.1.0 新加）	*1*（表示星期一）到 *7*（表示星期天）
S	每月天数后面的英文后缀，2 个字符	*st*，*nd*，*rd* 或者 *th*。可以和 *j* 一起用
w	星期中的第几天，数字表示	*0*（表示星期天）到 *6*（表示星期六）
z	年份中的第几天	*0* 到 *365*
星期	---	---
W	ISO-8601 格式年份中的第几周，每周从星期一开始（PHP 4.1.0 新加的）	例如：*42*（当年的第 42 周）
月	---	---
F	月份，完整的文本格式，例如 January 或者 March	*January* 到 *December*
m	数字表示的月份，有前导零	*01* 到 *12*
M	三个字母缩写表示的月份	*Jan* 到 *Dec*
n	数字表示的月份，没有前导零	*1* 到 *12*
t	给定月份所应有的天数	*28* 到 *31*
年	---	---
L	是否为闰年	如果是闰年为 *1*，否则为 *0*
o	ISO-8601 格式年份数字。这和 *Y* 的值相同，只除了如果 ISO 的星期数（*W*）属于前一年或下一年，则用那一年。（PHP 5.1.0 新加）	Examples: *1999* or *2003*
Y	4 位数字完整表示的年份	例如：*1999* 或 *2003*
y	2 位数字表示的年份	例如：*99* 或 *03*
时间	---	---
a	小写的上午和下午值	*am* 或 *pm*
A	大写的上午和下午值	*AM* 或 *PM*
B	Swatch Internet 标准时	*000* 到 *999*
g	小时，12 小时格式，没有前导零	*1* 到 *12*
G	小时，24 小时格式，没有前导零	*0* 到 *23*
h	小时，12 小时格式，有前导零	*01* 到 *12*
H	小时，24 小时格式，有前导零	*00* 到 *23*
i	有前导零的分钟数	*00* 到 *59*
s	秒数，有前导零	*00* 到 *59*
u	毫秒（PHP 5.2.2 新加）。需要注意的是 **date()** 函数总是返回 *000000* 因为它只接受 integer 参数，而 DateTime::format() 才支持毫秒。	示例: *654321*
时区	---	---
e	时区标识（PHP 5.1.0 新加）	例如：*UTC*，*GMT*，*Atlantic/Azores*
I	是否为夏令时	如果是夏令时为 *1*，否则为 *0*
O	与格林威治时间相差的小时数	例如：*+0200*
P	与格林威治时间（GMT）的差别，小时和分钟之间有冒号分隔（PHP 5.1.3 新加）	例如：*+02:00*
T	本机所在的时区	例如：*EST*，*MDT*（【译者注】在 Windows 下为完整文本格式，例如"Eastern Standard Time"，中文版会显示"中国标准时间"）。
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的。	*-43200* 到 *43200*
完整的日期 / 时间	---	---
c	ISO 8601 格式的日期（PHP 5 新加）	2004-02-12T15:19:21+00:00
r	RFC 822 格式的日期	例如：*Thu, 21 Dec 2000 16:01:07 +0200*
U	从 Unix 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数	参见 time()

完整的 PHP Date 参考手册

如需查看所有日期函数的完整参考手册，请访问我们的 完整的 PHP Date 参考手册。

该参考手册提供了每个函数的简要描述和应用实例！

PHP 包含文件

PHP include 和 require 语句

在 PHP 中，您可以在服务器执行 PHP 文件之前在该文件中插入一个文件的内容。

include 和 require 语句用于在执行流中插入写在其他文件中的有用的代码。

include 和 require 除了处理错误的方式不同之外，在其他方面都是相同的：

- require 生成一个致命错误（E_COMPILE_ERROR），在错误发生后脚本会停止执行。
- include 生成一个警告（E_WARNING），在错误发生后脚本会继续执行。

因此，如果您希望继续执行，并向用户输出结果，即使包含文件已丢失，那么请使用 include。否则，在框架、CMS 或者复杂的 PHP 应用程序编程中，请始终使用 require 向执行流引用关键文件。这有助于提高应用程序的安全性和完整性，在某个关键文件意外丢失的情况下。

包含文件省去了大量的工作。这意味着您可以为所有网页创建标准页头、页脚或者菜单文件。然后，在页头需要更新时，您只需更新这个页头包含文件即可。

语法

```
include 'filename';
```

或者

```
require 'filename';
```

PHP include 和 require 语句

基础实例

假设您有一个标准的页头文件，名为 "header.php"。如需在页面中引用这个页头文件，请使用 include/require：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<?php include 'header.php'; ?>
<h1>欢迎来到我的主页!</h1>
<p>一些文本。</p>

</body>
</html>
```

实例 2

假设我们有一个在所有页面中使用的标准菜单文件。

"menu.php":

```
echo '<a href="/">主页</a>
<a href="/html">HTML 教程</a>
<a href="/php">PHP 教程</a>';
```

网站中的所有页面均应引用该菜单文件。以下是具体的做法：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<div class="leftmenu">
<?php include 'menu.php'; ?>
</div>
<h1>欢迎来到我的主页!</h1>
<p>一些文本。</p>

</body>
</html>
```

实例 3

假设我们有一个定义变量的包含文件（"vars.php"）：

```
<?php
$color='red';
$car='BMW';
?>
```

这些变量可用在调用文件中：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<h1>欢迎来到我的主页!</h1>
<?php
include 'vars.php';
echo "I have a $color $car"; // I have a red BMW
?>

</body>
</html>
```

PHP 文件处理

fopen() 函数用于在 PHP 中打开文件。

打开文件

fopen() 函数用于在 PHP 中打开文件。

此函数的第一个参数含有要打开的文件的名称，第二个参数规定了使用哪种模式来打开文件：

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r");
?>

</body>
</html>
```

文件可能通过下列模式来打开：

模式	描述
r	只读。在文件的开头开始。
r+	读/写。在文件的开头开始。
w	只写。打开并清空文件的内容；如果文件不存在，则创建新文件。
w+	读/写。打开并清空文件的内容；如果文件不存在，则创建新文件。
a	追加。打开并向文件末尾进行写操作，如果文件不存在，则创建新文件。
a+	读/追加。通过向文件末尾写内容，来保持文件内容。
x	只写。创建新文件。如果文件已存在，则返回 FALSE 和一个错误。
x+	读/写。创建新文件。如果文件已存在，则返回 FALSE 和一个错误。

注释：如果 fopen() 函数无法打开指定文件，则返回 0 (false)。

实例

如果 fopen() 函数不能打开指定的文件，下面的实例会生成一段消息：

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
?>

</body>
</html>
```

关闭文件

fclose() 函数用于关闭打开的文件：

```
<?php
$file = fopen("test.txt","r");

//执行一些代码

fclose($file);
?>
```

检测文件末尾（EOF）

feof() 函数检测是否已到达文件末尾（EOF）。

在循环遍历未知长度的数据时，feof() 函数很有用。

注释：在 w 、a 和 x 模式下，您无法读取打开的文件！

if (feof($file)) echo "文件结尾";

逐行读取文件

fgets() 函数用于从文件中逐行读取文件。

注释：在调用该函数之后，文件指针会移动到下一行。

实例

下面的实例逐行读取文件，直到文件末尾为止：

```
<?php
$file = fopen("welcome.txt", "r") or exit("无法打开文件!");
// 读取文件每一行，直到文件结尾
while(!feof($file))
{
  echo fgets($file). "<br>";
}
fclose($file);
?>
```

逐字符读取文件

fgetc() 函数用于从文件中逐字符地读取文件。

注释：在调用该函数之后，文件指针会移动到下一个字符。

实例

下面的实例逐字符地读取文件，直到文件末尾为止：

```
<?php
$file=fopen("welcome.txt","r") or exit("无法打开文件!");
while (!feof($file))
{
  echo fgetc($file);
}
```

```
fclose($file);
?>
```

PHP Filesystem 参考手册

如需查看 PHP 文件系统函数的完整参考手册，请访问我们的PHP Filesystem 参考手册。

PHP 文件上传

通过 PHP，可以把文件上传到服务器。

本章节实例在 test 项目下完成，目录结构为：

```
test
|-----upload            # 文件上传的目录
|-----form.html         # 表单文件
|-----upload_file.php   # php 上传代码
```

源码下载

创建一个文件上传表单

允许用户从表单上传文件是非常有用的。

请看下面这个供上传文件的 HTML 表单：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<form action="upload_file.php" method="post" enctype="multipart/form-data">
 <label for="file">文件名：</label>
 <input type="file" name="file" id="file"><br>
 <input type="submit" name="submit" value="提交">
</form>

</body>
</html>
```

将以上代码保存到 form.html 文件中。

有关上面的 HTML 表单的一些注意项列举如下：

- **<form>** 标签的 **enctype** 属性规定了在提交表单时要使用哪种内容类型。在表单需要二进制数据时，比如文件内容，请使用 "**multipart/form-data**"。
- **<input>** 标签的 **type="file"** 属性规定了应该把输入作为文件来处理。举例来说，当在浏览器中预览时，会看到输入框旁边有一个浏览按钮。

注释：允许用户上传文件是一个巨大的安全风险。请仅仅允许可信的用户执行文件上传操作。

创建上传脚本

"upload_file.php" 文件含有供上传文件的代码：

```
<?php
if ($_FILES["file"]["error"] > 0)
{
 echo "错误: " . $_FILES["file"]["error"] . "<br>";
}
else
{
 echo "上传文件名: " . $_FILES["file"]["name"] . "<br>";
 echo "文件类型: " . $_FILES["file"]["type"] . "<br>";
 echo "文件大小: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
 echo "文件临时存储的位置: " . $_FILES["file"]["tmp_name"];
}
?>
```

通过使用 PHP 的全局数组 $_FILES。你可以从客户计算机向远程服务器上传文件。

第一个参数是表单的 input name，第二个下标可以是 "name"、"type"、"size"、"tmp_name" 或 "error"。如下所示：

- $_FILES["file"]["name"] - 上传文件的名称
- $_FILES["file"]["type"] - 上传文件的类型
- $_FILES["file"]["size"] - 上传文件的大小，以字节计
- $_FILES["file"]["tmp_name"] - 存储在服务器的文件的临时副本的名称
- $_FILES["file"]["error"] - 由文件上传导致的错误代码

这是一种非常简单文件上传方式。基于安全方面的考虑，您应当增加有关允许哪些用户上传文件的限制。

上传限制

在这个脚本中，我们增加了对文件上传的限制，用户只能上传 .gif、.jpeg、.jpg、.png 文件，文件大小必须小于 200 kB：

```
<?php
// 允许上传的图片后缀
$allowedExts = array("gif", "jpeg", "jpg", "png");
$temp = explode(".", $_FILES["file"]["name"]);
$extension = end($temp);     // 获取文件后缀名
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/jpg")
|| ($_FILES["file"]["type"] == "image/pjpeg")
|| ($_FILES["file"]["type"] == "image/x-png")
|| ($_FILES["file"]["type"] == "image/png"))
&& ($_FILES["file"]["size"] < 204800)   // 小于 200 kb
&& in_array($extension, $allowedExts))
{
 if ($_FILES["file"]["error"] > 0)
 {
  echo "错误: : " . $_FILES["file"]["error"] . "<br>";
 }
 else
 {
  echo "上传文件名: " . $_FILES["file"]["name"] . "<br>";
  echo "文件类型: " . $_FILES["file"]["type"] . "<br>";
  echo "文件大小: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
  echo "文件临时存储的位置: " . $_FILES["file"]["tmp_name"];
 }
}
else
{
 echo "非法的文件格式";
}
?>
```

保存被上传的文件

上面的实例在服务器的 PHP 临时文件夹中创建了一个被上传文件的临时副本。

这个临时的副本文件会在脚本结束时消失。要保存被上传的文件，我们需要把它拷贝到另外的位置：

```
<?php
// 允许上传的图片后缀
$allowedExts = array("gif", "jpeg", "jpg", "png");
$temp = explode(".", $_FILES["file"]["name"]);
echo $_FILES["file"]["size"];
$extension = end($temp);     // 获取文件后缀名
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/jpg")
|| ($_FILES["file"]["type"] == "image/pjpeg")
|| ($_FILES["file"]["type"] == "image/x-png")
|| ($_FILES["file"]["type"] == "image/png"))
&& ($_FILES["file"]["size"] < 204800)   // 小于 200 kb
&& in_array($extension, $allowedExts))
{
 if ($_FILES["file"]["error"] > 0)
 {
  echo "错误: : " . $_FILES["file"]["error"] . "<br>";
 }
 else
 {
  echo "上传文件名: " . $_FILES["file"]["name"] . "<br>";
  echo "文件类型: " . $_FILES["file"]["type"] . "<br>";
  echo "文件大小: " . ($_FILES["file"]["size"] / 1024) . " kB<br>";
  echo "文件临时存储的位置: " . $_FILES["file"]["tmp_name"] . "<br>";

  // 判断当期目录下的 upload 目录是否存在该文件
  // 如果没有 upload 目录，你需要创建它，upload 目录权限为 777
  if (file_exists("upload/" . $_FILES["file"]["name"]))
  {
   echo $_FILES["file"]["name"] . " 文件已经存在。 ";
  }
  else
  {
   // 如果 upload 目录不存在该文件则将文件上传到 upload 目录下
   move_uploaded_file($_FILES["file"]["tmp_name"], "upload/" . $_FILES["file"]["name"]);
   echo "文件存储在: " . "upload/" . $_FILES["file"]["name"];
  }
 }
}
else
{
 echo "非法的文件格式";
}
?>
```

上面的脚本检测了文件是否已存在，如果不存在，则把文件拷贝到名为 "upload" 的目录下。

文件上传演示操作如下所示：

PHP Cookie

---

cookie 常用于识别用户。

Cookie 是什么？

cookie 常用于识别用户。cookie 是一种服务器留在用户计算机上的小文件。每当同一台计算机通过浏览器请求页面时，这台计算机将会发送 cookie。通过 PHP，您能够创建并取回 cookie 的值。

如何创建 Cookie？

setcookie() 函数用于设置 cookie。

注释：setcookie() 函数必须位于 <html> 标签之前。

语法

```
setcookie(name, value, expire, path, domain);
```

实例 1

在下面的例子中，我们将创建名为 "user" 的 cookie，并为它赋值 "runoob"。我们也规定了此 cookie 在一小时后过期：

```
<?php
setcookie("user", "runoob", time()+3600);
?>
```

```
<html>
.....
```

注释：在发送 cookie 时，cookie 的值会自动进行 URL 编码，在取回时进行自动解码。（为防止 URL 编码，请使用 setrawcookie() 取而代之。）

实例 2

您还可以通过另一种方式设置 cookie 的过期时间。这也许比使用秒表示的方式简单。

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "runoob", $expire);
?>

<html>
.....
```

在上面的实例中，过期时间被设置为一个月（*60 秒 * 60 分 * 24 小时 * 30 天*）。

如何取回 Cookie 的值?

PHP 的 $_COOKIE 变量用于取回 cookie 的值。

在下面的实例中，我们取回了名为 "user" 的 cookie 的值，并把它显示在了页面上：

```
<?php
// 输出 cookie 值
echo $_COOKIE["user"];

// 查看所有 cookie
print_r($_COOKIE);
?>
```

在下面的实例中，我们使用 isset() 函数来确认是否已设置了 cookie：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<?php
if (isset($_COOKIE["user"]))
  echo "欢迎 " . $_COOKIE["user"] . "!<br>";
else
  echo "普通访客!<br>";
?>

</body>
</html>
```

如何删除 Cookie?

当删除 cookie 时，您应当使过期日期变更为过去的时间点。

删除的实例：

```
<?php
// 设置 cookie 过期时间为过去 1 小时
setcookie("user", "", time()-3600);
?>
```

如果浏览器不支持 Cookie 该怎么办?

如果您的应用程序需要与不支持 cookie 的浏览器打交道，那么您不得不使用其他的办法在您的应用程序中的页面之间传递信息。一种方式是通过表单传递数据（有关表单和用户输入的内容，在本教程的前面章节中我们已经介绍过了）。

下面的表单在用户单点击 "Submit" 按钮时，向 "welcome.php" 提交了用户输入：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<form action="welcome.php" method="post">
名字: <input type="text" name="name">
年龄: <input type="text" name="age">
<input type="submit">
</form>

</body>
</html>
```

取回 "welcome.php" 文件中的值，如下所示：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

欢迎 <?php echo $_POST["name"]; ?>.<br>
你 <?php echo $_POST["age"]; ?> 岁了。

</body>
</html>
```

PHP Session

PHP session 变量用于存储关于用户会话（session）的信息，或者更改用户会话（session）的设置。Session 变量存储单一用户的信息，并且对于应用程序中的所有页面都是可用的。

PHP Session 变量

您在计算机上操作某个应用程序时，您打开它，做些更改，然后关闭它。这很像一次对话（Session）。计算机知道您是谁。它清楚您在何时打开和关闭应用程序。然而，在因特网上问题出现了：由于 HTTP 地址无法保持状态，Web 服务器并不知道您是谁以及您做了什么。

PHP session 解决了这个问题，它通过在服务器上存储用户信息以便随后使用（比如用户名称、购买商品等）。然而，会话信息是临时的，在用户离开网站后将被删除。如果您需要永久存储信息，可以把数据存储在数据库中。

Session 的工作机制是：为每个访客创建一个唯一的 id (UID)，并基于这个 UID 来存储变量。UID 存储在 cookie 中，或者通过 URL 进行传导。

开始 PHP Session

在您把用户信息存储到 PHP session 中之前，首先必须启动会话。

注释：session_start() 函数必须位于 <html> 标签之前：

```
<?php session_start(); ?>

<html>
<body>

</body>
</html>
```

上面的代码会向服务器注册用户的会话，以便您可以开始保存用户信息，同时会为用户会话分配一个 UID。

存储 Session 变量

存储和取回 session 变量的正确方法是使用 PHP $_SESSION 变量：

```
<?php
session_start();
// 存储 session 数据
$_SESSION['views']=1;
?>

<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<?php
// 检索 session 数据
echo "浏览量： ". $_SESSION['views'];
?>

</body>
</html>
```

输出：

```
浏览量：1
```

在下面的实例中，我们创建了一个简单的 page-view 计数器。isset() 函数检测是否已设置 "views" 变量。如果已设置 "views" 变量，我们累加计数器。如果 "views" 不存在，则创建 "views" 变量，并把它设置为 1：

```
<?php
session_start();

if(isset($_SESSION['views']))
{
  $_SESSION['views']=$_SESSION['views']+1;
}
else
{
  $_SESSION['views']=1;
}
echo "浏览量：". $_SESSION['views'];
?>
```

销毁 Session

如果您希望删除某些 session 数据，可以使用 unset() 或 session_destroy() 函数。

unset() 函数用于释放指定的 session 变量：

```
<?php
session_start();
if(isset($_SESSION['views']))
{
  unset($_SESSION['views']);
}
?>
```

您也可以通过调用 session_destroy() 函数彻底销毁 session：

```
<?php
session_destroy();
?>
```

注释：session_destroy() 将重置 session，您将失去所有已存储的 session 数据。

PHP 发送电子邮件

PHP 允许您从脚本直接发送电子邮件。

PHP mail() 函数

PHP mail() 函数用于从脚本中发送电子邮件。

语法

mail(to,subject,message,headers,parameters)

参数	描述
to	必需。规定 email 接收者。
subject	必需。规定 email 的主题。注释：该参数不能包含任何新行字符。
message	必需。定义要发送的消息。应使用 LF (\n) 来分隔各行。每行应该限制在 70 个字符内。
headers	可选。规定附加的标题，比如 From、Cc 和 Bcc。应当使用 CRLF (\r\n) 分隔附加的标题。
parameters	可选。对邮件发送程序规定额外的参数。

注释：PHP 运行邮件函数需要一个已安装且正在运行的邮件系统(如：sendmail、postfix、qmail等)。所用的程序通过在 php.ini 文件中的配置设置进行定义。请在我们的 PHP Mail 参考手册 阅读更多内容。

PHP 简易 E-Mail

通过 PHP 发送电子邮件的最简单的方式是发送一封文本 email。

在下面的实例中，我们首先声明变量($to, $subject, $message, $from, $headers)，然后我们在 mail() 函数中使用这些变量来发送了一封 E-mail：

```
<?php
$to = "someone@example.com";         // 邮件接收者
$subject = "带有附件 2 的邮件";       // 邮件标题
$message = "Hello! 这是邮件的内容。"; // 邮件内容
$from = "anonymous@example.com";     // 邮件发送者
$headers = "From: " . $from;         // 头部信息设置
mail($to,$subject,$message,$headers);
echo "邮件已发送";
?>
```

PHP Mail 表单

通过 PHP，您能够在自己的站点制作一个反馈表单。下面的实例向指定的 e-mail 地址发送了一条文本消息：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<?php
if (isset($_REQUEST['email'])) { // 如果接收到邮箱参数则发送邮件
 // 发送邮件
 $email = $_REQUEST['email'] ;
 $subject = $_REQUEST['subject'] ;
 $message = $_REQUEST['message'] ;
 mail("someone@example.com", $subject,
 $message, "From:" . $email);
 echo "邮件发送成功";
} else { // 如果没有邮箱参数则显示表单
 echo "<form method='post' action='mailform.php'>
 Email: <input name='email' type='text'><br>
 Subject: <input name='subject' type='text'><br>
 Message:<br>
 <textarea name='message' rows='15' cols='40'>
 </textarea><br>
 <input type='submit'>
 </form>";
}
?>

</body>
</html>
```

实例解释：

- 首先，检查是否填写了邮件输入框
- 如果未填写（比如在页面被首次访问时），输出 HTML 表单
- 如果已填写（在表单被填写后），从表单发送电子邮件
- 当填写完表单点击提交按钮后，页面重新载入，可以看到邮件输入被重置，同时显示邮件发送成功的消息

注释：这个简易发送 e-mail 不安全。在本教程的下一章中，您将阅读到更多关于电子邮件脚本中的安全隐患，我们将为您讲解如何验证用户输入使它更安全。

PHP Mail 参考手册

如需查看更多关于 PHP mail() 函数的信息，请访问我们的 PHP Mail 参考手册。

PHP Secure E-mails

在上一节中的 PHP e-mail 脚本中，存在着一个漏洞。

PHP E-mail 注入

首先，请看上一章中的 PHP 代码：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

<?php
if (isset($_REQUEST['email'])) { // 如果接收到邮箱参数则发送邮件
 // 发送邮件
 $email = $_REQUEST['email'] ;
 $subject = $_REQUEST['subject'] ;
 $message = $_REQUEST['message'] ;
 mail("someone@example.com", $subject,
 $message, "From:" . $email);
 echo "邮件发送成功";
} else { // 如果没有邮箱参数则显示表单
 echo "<form method='post' action='mailform.php'>
 Email: <input name='email' type='text'><br>
 Subject: <input name='subject' type='text'><br>
 Message:<br>
 <textarea name='message' rows='15' cols='40'>
 </textarea><br>
 <input type='submit'>
 </form>";
}
?>

</body>
</html>
```

以上代码存在的问题是，未经授权的用户可通过输入表单在邮件头部插入数据。

假如用户在表单中的输入框内加入如下文本到电子邮件中，会出现什么错况呢？

```
someone@example.com%0ACc:person2@example.com
%0ABcc:person3@example.com,person3@example.com,
anotherperson4@example.com,person5@example.com
%0ATo:person6@example.com
```

与往常一样，mail() 函数把上面的文本放入邮件头部。那么现在头部有了额外的 Cc:、Bcc: 和 To: 字段。当用户点击提交按钮时，这封 e-mail 会被发送到上面所有的地址！

PHP 防止 E-mail 注入

防止 e-mail 注入的最好方法是对输入进行验证。

下面的代码与上一章中的类似，不过这里我们已经增加了检测表单中 email 字段的输入验证程序：

```
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>
<?php
function spamcheck($field)
{
 // filter_var() 过滤 e-mail
 // 使用 FILTER_SANITIZE_EMAIL
 $field=filter_var($field, FILTER_SANITIZE_EMAIL);

 //filter_var() 过滤 e-mail
 // 使用 FILTER_VALIDATE_EMAIL
 if(filter_var($field, FILTER_VALIDATE_EMAIL))
 {
  return TRUE;
 }
 else
 {
  return FALSE;
 }
}

if (isset($_REQUEST['email']))
{
 // 如果接收到邮箱参数则发送邮件

 // 判断邮箱是否合法
 $emailcheck = spamcheck($_REQUEST['email']);
 if ($emailcheck==FALSE)
 {
  echo "非法输入";
 }
 else
 {
  // 发送邮件
  $email = $_REQUEST['email'] ;
  $subject = $_REQUEST['subject'] ;
  $message = $_REQUEST['message'] ;
  mail("someone@example.com", "Subject: $subject",
  $message, "From: $email" );
  echo "Thank you for using our mail form";
 }
}
else
{
 // 如果没有邮箱参数则显示表单
 echo "<form method='post' action='mailform.php'>
 Email: <input name='email' type='text'><br>
 Subject: <input name='subject' type='text'><br>
 Message:<br>
 <textarea name='message' rows='15' cols='40'>
 </textarea><br>
 <input type='submit'>
 </form>";
}
?>

</body>
</html>
```

在上面的代码中，我们使用了 PHP 过滤器来对输入进行验证：

- FILTER_SANITIZE_EMAIL 过滤器从字符串中删除电子邮件的非法字符
- FILTER_VALIDATE_EMAIL 过滤器验证电子邮件地址的值

您可以在我们的 PHP Filter 中阅读更多关于过滤器的知识。

PHP 错误处理

在 PHP 中，默认的错误处理很简单。一条错误消息会被发送到浏览器，这条消息带有文件名，行号以及描述错误的消息。

PHP 错误处理

在创建脚本和 Web 应用程序时，错误处理是一个重要的部分。如果您的代码缺少错误检测编码，那么程序看上去很不专业，也为安全风险敞开了大门。

本教程介绍了 PHP 中一些最为重要的错误检测方法。

我们将为您讲解不同的错误处理方法：

- 简单的 "die()" 语句
- 自定义错误和错误触发器
- 错误报告

基本的错误处理：使用 die() 函数

第一个实例展示了一个打开文本文件的简单脚本：

```
<?php
$file=fopen("welcome.txt","r");
?>
```

如果文件不存在，您会得到类似这样的错误：

```
Warning: fopen(welcome.txt) [function.fopen]: failed to open stream:
No such file or directory in /www/runoob/test/test.php on line 2
```

为了避免用户得到类似上面的错误消息，我们在访问文件之前检测该文件是否存在：

```
<?php
if(!file_exists("welcome.txt"))
{
 die("文件不存在");
}
else
{
 $file=fopen("welcome.txt","r");
}
?>
```

现在，如果文件不存在，您会得到类似这样的错误消息：

```
文件不存在
```

相比之前的代码，上面的代码更有效，这是由于它采用了一个简单的错误处理机制在错误之后终止了脚本。

然而，简单地终止脚本并不总是恰当的方式。让我们研究一下用于处理错误的备选的 PHP 函数。

创建自定义错误处理器

创建一个自定义的错误处理器非常简单。我们很简单地创建了一个专用函数，可以在 PHP 中发生错误时调用该函数。

该函数必须有能力处理至少两个参数 (error level 和 error message)，但是可以接受最多五个参数（可选的：file, line-number 和 error context）：

语法

```
error_function(error_level,error_message,
error_file,error_line,error_context)
```

参数	描述
error_level	必需。为用户定义的错误规定错误报告级别。必须是一个数字。参见下面的表格：错误报告级别。
error_message	必需。为用户定义的错误规定错误消息。
error_file	可选。规定错误发生的文件名。
error_line	可选。规定错误发生的行号。
error_context	可选。规定一个数组，包含了当错误发生时在用的每个变量以及它们的值。

错误报告级别

这些错误报告级别是用户自定义的错误处理程序处理的不同类型的错误：

值	常量	描述
2	E_WARNING	非致命的 run-time 错误。不暂停脚本执行。
8	E_NOTICE	run-time 通知。在脚本发现可能有错误时发生，但也可能在脚本正常运行时发生。
256	E_USER_ERROR	致命的用户生成的错误。这类似于程序员使用 PHP 函数 trigger_error() 设置的 E_ERROR。
512	E_USER_WARNING	非致命的用户生成的警告。这类似于程序员使用 PHP 函数 trigger_error() 设置的 E_WARNING。
1024	E_USER_NOTICE	用户生成的通知。这类似于程序员使用 PHP 函数 trigger_error() 设置的 E_NOTICE。
4096	E_RECOVERABLE_ERROR	可捕获的致命错误。类似 E_ERROR，但可被用户定义的处理程序捕获。（参见 set_error_handler()）
8191	E_ALL	所有错误和警告。（在 PHP 5.4 中，E_STRICT 成为 E_ALL 的一部分）

现在，让我们创建一个处理错误的函数：

```
function customError($errno, $errstr)
{
 echo "<b>Error:</b> [$errno] $errstr<br>";
 echo "脚本结束";
 die();
}
```

上面的代码是一个简单的错误处理函数。当它被触发时，它会取得错误级别和错误消息。然后它会输出错误级别和消息，并终止脚本。

现在，我们已经创建了一个错误处理函数，我们需要确定在何时触发该函数。

设置错误处理程序

PHP 的默认错误处理程序是内建的错误处理程序。我们打算把上面的函数改造为脚本运行期间的默认错误处理程序。

可以修改错误处理程序，使其仅应用到某些错误，这样脚本就能以不同的方式来处理不同的错误。然而，在本例中，我们打算针对所有错误来使用我们自定义的错误处理程序：

```
set_error_handler("customError");
```

由于我们希望我们的自定义函数能处理所有错误，set_error_handler() 仅需要一个参数，可以添加第二个参数来规定错误级别。

实例

通过尝试输出不存在的变量，来测试这个错误处理程序：

```
<?php
// 错误处理函数
function customError($errno, $errstr)
{
 echo "<b>Error:</b> [$errno] $errstr";
}

// 设置错误处理函数
set_error_handler("customError");

// 触发错误
echo($test);
?>
```

以上代码的输出如下所示：

```
Error: [8] Undefined variable: test
```

触发错误

在脚本中用户输入数据的位置，当用户的输入无效时触发错误是很有用的。在 PHP 中，这个任务由 trigger_error() 函数完成。

实例

在本例中，如果 "test" 变量大于 "1"，就会发生错误：

```
<?php
$test=2;
if ($test>1)
{
 trigger_error("变量值必须小于等于 1");
}
?>
```

以上代码的输出如下所示：

```
Notice: 变量值必须小于等于 1
in /www/test/runoob.php on line 5
```

您可以在脚本中任何位置触发错误，通过添加的第二个参数，您能够规定所触发的错误级别。

可能的错误类型：

- E_USER_ERROR - 致命的用户生成的 run-time 错误。错误无法恢复。脚本执行被中断。
- E_USER_WARNING - 非致命的用户生成的 run-time 警告。脚本执行不被中断。
- E_USER_NOTICE - 默认。用户生成的 run-time 通知。在脚本发现可能有错误时发生，但也可能在脚本正常运行时发生。

实例

在本例中，如果 "test" 变量大于 "1"，则发生 E_USER_WARNING 错误。如果发生了 E_USER_WARNING，我们将使用我们自定义的错误处理程序并结束脚本：

```
<?php
// 错误处理函数
function customError($errno, $errstr)
{
 echo "<b>Error:</b> [$errno] $errstr<br>";
 echo "脚本结束";
 die();
}

// 设置错误处理函数
set_error_handler("customError",E_USER_WARNING);

// 触发错误
$test=2;
if ($test>1)
{
 trigger_error("变量值必须小于等于 1",E_USER_WARNING);
}
?>
```

以上代码的输出如下所示：

```
Error: [512] 变量值必须小于等于 1
脚本结束
```

现在，我们已经学习了如何创建自己的 error，以及如何触发它们，接下来我们研究一下错误记录。

错误记录

在默认的情况下，根据在 php.ini 中的 error_log 配置，PHP 向服务器的记录系统或文件发送错误记录。通过使用 error_log() 函数，您可以向指定的文件或远程目的地发送错误记录。

通过电子邮件向您自己发送错误消息，是一种获得指定错误的通知的好办法。

通过 E-Mail 发送错误消息

在下面的例子中，如果特定的错误发生，我们将发送带有错误消息的电子邮件，并结束脚本：

```
<?php
// 错误处理函数
function customError($errno, $errstr)
{
 echo "<b>Error:</b> [$errno] $errstr<br>";
 echo "已通知网站管理员";
 error_log("Error: [$errno] $errstr",1,
 "someone@example.com","From: webmaster@example.com");
}

// 设置错误处理函数
set_error_handler("customError",E_USER_WARNING);

// 触发错误
$test=2;
if ($test>1)
{
 trigger_error("变量值必须小于等于 1",E_USER_WARNING);
}
?>
```

以上代码的输出如下所示：

```
Error: [512] 变量值必须小于等于 1
已通知网站管理员
```

接收自以上代码的邮件如下所示：

```
Error: [512] 变量值必须小于等于 1
```

这个方法不适合所有的错误。常规错误应当通过使用默认的 PHP 记录系统在服务器上进行记录。

PHP 异常处理

异常用于在指定的错误发生时改变脚本的正常流程。

异常是什么

PHP 5 提供了一种新的面向对象的错误处理方法。

异常处理用于在指定的错误（异常）情况发生时改变脚本的正常流程。这种情况称为异常。

当异常被触发时，通常会发生：

- 当前代码状态被保存
- 代码执行被切换到预定义（自定义）的异常处理器函数
- 根据情况，处理器也许会从保存的代码状态重新开始执行代码，终止脚本执行，或从代码中另外的位置继续执行脚本

我们将展示不同的错误处理方法：

- 异常的基本使用
- 创建自定义的异常处理器
- 多个异常
- 重新抛出异常
- 设置顶层异常处理器

注释：异常应该仅仅在错误情况下使用，而不应该用于在一个指定的点跳转到代码的另一个位置。

异常的基本使用

当异常被抛出时，其后的代码不会继续执行，PHP 会尝试查找匹配的 "catch" 代码块。

如果异常没有被捕获，而且又没用使用 set_exception_handler() 作相应的处理的话，那么将发生一个严重的错误（致命错误），并且输出 "Uncaught Exception"（未捕获异常）的错误消息。

让我们尝试抛出一个异常，同时不去捕获它：

<?php // 创建一个有异常处理的函数 function checkNum($number) { if($number>1) { throw new Exception("Value must be 1 or below"); } return true; } // 触发异常 checkNum(2); ?>

上面的代码会得到类似这样的一个错误：

```
Fatal error: Uncaught exception 'Exception' with message 'Value must be 1 or below' in /www/runoob/test/test.php:7 Stack trace: #0 /www/runoob/test/test.php(13): checkNum(2) #1 {main} thrown in /www/runoob/test/test.php on line 7
```

Try、throw 和 catch

要避免上面实例中出现的错误，我们需要创建适当的代码来处理异常。

适当的处理异常代码应该包括：

1. Try - 使用异常的函数应该位于 "try" 代码块内。如果没有触发异常，则代码将照常继续执行。但是如果异常被触发，会抛出一个异常。
2. Throw - 里规定如何触发异常。每一个 "throw" 必须对应至少一个 "catch"。
3. Catch - "catch" 代码块会捕获异常，并创建一个包含异常信息的对象。

让我们触发一个异常：

<?php // 创建一个有异常处理的函数 function checkNum($number) { if($number>1) { throw new Exception("变量值必须小于等于 1"); } return true; } // 在 try 块 触发异常 try { checkNum(2); // 如果抛出异常，以下文本不会输出 echo '如果输出该内容，说明 $number 变量'; } // 捕获异常 catch(Exception $e) { echo 'Message: ' .$e->getMessage(); } ?>

上面代码将得到类似这样一个错误：

```
Message: 变量值必须小于等于 1
```

实例解释：

上面的代码抛出了一个异常，并捕获了它：

1. 创建 checkNum() 函数。它检测数字是否大于 1。如果是，则抛出一个异常。
2. 在 "try" 代码块中调用 checkNum() 函数。
3. checkNum() 函数中的异常被抛出。
4. "catch" 代码块接收到该异常，并创建一个包含异常信息的对象 ($e)。
5. 通过从这个 exception 对象调用 $e->getMessage()，输出来自该异常的错误消息。

然而，为了遵循 "每个 throw 必须对应一个 catch" 的原则，可以设置一个顶层的异常处理器来处理漏掉的错误。

创建一个自定义的 Exception 类

创建自定义的异常处理程序非常简单。我们简单地创建了一个专门的类，当 PHP 中发生异常时，可调用其函数。该类必须是 exception 类的一个扩展。

这个自定义的 customException 类继承了 PHP 的 exception 类的所有属性。您可向其添加自定义的函数。

我们开始创建 customException 类：

<?php class customException extends Exception { public function errorMessage() { // 错误信息 $errorMsg = '错误行号 '.$this->getLine().' in '.$this->getFile() .': '.$this->getMessage().' 不是一个合法的 E-Mail 地址'; return $errorMsg; } } $email = "someone@example...com"; try { // 检测邮箱 if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) { // 如果是个不合法的邮箱地址，抛出异常 throw new customException($email); } } catch (customException $e) { //display custom message echo $e->errorMessage(); } ?>

这个新的类是旧的 exception 类的副本，外加 errorMessage() 函数。正因为它是旧类的副本，因此它从旧类继承了属性和方法，我们可以使用 exception 类的方法，比如 getLine()、getFile() 和 getMessage()。

实例解释：

上面的代码抛出了一个异常，并通过一个自定义的 exception 类来捕获它：

1. customException() 类是作为旧的 exception 类的一个扩展来创建的。这样它就继承了旧的 exception 类的所有属性和方法。
2. 创建 errorMessage() 函数。如果 e-mail 地址不合法，则该函数返回一条错误消息。
3. 把 $email 变量设置为一个合法的 e-mail 地址字符串。
4. 执行 "try" 代码块，由于 e-mail 地址不合法，因此抛出一个异常。
5. "catch" 代码块捕获异常，并显示错误消息。

多个异常

可以为一段脚本使用多个异常，来检测多种情况。

可以使用多个 if..else 代码块，或一个 switch 代码块，或者嵌套多个异常。这些异常能够使用不同的 exception 类，并返回不同的错误消息：

<?php class customException extends Exception { public function errorMessage() { // 错误信息 $errorMsg = '错误行号 '.$this->getLine().' in '.$this->getFile() .': '.$this->getMessage().' 不是一个合法的 E-Mail 地址'; return $errorMsg; } } $email = "someone@example.com"; try { // 检测邮箱 if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) { // 如果是个不合法的邮箱地址，抛出异常 throw new customException($email); } // 检测 "example" 是否在邮箱地址中 if(strpos($email, "example") !== FALSE) { throw new Exception("$email 是 example 邮箱"); } } catch (customException $e) { echo $e->errorMessage(); } catch(Exception $e) { echo $e->getMessage(); } ?>

实例解释：

上面的代码测试了两种条件。如果其中任何一个条件不成立，则抛出一个异常：

1. customException() 类是作为旧的 exception 类的一个扩展来创建的。这样它就继承了旧的 exception 类的所有属性和方法。
2. 创建 errorMessage() 函数。如果 e-mail 地址不合法，则该函数返回一个错误消息。
3. 把 $email 变量设置为一个字符串，该字符串是一个有效的 e-mail 地址，但包含字符串 "example"。
4. 执行 "try" 代码块，在第一个条件下，不会抛出异常。
5. 由于 e-mail 含有字符串 "example"，第二个条件会触发异常。
6. "catch" 代码块会捕获异常，并显示恰当的错误消息。

如果 customException 类抛出了异常，但没有捕获 customException，仅仅捕获了 base exception，则在那里处理异常。

重新抛出异常

有时，当异常被抛出时，您也许希望以不同于标准的方式对它进行处理。可以在一个 "catch" 代码块中再次抛出异常。

脚本应该对用户隐藏系统错误。对程序员来说，系统错误也许很重要，但是用户对它们并不感兴趣。为了让用户更容易使用，您可以再次抛出带有对用户比较友好的消息的异常：

<?php class customException extends Exception { public function errorMessage() { // 错误信息 $errorMsg = $this->getMessage().' 不是一个合法的 E-Mail 地址。'; return $errorMsg; } } $email = "someone@example.com"; try { try { // 检测 "example" 是否在邮箱地址中 if(strpos($email, "example") !== FALSE) { // 如果是个不合法的邮箱地址，抛出异常 throw new Exception($email); } } catch(Exception $e) { // 重新抛出异常 throw new customException($email); } } catch (customException $e) { // 显示自定义信息 echo $e->errorMessage(); } ?>

实例解释：

上面的代码检测在邮件地址中是否含有字符串 "example"。如果有，则再次抛出异常：

1. customException() 类是作为旧的 exception 类的一个扩展来创建的。这样它就继承了旧的 exception 类的所有属性和方法。
2. 创建 errorMessage() 函数。如果 e-mail 地址不合法，则该函数返回一个错误消息。
3. 把 $email 变量设置为一个字符串，该字符串是一个有效的 e-mail 地址，但包含字符串 "example"。
4. "try" 代码块包含另一个 "try" 代码块，这样就可以再次抛出异常。
5. 由于 e-mail 包含字符串 "example"，因此触发异常。
6. "catch" 代码块捕获到该异常，并重新抛出 "customException"。
7. 捕获到 "customException"，并显示一条错误消息。

如果在当前的 "try" 代码块中异常没有被捕获，则它将在更高层级上查找 catch 代码块。

设置顶层异常处理器

set_exception_handler() 函数可设置处理所有未捕获异常的用户定义函数。

<?php function myException($exception) { echo "Exception: " , $exception->getMessage(); } set_exception_handler('myException'); throw new Exception('Uncaught Exception occurred'); ?>

以上代码的输出如下所示：

```
Exception: Uncaught Exception occurred
```

在上面的代码中，不存在 "catch" 代码块，而是触发顶层的异常处理程序。应该使用此函数来捕获所有未被捕获的异常。

异常的规则

- 需要进行异常处理的代码应该放入 try 代码块内，以便捕获潜在的异常。
- 每个 try 或 throw 代码块必须至少拥有一个对应的 catch 代码块。
- 使用多个 catch 代码块可以捕获不同种类的异常。
- 可以在 try 代码块内的 catch 代码块中抛出（再次抛出）异常。

简而言之：如果抛出了异常，就必须捕获它。

PHP 过滤器

PHP 过滤器用于验证和过滤来自非安全来源的数据，比如用户的输入。

什么是 PHP 过滤器？

PHP 过滤器用于验证和过滤来自非安全来源的数据。

测试、验证和过滤用户输入或自定义数据是任何 Web 应用程序的重要组成部分。

PHP 的过滤器扩展的设计目的是使数据过滤更轻松快捷。

为什么使用过滤器？

几乎所有的 Web 应用程序都依赖外部的输入。这些数据通常来自用户或其他应用程序（比如 web 服务）。通过使用过滤器，您能够确保应用程序获得正确的输入类型。

您应该始终对外部数据进行过滤！

输入过滤是最重要的应用程序安全课题之一。

什么是外部数据？

- 来自表单的输入数据
- Cookies
- Web services data
- 服务器变量
- 数据库查询结果

函数和过滤器

如需过滤变量，请使用下面的过滤器函数之一：

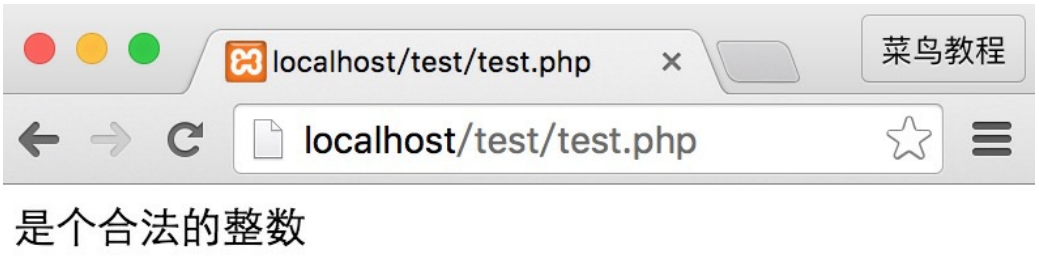
- filter_var() - 通过一个指定的过滤器来过滤单一的变量
- filter_var_array() - 通过相同的或不同的过滤器来过滤多个变量
- filter_input - 获取一个输入变量，并对它进行过滤
- filter_input_array - 获取多个输入变量，并通过相同的或不同的过滤器对它们进行过滤

在下面的实例中，我们用 filter_var() 函数验证了一个整数：

实例

<?php $int = 123; if(!filter_var($int, FILTER_VALIDATE_INT)) { echo("不是一个合法的整数"); } else { echo("是个合法的整数"); } ?>

上面的代码使用了 "FILTER_VALIDATE_INT" 过滤器来过滤变量。由于这个整数是合法的，因此上面的代码将输出：



如果我们尝试使用一个非整数的变量（比如 "123abc"），则将输出："Integer is not valid"。

如需查看完整的函数和过滤器列表，请访问我们的 PHP Filter 参考手册。

Validating 和 Sanitizing

有两种过滤器：

Validating 过滤器：

- 用于验证用户输入
- 严格的格式规则（比如 URL 或 E-Mail 验证）
- 如果成功则返回预期的类型，如果失败则返回 FALSE

Sanitizing 过滤器：

- 用于允许或禁止字符串中指定的字符
- 无数据格式规则
- 始终返回字符串

选项和标志

选项和标志用于向指定的过滤器添加额外的过滤选项。

不同的过滤器有不同的选项和标志。

在下面的实例中，我们用 filter_var() 和 "min_range" 以及 "max_range" 选项验证了一个整数：

实例

```
<?php $var=300; $int_options = array( "options"=>array ( "min_range"=>0, "max_range"=>256 ) ); if(!filter_var($var, FILTER_VALIDATE_INT, $int_options)) { echo("不是一个合法的整数"); } else { echo("是个合法的整数"); } ?>
```

就像上面的代码一样，选项必须放入一个名为 "options" 的相关数组中。如果使用标志，则不需在数组内。

由于整数是 "300"，它不在指定的范围内，以上代码的输出将是：

不是一个合法的整数

如需查看完整的函数和过滤器列表，请访问我们的 PHP Filter 参考手册。您可以看到每个过滤器的可用选项和标志。

验证输入

让我们试着验证来自表单的输入。

我们需要做的第一件事情是确认是否存在我们正在查找的输入数据。

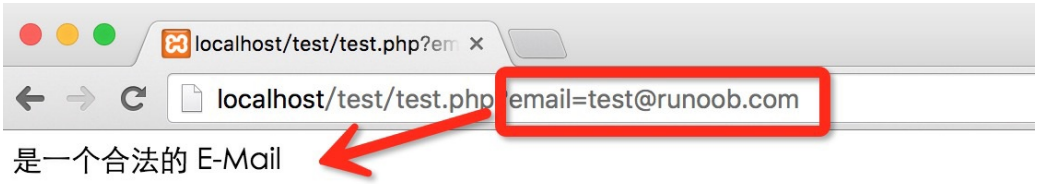
然后我们用 filter_input() 函数过滤输入的数据。

在下面的实例中，输入变量 "email" 被传到 PHP 页面：

实例

```
<?php if(!filter_has_var(INPUT_GET, "email")) { echo("没有 email 参数"); } else { if (!filter_input(INPUT_GET, "email", FILTER_VALIDATE_EMAIL)) { echo "不是一个合法的 E-Mail"; } else { echo "是一个合法的 E-Mail"; } } ?>
```

以上实例测试结果如下：



实例解释

上面的实例有一个通过 "GET" 方法传送的输入变量 (email)：

1. 检测是否存在 "GET" 类型的 "email" 输入变量
2. 如果存在输入变量，检测它是否是有效的 e-mail 地址

净化输入

让我们试着清理一下从表单传来的 URL。

首先，我们要确认是否存在我们正在查找的输入数据。

然后，我们用 filter_input() 函数来净化输入数据。

在下面的实例中，输入变量 "url" 被传到 PHP 页面：

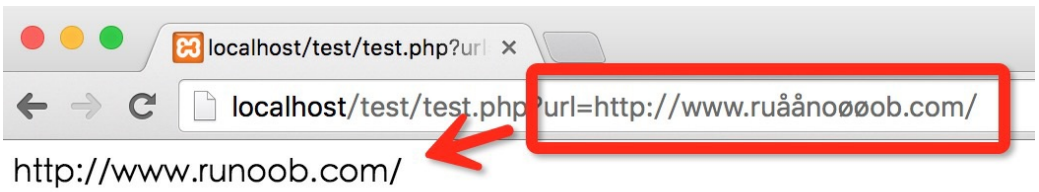
```
<?php
if(!filter_has_var(INPUT_GET, "url"))
{
 echo("没有 url 参数");
}
else
{
 $url = filter_input(INPUT_GET,
 "url", FILTER_SANITIZE_URL);
 echo $url;
}
?>
```

实例解释

上面的实例有一个通过 "GET" 方法传送的输入变量 (url)：

1. 检测是否存在 "GET" 类型的 "url" 输入变量
2. 如果存在此输入变量，对其进行净化（删除非法字符），并将其存储在 $url 变量中

假如输入变量是一个类似这样的字符串："http://www.ruåånoøøob.com"，则净化后的 $url 变量如下所示：



过滤多个输入

表单通常由多个输入字段组成。为了避免对 filter_var 或 filter_input 函数重复调用，我们可以使用 filter_var_array 或 the filter_input_array 函数。

在本例中，我们使用 filter_input_array() 函数来过滤三个 GET 变量。接收到的 GET 变量是一个名字、一个年龄以及一个 e-mail 地址：

实例

```
<?php $filters = array ( "name" => array ( "filter"=>FILTER_SANITIZE_STRING ), "age" => array ( "filter"=>FILTER_VALIDATE_INT, "options"=>array ( "min_range"=>1, "max_range"=>120 ) ), "email"=> FILTER_VALIDATE_EMAIL ); $result = filter_input_array(INPUT_GET, $filters); if (!$result["age"]) { echo("年龄必须在 1 到 120 之间。<br>"); } elseif(!$result["email"]) { echo("E-Mail 不合法<br>"); } else { echo("输入正确"); } ?>
```

实例解释

上面的实例有三个通过 "GET" 方法传送的输入变量 (name、age 和 email)：

1. 设置一个数组，其中包含了输入变量的名称和用于指定的输入变量的过滤器
2. 调用 filter_input_array() 函数，参数包括 GET 输入变量及刚才设置的数组
3. 检测 $result 变量中的 "age" 和 "email" 变量是否有非法的输入。（如果存在非法输入，在使用 filter_input_array() 函数之后，输入变量为 FALSE。）

filter_input_array() 函数的第二个参数可以是数组或单一过滤器的 ID。

如果该参数是单一过滤器的 ID，那么这个指定的过滤器会过滤输入数组中所有的值。

如果该参数是一个数组，那么此数组必须遵循下面的规则：

- 必须是一个关联数组，其中包含的输入变量是数组的键（比如 "age" 输入变量）
- 此数组的值必须是过滤器的 ID，或者是规定了过滤器、标志和选项的数组

使用 Filter Callback

通过使用 FILTER_CALLBACK 过滤器，可以调用自定义的函数。把它作为一个过滤器来使用。这样，我们就拥有了数据过滤的完全控制权。

您可以创建自己的自定义函数，也可以使用已存在的 PHP 函数。

将您准备用到的过滤器的函数，按指定选项的规定方法进行规定。在关联数组中，带有名称 "options"。

在下面的实例中，我们使用了一个自定义的函数把所有 "_" 转换为 "."：

实例

```
<?php function convertSpace($string) { return str_replace("_", ".", $string); } $string = "www_runoob_com!"; echo filter_var($string, FILTER_CALLBACK, array("options"=>"convertSpace")); ?>
```

上面代码的结果如下所示：

www.runoob.com!

实例解释

上面的实例把所有 "_" 转换成 "." ：

1. 创建一个把 "_" 替换为 "." 的函数
2. 调用 filter_var() 函数，它的参数是 FILTER_CALLBACK 过滤器以及包含我们的函数的数组

PHP MySQL 简介

通过 PHP，您可以连接和操作数据库。

MySQL 是最 PHP 配套使用的最流行的开源数据库系统。

如果想学习更多 MySQL 知识可以查看本站MySQL 教程。

MySQL 是什么?

- MySQL 是一种在 Web 上使用的数据库系统。
- MySQL 是一种在服务器上运行的数据库系统。
- MySQL 不管在小型还是大型应用程序中，都是理想的选择。
- MySQL 是非常快速，可靠，且易于使用的。
- MySQL 支持标准的 SQL。
- MySQL 在一些平台上编译。
- MySQL 是免费下载使用的。
- MySQL 是由 Oracle 公司开发、发布和支持的。
- MySQL 是以公司创始人 Monty Widenius's daughter: My 命名的。

MySQL 中的数据存储在表中。表格是一个相关数据的集合，它包含了列和行。

在分类存储信息时，数据库非常有用。一个公司的数据库可能拥有以下表：

- Employees
- Products
- Customers
- Orders

PHP + MySQL

- PHP 与 MySQL 结合是跨平台的。（您可以在 Windows 上开发，在 Unix 平台上应用。）

查询

查询是一种询问或请求。

通过 MySQL，我们可以向数据库查询具体的信息，并得到返回的记录集。

请看下面的查询（使用标准 SQL）：

```
mysql> set names utf8;
mysql> SELECT name FROM websites;
+---------------+
| name          |
+---------------+
| Google        |
| 菜鸟教程      |
| 微博          |
| Facebook      |
| stackoverflow |
+---------------+
6 rows in set (0.00 sec)
```

语句 **set names utf8;**用于设定数据库编码，让中文可以正常显示。

上面的查询选取了 "websites" 表中 "name" 列的所有数据。

如需学习更多关于 SQL 的知识，请访问我们的 SQL 教程。

下载 MySQL 数据库

如果您的 PHP 服务器没有 MySQL 数据库，可以在此免费下载 MySQL：http://www.mysql.com。

关于 MySQL 数据库的事实

关于 MySQL 的一点很棒的特性是，可以对它进行缩减，来支持嵌入的数据库应用程序。也许正因为如此，许多人认为 MySQL 仅仅能处理中小型的系统。

事实上，对于那些支持巨大数据和访问量的网站（比如 Friendster、Yahoo、Google），MySQL 是事实上的标准数据库。

这个地址提供了使用 MySQL 的公司的概览：http://www.mysql.com/customers/。

PHP 连接 MySQL

PHP 5 及以上版本建议使用以下方式连接 MySQL：

- **MySQLi extension** ("i" 意为 improved)
- **PDO (PHP Data Objects)**

在 PHP 早起版本中我们使用 MySQL 扩展。但该扩展在 2012 年开始不建议使用。

我是该用 MySQLi，还是 PDO?

如果你需要一个简短的回答，即 "你习惯哪个就用哪个"。

MySQLi 和 PDO 有它们自己的优势：

PDO 应用在 12 种不同数据库中，MySQLi 只针对 MySQL 数据库。

所以，如果你的项目需要在多种数据库中切换，建议使用 PDO，这样你只需要修改连接字符串和部分查询语句即可。使用 MySQLi, 如果不同数据库，你需要重新编写所有代码，包括查询。

两者都是面向对象, 但 MySQLi 还提供了 API 接口。

两者都支持预处理语句。预处理语句可以防止 SQL 注入，对于 web 项目的安全性是非常重要的。

MySQLi 和 PDO 连接 MySQL 实例

在本章节及接下来的章节中，我们会使用以下三种方式来演示 PHP 操作 MySQL:

- MySQLi (面向对象)
- MySQLi (面向过程)
- PDO

MySQLi 安装

Linux 和 Windows 在 php5 mysql 包安装时 MySQLi 扩展多数情况下是自动安装的。

安装详细信息，请查看： http://php.net/manual/en/mysqli.installation.php

可以通过 phpinfo() 查看是否安装成功：

mysqli

Mysqli Support	enabled
Client API library version	mysqlnd 5.0.11-dev - 20120503 - $Id: f373ea5dd5538761406a8022a4b8a374418b240e $
Active Persistent Links	0
Inactive Persistent Links	0
Active Links	0

Directive	Local Value	Master Value
mysqli.allow_local_infile	On	On
mysqli.allow_persistent	On	On
mysqli.default_host	no value	no value

PDO 安装

For 安装详细信息，请查看： http://php.net/manual/en/pdo.installation.php

可以通过 phpinfo() 查看是否安装成功：

PDO

PDO support	enabled
PDO drivers	mysql, pgsql, sqlite

pdo_mysql

PDO Driver for MySQL	enabled
Client API version	mysqlnd 5.0.11-dev - 20120503 - $Id: f373ea5dd5538761406a8022a4b8a374418b240e $

连接 MySQL

在我们访问 MySQL 数据库前，我们需要先连接到数据库服务器：

实例 (MySQLi - 面向对象)

```
<?php $servername = "localhost"; $username = "username"; $password = "password"; // 创建连接 $conn = new mysqli($servername, $username, $password); // 检测连接 if ($conn->connect_error) { die("连接失败: " . $conn->connect_error); } echo "连接成功"; ?>
```

注意在以上面向对象的实例中 $connect_error 是在 PHP 5.2.9 和 5.3.0 中添加的。如果你需要兼容更早版本 请使用以下代码替换：

```
// 检测连接
if (mysqli_connect_error()) {
    die("数据库连接失败: " . mysqli_connect_error());
}
```

实例 (MySQLi - 面向过程)

```
<?php $servername = "localhost"; $username = "username"; $password = "password"; // 创建连接 $conn = mysqli_connect($servername, $username, $password); // 检测连接 if (!$conn) { die("Connection failed: " . mysqli_connect_error()); } echo "连接成功"; ?>
```

实例 (PDO)

```
<?php $servername = "localhost"; $username = "username"; $password = "password"; try { $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password); echo "连接成功"; } catch(PDOException $e) { echo $e->getMessage(); } ?>
```

注意在以上 PDO 实例中我们已经指定了数据库 (myDB)。PDO 在连接过程需要设置数据库名。如果没有指定，则会抛出异常。

关闭连接

连接在脚本执行完后会自动关闭。你也可以使用以下代码来关闭连接：

实例 (MySQLi - 面向对象)

```
$conn->close();
```

实例 (MySQLi - 面向过程)

```
mysqli_close($conn);
```

实例 (PDO)

```
$conn = null;
```

PHP MySQL 创建数据库

数据库存有一个或多个表。

你需要 CREATE 权限来创建或删除 MySQL 数据库。

使用 MySQLi 和 PDO 创建 MySQL 数据库

CREATE DATABASE 语句用于在 MySQL 中创建数据库。

在下面的实例中，创建了一个名为 "myDB" 的数据库：

实例 (MySQLi - 面向对象)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// 创建连接
$conn = new mysqli($servername, $username, $password);
// 检测连接
if ($conn->connect_error) {
    die("连接失败: " . $conn->connect_error);
}

// 创建数据库
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "数据库创建成功";
} else {
    echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```

注意： 当你创建一个新的数据库时，你必须为 mysqli 对象指定三个参数 (servername, username 和 password)。

Tip: 如果你使用其他端口（默认为3306），为数据库参数添加空字符串，如: new mysqli("localhost", "username", "password", "", port)

实例 (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// 创建连接
$conn = mysqli_connect($servername, $username, $password);
// 检测连接
if (!$conn) {
    die("连接失败: " . mysqli_connect_error());
}

// 创建数据库
$sql = "CREATE DATABASE myDB";
if (mysqli_query($conn, $sql)) {
    echo "数据库创建成功";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

注意： 以下使用 PDO 实例创建数据库 "myDBPDO":

实例

使用 PDO：

```
<?php
$servername = "localhost";
```

```php
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password);

    // 设置 PDO 错误模式为异常
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "CREATE DATABASE myDBPDO";

    // 使用 exec() ，因为没有结果返回
    $conn->exec($sql);

    echo "数据库创建成功<br>";
}
catch(PDOException $e)
{
    echo $sql . "<br>" . $e->getMessage();
}

$conn = null;
?>
```

提示：使用 PDO 的最大好处是在数据库查询过程出现问题时可以使用异常类来 处理问题。如果 try{ } 代码块出现异常，脚本会停止执行并会跳到第一个 catch(){ } 代码块执行代码。 在以上捕获的代码块中我们输出了 SQL 语句并生成错误信息。

PHP MySQL 插入数据

使用 MySQLi 和 PDO 向 MySQL 插入数据

在创建完数据库和表后，我们可以向表中添加数据。

以下为一些语法规则：

- PHP 中 SQL 查询语句必须使用引号
- 在 SQL 查询语句中的字符串值必须加引号
- 数值的值不需要引号
- NULL 值不需要引号

INSERT INTO 语句通常用于向 MySQL 表添加新的记录：

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

学习更多关于 SQL 知识，请查看我们的 SQL 教程。

在前面的几个章节中我们已经创建了表 "MyGuests"，表字段有: "id", "firstname", "lastname", "email" 和 "reg_date"。 现在，让我们开始向表填充数据。

注意： 如果列设置 AUTO_INCREMENT (如 "id" 列) 或 TIMESTAMP (如 "reg_date" 列)，我们就不需要在 SQL 查询语句中指定值： MySQL 会自动为该列添加值。

以下实例向 "MyGuests" 表添加了新的记录:

实例 (MySQLi - 面向对象)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// 创建连接
$conn = new mysqli($servername, $username, $password, $dbname);
// 检测连接
if ($conn->connect_error) {
    die("连接失败: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "新记录插入成功";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

实例 (MySQLi - 面向过程)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// 创建连接
$conn = mysqli_connect($servername, $username, $password, $dbname);
// 检测连接
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "新记录插入成功";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

实例 (PDO)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // 设置 PDO 错误模式，用于抛出异常
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('John', 'Doe', 'john@example.com')";
    // 使用 exec() ，没有结果返回
    $conn->exec($sql);
    echo "新记录插入成功";
}
catch(PDOException $e)
{
    echo $sql . "<br>" . $e->getMessage();
}

$conn = null;
?>
```

PHP MySQL 读取数据

从 MySQL 数据库读取数据

SELECT 语句用于从数据表中读取数据:

```
SELECT column_name(s) FROM table_name
```

我们可以使用 * 号来读取所有数据表中的字段：

```
SELECT * FROM table_name
```

如需学习更多关于 SQL 的知识，请访问我们的 SQL 教程。

使用 MySQLi

以下实例中我们从 myDB 数据库的 MyGuests 表读取了 id, firstname 和 lastname 列的数据并显示在页面上:

实例 (MySQLi - 面向对象)

```php
<?php $servername = "localhost"; $username = "username"; $password = "password"; $dbname = "myDB"; // 创建连接 $conn = new mysqli($servername, $username, $password, $dbname); // Check connection if ($conn->connect_error) { die("连接失败: " . $conn->connect_error); } $sql = "SELECT id, firstname, lastname FROM MyGuests"; $result = $conn->query($sql); if ($result->num_rows > 0) { // 输出数据 while($row = $result->fetch_assoc()) { echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>"; } } else { echo "0 结果"; } $conn->close(); ?>
```

以上代码解析如下:

首先，我们设置了 SQL 语句从 MyGuests数据表中读取 id, firstname 和 lastname 三个字段。之后我们使用改 SQL 语句从数据库中取出结果集并赋给复制给变量 $result。

函数 num_rows() 判断返回的数据。

如果返回的是多条数据，函数 fetch_assoc() 将结合集放入到关联数组并循环输出。 while() 循环出结果集，并输出 id, firstname 和 lastname 三个字段值。

以下实例使用 MySQLi 面向过程的方式，效果类似以上代码:

实例 (MySQLi - 面向过程)

```php
<?php $servername = "localhost"; $username = "username"; $password = "password"; $dbname = "myDB"; // 创建连接 $conn = mysqli_connect($servername, $username, $password, $dbname); // Check connection if (!$conn) { die("连接失败: " . mysqli_connect_error()); } $sql = "SELECT id, firstname, lastname FROM MyGuests"; $result = mysqli_query($conn, $sql); if (mysqli_num_rows($result) > 0) { // 输出数据 while($row = mysqli_fetch_assoc($result)) { echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>"; } } else { echo "0 结果"; } mysqli_close($conn); ?>
```

使用 PDO (+ 预处理)

以下实例使用了预处理语句。

选取了 MyGuests 表中的 id, firstname 和 lastname 字段，并放到 HTML 表格中：

实例 (PDO)

```php
<?php echo "<table style='border: solid 1px black;'>"; echo "<tr><th>Id</th><th>Firstname</th><th>Lastname</th></tr>"; class TableRows extends RecursiveIteratorIterator { function __construct($it) { parent::__construct($it, self::LEAVES_ONLY); } function current() { return "<td style='width:150px;border:1px solid black;'>" . parent::current(). "</td>"; } function beginChildren() { echo "<tr>"; } function endChildren() { echo "</tr>" . "\n"; } } $servername = "localhost"; $username = "username"; $password = "password"; $dbname = "myDBPDO"; try { $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password); $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION); $stmt = $conn->prepare("SELECT id, firstname, lastname FROM MyGuests"); $stmt->execute(); // 设置结果集为关联数组 $result = $stmt->setFetchMode(PDO::FETCH_ASSOC); foreach(new TableRows(new RecursiveArrayIterator($stmt->fetchAll())) as $k=>$v) { echo $v; } } catch(PDOException $e) { echo "Error: " . $e->getMessage(); } $conn = null; echo "</table>"; ?>
```

PHP MySQL Where 子句

WHERE 子句用于过滤记录。

WHERE 子句

WHERE 子句用于提取满足指定标准的的记录。

语法

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

如需学习更多关于 SQL 的知识，请访问我们的 SQL 教程。

为了让 PHP 执行上面的语句，我们必须使用 mysqli_query() 函数。该函数用于向 MySQL 连接发送查询或命令。

实例

下面的实例将从 "Persons" 表中选取所有 FirstName='Peter' 的行：

```
<?php
$con=mysqli_connect("localhost","username","password","database");
// 检测连接
if (mysqli_connect_errno())
{
 echo "连接失败: " . mysqli_connect_error();
}

$result = mysqli_query($con,"SELECT * FROM Persons
WHERE FirstName='Peter'");

while($row = mysqli_fetch_array($result))
{
 echo $row['FirstName'] . " " . $row['LastName'];
 echo "<br>";
}
?>
```

以上代码将输出：

```
Peter Griffin
```

PHP MySQL Order By 关键词

ORDER BY 关键词用于对记录集中的数据进行排序。

ORDER BY 关键词

ORDER BY 关键词用于对记录集中的数据进行排序。

ORDER BY 关键词默认对记录进行升序排序。

如果你想降序排序，请使用 DESC 关键字。

语法

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

如需学习更多关于 SQL 的知识，请访问我们的 SQL 教程。

实例

下面的实例选取 "Persons" 表中存储的所有数据，并根据 "Age" 列对结果进行排序：

```
<?php
$con=mysqli_connect("localhost","username","password","database");
// 检测连接
if (mysqli_connect_errno())
{
 echo "连接失败: " . mysqli_connect_error();
}

$result = mysqli_query($con,"SELECT * FROM Persons ORDER BY age");

while($row = mysqli_fetch_array($result))
{
 echo $row['FirstName'];
 echo " " . $row['LastName'];
 echo " " . $row['Age'];
 echo "<br>";
}

mysqli_close($con);
?>
```

以上结果将输出：

```
Glenn Quagmire 33
Peter Griffin 35
```

根据两列进行排序

可以根据多个列进行排序。当按照多个列进行排序时，只有第一列的值相同时才使用第二列：

```
SELECT column_name(s)
FROM table_name
ORDER BY column1, column2
```

PHP MySQL Update

UPDATE 语句用于中修改数据库表中的数据。

更新数据库中的数据

UPDATE 语句用于更新数据库表中已存在的记录。

语法

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

注释：请注意 UPDATE 语法中的 WHERE 子句。WHERE 子句规定了哪些记录需要更新。如果您想省去 WHERE 子句，所有的记录都会被更新！

如需学习更多关于 SQL 的知识，请访问我们的 SQL 教程。

为了让 PHP 执行上面的语句，我们必须使用 mysqli_query() 函数。该函数用于向 MySQL 连接发送查询或命令。

实例

在本教程的前面章节中，我们创建了一个名为 "Persons" 的表，如下所示：

FirstName	LastName	Age
Peter	Griffin	35
Glenn	Quagmire	33

下面的例子更新 "Persons" 表的一些数据：

```
<?php
$con=mysqli_connect("localhost","username","password","database");
// 检测连接
if (mysqli_connect_errno())
{
 echo "连接失败: " . mysqli_connect_error();
}

mysqli_query($con,"UPDATE Persons SET Age=36
WHERE FirstName='Peter' AND LastName='Griffin'");

mysqli_close($con);
?>
```

在这次更新后，"Persons" 表如下所示：

FirstName	LastName	Age
Peter	Griffin	36
Glenn	Quagmire	33

PHP MySQL Delete

DELETE è¯å¥ç¨äºä»æ°æ®åºè¡¨ä¸å é¤è¡ã

å é¤æ°æ®åºä¸çæ°æ®

DELETE FROM è¯å¥ç¨äºä»æ°æ®åºè¡¨ä¸å é¤è®°å½ã

è¯æ³

```
DELETE FROM table_name
WHERE some_column = some_value
```

æ³¨éï¼è¯·æ³¨æ DELETE è¯æ³ä¸ç WHERE åå¥ãWHERE åå¥è§å®äºåªäºè®°å½éè¦å é¤ãå¦ææ¨æ³çç¥ WHERE åå¥ï¼æææè®°å½é½ä¼è¢«å é¤ï¼

å¦éå¦ä¹ æ´å¤å³äº SQL çç¥è¯ï¼è¯·è®¿é®æä»¬ç SQL æç¨ã

ä¸ºäºè®© PHP æ§è¡ä¸é¢çè¯å¥ï¼æä»¬å¿é¡»ä½¿ç¨ mysqli_query() å½æ°ãè¯¥å½æ°ç¨äºå MySQL è¿æ¥åéæ¥è¯¢æå½ä»¤ã

å®ä¾

è¯·çä¸é¢ç "Persons" è¡¨ï¼

FirstName	LastName	Age
Peter	Griffin	35
Glenn	Quagmire	33

ä¸é¢çå®ä¾å é¤ "Persons" è¡¨ä¸æææ LastName='Griffin' çè®°å½ï¼

```
<?php
$con=mysqli_connect("localhost","username","password","database");
// æ£æµè¿æ¥
if (mysqli_connect_errno())
{
 echo "è¿æ¥å¤±è´¥: " . mysqli_connect_error();
}

mysqli_query($con,"DELETE FROM Persons WHERE LastName='Griffin'");

mysqli_close($con);
?>
```

å¨è¿æ¬¡å é¤åï¼"Persons" è¡¨å¦ä¸æç¤ºï¼

FirstName	LastName	Age
Glenn	Quagmire	33

PHP 数据库 ODBC

ODBC 是一种应用程序编程接口（Application Programming Interface，API），使我们有能力连接到某个数据源（比如一个 MS Access 数据库）。

创建 ODBC 连接

通过一个 ODBC 连接，您可以连接到您的网络中的任何计算机上的任何数据库，只要 ODBC 连接是可用的。

这是创建到达 MS Access 数据库的 ODBC 连接的方法：

1. 在控制面板中打开管理工具图标。
2. 双击其中的数据源**(ODBC)**图标。
3. 选择系统 **DSN** 选项卡。
4. 点击系统 DSN 选项卡中的添加。
5. 选择**Microsoft Access Driver**。点击完成。
6. 在下一个界面，点击**选择**来定位数据库。
7. 为数据库起一个**数据源名(DSN)**。
8. 点击**确定**。

请注意，必须在您的网站所在的计算机上完成这个配置。如果您的计算机上正在运行 Internet 信息服务(IIS)，上面的指令将会生效，但是如果您的网站位于远程服务器，您必须拥有对该服务器的物理访问权限，或者请您的主机提供商为您建立 DSN。

连接到 ODBC

odbc_connect() 函数用于连接到 ODBC 数据源。该函数有四个参数：数据源名、用户名、密码以及可选的指针类型。

odbc_exec() 函数用于执行 SQL 语句。

实例

下面的实例创建了到达名为 northwind 的 DSN 的连接，没有用户名和密码。然后创建并执行一条 SQL 语句：

```
$conn=odbc_connect('northwind','','');
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
```

取回记录

odbc_fetch_row() 函数用于从结果集中返回记录。如果能够返回行，则函数返回 true，否则返回 false。

该函数有两个参数：ODBC 结果标识符和可选的行号：

```
odbc_fetch_row($rs)
```

从记录中取回字段

odbc_result() 函数用于从记录中读取字段。该函数有两个参数：ODBC 结果标识符和字段编号或名称。

下面的代码行从记录中返回第一个字段的值：

```
$compname=odbc_result($rs,1);
```

下面的代码行返回名为 "CompanyName" 的字段的值：

```
$compname=odbc_result($rs,"CompanyName");
```

关闭 ODBC 连接

odbc_close() 函数用于关闭 ODBC 连接。

```
odbc_close($conn);
```

ODBC 实例

下面的实例展示了如何首先创建一个数据库连接，接着创建一个结果集，然后在 HTML 表格中显示数据。

```
<html>
<body>

<?php
$conn=odbc_connect('northwind','','');
if (!$conn)
{
 exit("连接失败: " . $conn);
}

$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);

if (!$rs)
{
 exit("SQL 语句错误");
}
echo "<table><tr>";
echo "<th>Companyname</th>";
echo "<th>Contactname</th></tr>";

while (odbc_fetch_row($rs))
{
 $compname=odbc_result($rs,"CompanyName");
 $conname=odbc_result($rs,"ContactName");
 echo "<tr><td>$compname</td>";
 echo "<td>$conname</td></tr>";
}
odbc_close($conn);
echo "</table>";
?>

</body>
</html>
```

PHP XML Expat è§£æå¨

åå»ºç Expat è§£æå¨ä½¿å¨ PHP ä¸å¤ç XML ææ¡£æä¸ºå¯è½ã

XML æ¯ä»ä¹ï¼

XML ç¨äºæè¿°æ°æ®ï¼å¶ç¦ç¹æ¯æ°æ®æ¯ä»ä¹ãXML æä»¶æè¿°äºæ°æ®çç»æã

å¨ XML ä¸ï¼æ²¡æé¢å®ä¹çæ ç¾ãæ¨å¿é¡»å®ä¹èªå·±çæ ç¾ã

å¦éå¦ä¹ æ´å¤å³äº XML çç¥è¯ï¼è¯·è®¿é®æä»¬ç [XML æç¨]ã

Expat æ¯ä»ä¹ï¼

å¦éè¯»åå´æ° - åå»ºååç - ä¸ä¸ª XML ææ¡£ï¼æ¨éè¦ XML è§£æå¨ã

æä¸¤ç§åºæ¬ç XML è§£æå¨ç±»åï¼

- åºäºæ çè§£æå¨ï¼è¿ç§è§£æå¨æ XML ææ¡£è½¬æ¢ä¸ºæ åç»æãå®åæ´ç¯æææ¡£ï¼å¹¶æä¾å¯¹æ ä¸åç´ çè®¿é®ï¼ä¾å¦ææ¡£å¯¹è±¡æ¨¡å (DOM)ã
- åºäºäºä»¶çè§£æå¨ï¼å° XML ææ¡£è§ä¸ºä¸ç³»åçäºä»¶ãå½æä¸ªå·ä½çäºä»¶åçæ¶ï¼è§£æå¨ä¼è°ç¨å½æ°æ¥å¤çã

Expat è§£æå¨æ¯åºäºäºä»¶çè§£æå¨ã

åºäºäºä»¶çè§£æå¨éä¸å¨ XML ææ¡£çåå®¹ï¼èä¸æ¯å®ä»¬çç»æãæ£å ä¸ºå¦æ¤ï¼åºäºäºä»¶çè§£æå¨è½å¤æ¯åºäºæ çè§£æå¨æ´å¿«å°è®¿é®æ°æ®ã

è¯·çä¸é¢ç XML çæ®µï¼

<from>Jani</from>

åºäºäºä»¶çè§£æå¨æä¸é¢ç XML æ¥åä¸ºä¸è¿ä¸²ä¸ä¸ªäºä»¶ï¼

- å¼å§åç´ ï¼from
- å¼å§ CDATA é¨åï¼å¼ï¼Jani
- å³éåç´ ï¼from

ä¸é¢ç XML å®ä¾åå«äºå½¢å¼è¯å¥½ç XMLãä¸è¿ï¼è¿ä¸ªå®ä¾æ¯æ æç XMLï¼å ä¸ºæ²¡æä¸å®å³èçææ¡£ç±»åå£°æ (DTD)ã

ç¶èï¼å¨ä½¿ç¨ Expat è§£æå¨æ¶ï¼è¿æ²¡æåºå«ãExpat æ¯ä¸æ£æ¥æææ§çè§£æå¨ï¼å¿½ç¥ä»»ä½ DTDã

ä½ä¸ºä¸ä¸ªæ¬¾åºäºäºä»¶ãééªè¯ç XML è§£æå¨ï¼Expat å¿«éä¸è½»å·§ï¼èéå PHP ç Web åºç¨ç¨åºã

æ³¨éï¼XML ææ¡£å¿é¡»å½¢å¼è¯å¥½ï¼å¦å Expat ä¼çæéè¯¯ã

å®è£

XML Expat è§£æå¨å½æ°æ¯ PHP æ ¸å¿çç»æé¨åãæ éå®è£å°±å¯ä»¥ä½¿ç¨è¿äºå½æ°ã

XML æä»¶

ä¸é¢ç XML æä»¶å°åºç¨å¨æä»¬çå®ä¾ä¸ï¼

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

åå§å XML è§£æå¨

æä»¬è¦å¨ PHP ä¸åå§å XML è§£æå¨ï¼ä¸ºä¸äº XML äºä»¶å®ä¹å¤çå¨ï¼ç¶åè§£æè¿ä¸ª XML æä»¶ã

å®ä¾

```
<?php
//Initialize the XML parser
$parser=xml_parser_create();

//Function to use at the start of an element
function start($parser,$element_name,$element_attrs)
{
switch($element_name)
{
case "NOTE":
echo "-- Note --<br>";
break;
case "TO":
echo "To: ";
break;
case "FROM":
echo "From: ";
break;
case "HEADING":
echo "Heading: ";
break;
case "BODY":
echo "Message: ";
}
}

//Function to use at the end of an element
function stop($parser,$element_name)
{
echo "<br>";
}

//Function to use when finding character data
function char($parser,$data)
{
echo $data;
}

//Specify element handler
xml_set_element_handler($parser,"start","stop");

//Specify data handler
xml_set_character_data_handler($parser,"char");

//Open XML file
$fp=fopen("test.xml","r");

//Read data
while ($data=fread($fp,4096))
{
xml_parse($parser,$data,feof($fp)) or
die (sprintf("XML Error: %s at line %d",
xml_error_string(xml_get_error_code($parser)),
xml_get_current_line_number($parser)));
}

//Free the XML parser
xml_parser_free($parser);
?>
```

ä»¥ä¸ä»£ç å°è¾åºï¼

-- Note --
To: Tove
From: Jani
Heading: Reminder
Message: Don't forget me this weekend!

å·¥ä½åçï¼

1. éè¿ xml_parser_create() å½æ°åå§å XML è§£æå¨
2. åå»ºéåä¸åäº‹ä»¶å¤çç¨åºçå½æ°
3. æ·»å xml_set_element_handler() å½æ°æ¥å®ä¹ï¼å½è§£æå¨éå°å¼å§åç»ææ ç¾æ¶æ§è¡åªä¸ªå½æ°
4. æ·»å xml_set_character_data_handler() å½æ°æ¥å®ä¹ï¼å½è§£æå¨éå°åç¬¦æ°æ®æ¶æ§è¡åªä¸ªå½æ°
5. éè¿ xml_parse() å½æ°æ¥è§£ææä»¶ "test.xml"
6. ä¸ä¸æéè¯¯çè¯ï¼æ·»å xml_error_string() å½æ°æ XML éè¯¯è½¬æ¢ä¸ºææ¬è¯´æ
7. è°ç¨ xml_parser_free() å½æ°æ¥éæ¾åéç» xml_parser_create() å½æ°çå

æ´å¤ PHP Expat è§£æå¨çä¿¡æ¯

å¦éäºè§£æ´å¤å³äº PHP Expat å½æ°çä¿¡æ¯ï¼è¯·è®¿é®æä»¬ç PHP XML Parser åèæåã

PHP XML DOM

内建的 DOM 解析器使在 PHP 中处理 XML 文档成为可能。

DOM 是什么?

W3C DOM 提供了针对 HTML 和 XML 文档的标准对象集，以及用于访问和操作这些文档的标准接口。

W3C DOM 被分为不同的部分（Core, XML 和 HTML）和不同的级别（DOM Level 1/2/3）：

* Core DOM - 为任何结构化文档定义标准的对象集
* XML DOM - 为 XML 文档定义标准的对象集
* HTML DOM - 为 HTML 文档定义标准的对象集

如需学习更多关于 XML DOM 的知识，请访问我们的 XML DOM 教程。

XML 解析

如需读取和更新 - 创建和处理 - 一个 XML 文档，您需要 XML 解析器。

有两种基本的 XML 解析器类型:

- 基于树的解析器：这种解析器把 XML 文档转换为树型结构。它分析整篇文档，并提供了对树中元素的访问，例如文档对象模型 (DOM)。
- 基于时间的解析器：将 XML 文档视为一系列的事件。当某个具体的事件发生时，解析器会调用函数来处理。

DOM 解析器是基于树的解析器。

请看下面的 XML 文档片段:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<from>Jani</from>
```

XML DOM 把上面的 XML 视为一个树形结构:

- Level 1: XML 文档
- Level 2: 根元素： <from>
- Level 3: 文本元素： "Jani"

安装

DOM XML 解析器函数是 PHP 核心的组成部分。无需安装就可以使用这些函数。

XML 文件

下面的 XML 文件将应用在我们的实例中:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

加载和输出 XML

我们需要初始化 XML 解析器，加载 XML，并把它输出:

实例

```
<?php
$xmlDoc = new DOMDocument();
$xmlDoc->load("note.xml");

print $xmlDoc->saveXML();
?>
```

以上代码将输出:

ToveJaniReminder Don't forget me this weekend!

如果您在浏览器窗口中查看源代码，会看到下面的 HTML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

上面的实例创建了一个 DOMDocument-Object，并把 "note.xml" 中的 XML 载入这个文档对象中。

saveXML() 函数把内部 XML 文档放入一个字符串，这样我们就可以输出它。

遍历 XML

我们要初始化 XML 解析器，加载 XML，并遍历 <note> 元素的所有元素:

实例

```
<?php
$xmlDoc = new DOMDocument();
$xmlDoc->load("note.xml");

$x = $xmlDoc->documentElement;
foreach ($x->childNodes AS $item)
{
print $item->nodeName . " = " . $item->nodeValue . "<br>";
}
?>
```

以上代码将输出:

```
#text =
to = Tove
#text =
from = Jani
#text =
heading = Reminder
#text =
body = Don't forget me this weekend!
#text =
```

在上面的实例中，您看到了每个元素之间存在空的文本节点。

当 XML 生成时，它通常会在节点之间包含空白。XML DOM 解析器把它们当作普通的元素，如果您不注意它们，有时会产生问题。

如需学习更多关于 XML DOM 的知识，请访问我们的 XML DOM 教程。

PHP SimpleXML

PHP SimpleXML 处理最普通的 XML 任务，其余的任务则交由其它扩展处理。

什么是 PHP SimpleXML?

SimpleXML 是 PHP 5 中的新特性。

SimpleXML 扩展提供了一种获取 XML 元素的名称和文本的简单方式。

与 DOM 或 Expat 解析器相比，SimpleXML 仅仅用几行代码就可以从 XML 元素中读取文本数据。

SimpleXML 可把 XML 文档（或 XML 字符串）转换为对象，比如:

- 元素被转换为 SimpleXMLElement 对象的单一属性。当同一级别上存在多个元素时，它们会被置于数组中。
- 属性通过使用关联数组进行访问，其中的索引对应属性名称。
- 元素内部的文本被转换为字符串。如果一个元素拥有多个文本节点，则按照它们被找到的顺序进行排列。

当执行类似下列的基础任务时，SimpleXML 使用起来非常快捷:

- 读取/提取 XML 文件/字符串的数据
- 编辑文本节点或属性

然而，在处理高级 XML 时，比如命名空间，最好使用 Expat 解析器或 XML DOM。

安装

从 PHP 5 开始，SimpleXML 函数是 PHP 核心的组成部分。无需安装就可以使用这些函数。

PHP SimpleXML 实例

假设我们有如下的 XML 文件，"note.xml":

<?xml version="1.0" encoding="ISO-8859-1"?> <note> <to>Tove</to> <from>Jani</from> <heading>Reminder</heading> <body>Don't forget me this weekend!</body> </note>

现在我们想要输出上面的 XML 文件的不同信息:

实例 1

输出 $xml 变量（是 SimpleXMLElement 对象）的键和元素:

```
<?php $xml=simplexml_load_file("note.xml"); print_r($xml); ?>
```

运行实例 »

以上代码将输出:

```
SimpleXMLElement Object ( [to] => Tove [from] => Jani [heading] => Reminder [body] => Don't forget me this weekend! )
```

实例 2

输出 XML 文件中每个元素的数据：

<?php $xml=simplexml_load_file("note.xml"); echo $xml->to . "
"; echo $xml->from . "
"; echo $xml->heading . "
"; echo $xml->body; ?>

[运行实例 »]

以上代码将输出：

```
Tove
Jani
Reminder
Don't forget me this weekend!
```

实例 3

输出每个子节点的元素名称和数据：

<?php $xml=simplexml_load_file("note.xml"); echo $xml->getName() . "
"; foreach($xml->children() as $child) { echo $child->getName() . ": " . $child . "
"; } ?>

[运行实例 »]

以上代码将输出：

```
note
to: Tove
from: Jani
heading: Reminder
body: Don't forget me this weekend!
```

更多 PHP SimpleXML 的信息

如需了解更多关于 PHP SimpleXML 函数的信息，请访问我们的 [PHP SimpleXML 参考手册]。

AJAX ç®ä»

AJAX æ¯ä¸ç§å¨æ éé¢æ°å è½½æ´ä¸ªç½é¡µçæåµä¸ï¼è½å¤æ´æ°é¨åç½é¡µçææ¯ã

AJAX æ¯ä»ä¹ï¼

AJAX = Asynchronous JavaScript and XML.

AJAX æ¯ä¸ç§ç¨äºåå»ºå¿«éå¨æç½é¡µçææ¯ã

AJAX éè¿å¨åå°ä¸æå¡å¨è¿è¡å°éæ°æ®äº¤æ¢ï¼ä½¿ç½é¡µå®ç°å¼æ¥æ´æ°ãè¿æå³çå¯ä»¥å¨ä¸éè½½æ´ä¸ªé¡µé¢çæåµä¸ï¼å¯¹ç½é¡µçæé¨åè¿è¡æ´æ°ã

ä¼ ç»çç½é¡µï¼ä¸ä½¿ç¨ AJAXï¼å¦æéè¦æ´æ°åå®¹ï¼å¿é¡»éè½½æ´ä¸ªé¡µé¢ã

æå¾å¤ä½¿ç¨ AJAX çåºç¨ç¨åºæ¡ä¾ï¼Google MapsãGmailãYoutube å Facebookã

AJAX å¦ä½å·¥ä½

AJAX

AJAX åºäºå ç¹ç½æ å

AJAX åºäºå ç¹ç½æ åï¼å¹¶ä½¿ç¨ä»¥ä¸ææ¯ç»åï¼

- XMLHttpRequest å¯¹è±¡ï¼ä¸æå¡å¨å¼æ¥äº¤äºæ°æ®ï¼
- JavaScript/DOMï¼æ¾ç¤º/åå¤çä¿¡æ¯ï¼
- CSSï¼è®¾ç½®æ°æ®çæ ·å¼ï¼
- XMLï¼å¸¸ç¨ä½æ°æ®ä¼ è¾çæ ¼å¼ï¼

AJAX åºç¨ç¨åºä¸æµè§å¨åå¹³å°æ å³çï¼

è°·ææç´¢å»ºè®®ï¼Google Suggestï¼

éçè°·ææç´¢å»ºè®®åè½½å¨ 2005 çåå¸ï¼AJAX å¼å§æµè¡èµ·æ¥ã

[è°·ææç´¢å»ºè®®ï¼Google Suggestï¼] ä½¿ç¨ AJAX åé å»ºä¸ä¸ªé«åº¦å¨æç web çé¢ï¼å½æ¨å¨è°·æçæç´¢æ¡ä¸è¾å¥å³é®åæ¶ï¼JavaScript ä¼æåç¬¦åéå°æå¡å¨ï¼æå¡å¨ä¼è¿åä¸ä¸ªå»ºè®®åè¡¨ã

ä»å¤©å°±å¼å§ä½¿ç¨ AJAX

å¨æä»¬ PHP æç¨ä¸ï¼æä»¬å°æ¼ç¤º AJAX å¦ä½å¨ä¸éè½½æ´ä¸ªé¡µé¢çæåµä¸å¯¹ç½é¡µçæé¨åè¿è¡æ´æ°ãæå¡å¨èæ¬å°é¨ç¨ PHP æ¥ç¼åã

å¦ææ¨æ³è¦å¦ä¹ æ´å¤å³äº AJAX çç¥è¯ï¼è¯·è®¿é®æä»¬ [AJAX æç¨]ã

PHP - AJAX ä¸ PHP

AJAX è¢«ç¨äºåå»ºäº¤äºæ§æ´å¼ºçåºç¨ç¨åºã

AJAX PHP å®ä¾

ä¸é¢çå®ä¾æ¼ç¤ºäºå½ç¨æ·å¨è¾å¥æ¡ä¸é®å¥åç¬¦æ¶ï¼ç½é¡µå¦ä½ä¸ Web æå¡å¨è¿è¡éä¿¡ï¼

å®ä¾

å¨è¾å¥æ¡ä¸è¾å¥ä¸ä¸ªååï¼å¦ï¼Anna:

å§å:

è¿åå¼:

å®ä¾è§£é - HTML é¡µé¢

å½ç¨æ·å¨ä¸é¢çè¾å¥æ¡ä¸é®å¥åç¬¦æ¶ï¼ä¼æ§è¡ "showHint()" å½æ°ãè¯¥å½æ°ç± "onkeyup" äºä»¶è§¦åï¼

```
<html>
<head>
<script>
function showHint(str)
{
 if (str.length==0)
 {
  document.getElementById("txtHint").innerHTML="";
  return;
 }
 if (window.XMLHttpRequest)
 {
  // IE7+, Firefox, Chrome, Opera, Safari æµè§å¨æ§è¡ä»£ç
  xmlhttp=new XMLHttpRequest();
 }
 else
 {
  //IE6, IE5 æµè§å¨æ§è¡ä»£ç
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
 xmlhttp.onreadystatechange=function()
 {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {
   document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
  }
 }
 xmlhttp.open("GET","gethint.php?q="+str,true);
 xmlhttp.send();
}
</script>
</head>
<body>

<p><b>å¨è¾å¥æ¡ä¸è¾å¥ä¸ä¸ªåå:</b></p>
<form>
å§å: <input type="text" onkeyup="showHint(this.value)">
</form>
<p>è¿åå¼: <span id="txtHint"></span></p>

</body>
</html>
```

æºä»£ç è§£éï¼

å¦æè¾å¥æ¡æ¯ç©ºçï¼str.length==0ï¼ï¼è¯¥å½æ°ä¼æ¸ç©º txtHint å ä½ç¬¦çåå®¹ï¼å¹¶éåºè¯¥å½æ°ã

å¦æè¾å¥æ¡ä¸æ¯ç©ºçï¼é£ä¹ showHint() ä¼æ§è¡ä»¥ä¸æ¥éª¤ï¼

- åå»º XMLHttpRequest å¯¹è±¡
- åå»ºå¨æå¡å¨åºå°±ç»ªæ¶æ§è¡çå½æ°
- åæå¡å¨ä¸çæä»¶åéè¯·æ±
- è¯·æ³¨ææ·»å å° URL æ«ç«¯çåæ°ï¼qï¼ï¼åå«è¾å¥æ¡çåå®¹ï¼

PHP æä»¶

ä¸é¢éè¿ä¸é¢ç JavaScript è°ç¨çæå¡å¨é¡µé¢æ¯åä¸º "gethint.php" ç PHP æä»¶ã

"gethint.php" ä¸çä»£ç ä¼æ£æ¥ååæ°ç»ï¼ç¶åå¯¹åºçååè¿åç»æµè§å¨ï¼

```
<?php
// å°ååå¡«åæ°ç»ä¸
$a[]="Anna";
$a[]="Brittany";
$a[]="Cinderella";
$a[]="Diana";
$a[]="Eva";
$a[]="Fiona";
$a[]="Gunda";
$a[]="Hege";
$a[]="Inga";
$a[]="Johanna";
$a[]="Kitty";
$a[]="Linda";
$a[]="Nina";
$a[]="Ophelia";
$a[]="Petunia";
$a[]="Amanda";
$a[]="Raquel";
$a[]="Cindy";
$a[]="Doris";
$a[]="Eve";
$a[]="Evita";
$a[]="Sunniva";
$a[]="Tove";
$a[]="Unni";
$a[]="Violet";
$a[]="Liza";
$a[]="Elizabeth";
$a[]="Ellen";
$a[]="Wenche";
$a[]="Vicky";

//ä»è¯·æ±URLå°åä¸è·å q åæ°
$q=$_GET["q"];

//æ¥æ¾æ¯å¦ç±å¹éå¼ï¼ å¦æ q>0
if (strlen($q) > 0)
{
 $hint="";
```

```
 for($i=0; $i<count($a); $i++)
 {
  if (strtolower($q)==strtolower(substr($a[$i],0,strlen($q))))
  {
   if ($hint=="")
   {
    $hint=$a[$i];
   }
   else
   {
    $hint=$hint." , ".$a[$i];
   }
  }
 }
}

// å¦æ²¡æå¹éå¼è®¾ç½®è¾åºä¸º "no suggestion"
if ($hint == "")
{
 $response="no suggestion";
}
else
{
 $response=$hint;
}

//è¾åºè¿åå¼
echo $response;
?>
```

è§£éï¼å¦æ JavaScript åéäº»»ä½æ¬ï¼å³ strlen($q) > 0ï¼ï¼åï¼

1. æ¥æ¾å¹é JavaScript åéçå§å
2. å¦ææ¾å°å¹éï¼åå°åç¬¦ä¸²è®¾ç½®ä¸º "no suggestion"
3. å¦ææ¾å°ä¸ä¸ªæå¤ä¸ªå¹éåï¼åç¨æå§åè®¾ç½®åå°åç¬¦ä¸²
4. æåºåéå "txtHint" å ä½ç¬¦

PHP Ajax è·¨åé®é¢è§£å³æ¹æ¡

å¦æä½ çå¼æ¥è¯·æ±éè¦è·¨åå¯ä»¥æ¥çï¼PHP Ajax è·¨åé®é¢è§£å³æ¹æ¡ã

PHP - AJAX ä¸ MySQL

AJAX å¯ç¨æ¥ä¸æ°æ®åºè¿è¡äº¤äºå¼éä¿¡ã

AJAX æ°æ®åºå®ä¾

ä¸é¢çå®ä¾å°æ¼ç¤ºç½é¡µå¦ä½éè¿ AJAX ä»æ°æ®åºè¯»åä¿¡æ¯ï¼

æ¬æç¨ä½¿ç¨å°ç Websites è¡¨ SQL æä»¶ï¼websites.sqlã

å®ä¾

 éæ©ä¸ä¸ªç½ç«:

éæ©å¹¶éé¡¹ï¼ç¨æ·ä¿¡æ¯å°æ¾ç¤ºå¨è¿éâ¦

å®ä¾è§£é - MySQL æ°æ®åº

å¨ä¸é¢çå®ä¾ä¸ï¼æä»¬ä½¿ç¨çæ°æ®åºè¡¨å¦ä¸æç¤ºï¼

```
mysql> select * from websites;
+----+--------------+---------------------------+-------+---------+
| id | name         | url                       | alexa | country |
+----+--------------+---------------------------+-------+---------+
| 1  | Google       | https://www.google.cm/    | 1     | USA     |
| 2  | æ·å®       | https://www.taobao.com/   | 13    | CN      |
| 3  | èé¸æç¨ | http://www.runoob.com/    | 4689  | CN      |
| 4  | å¾®å       | http://weibo.com/         | 20    | CN      |
| 5  | Facebook     | https://www.facebook.com/ | 3     | USA     |
+----+--------------+---------------------------+-------+---------+
5 rows in set (0.01 sec)
```

å®ä¾è§£é - HTML é¡µé¢

å½ç¨æ·å¨ä¸é¢çä¸æåè¡¨ä¸éæ©æä½ç¨æ·æ¶ï¼ä¼æ§è¡åä¸º "showSite()" çå½æ°ãè¯¥å½æ°ç± "onchange" äºä»¶è§¦åï¼

test.html æä»¶ä»£ç :

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>èé¸æç¨(runoob.com)</title> <script>
function showSite(str) { if (str=="") { document.getElementById("txtHint").innerHTML=""; return; } if (window.XMLHttpRequest) { // IE7+, Firefox, Chrome, Opera, Safari æµè§å¨æ§è¡ä»£ç  xmlhttp=new XMLHttpRequest(); } else { // IE6, IE5 æµè§å¨æ§è¡ä»£ç  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP"); } xmlhttp.onreadystatechange=function() { if (xmlhttp.readyState==4 && xmlhttp.status==200) {
document.getElementById("txtHint").innerHTML=xmlhttp.responseText; } } xmlhttp.open("GET","getsite_mysql.php?q="+str,true); xmlhttp.send(); }
</script> </head> <body> <form> <select name="users" onchange="showSite(this.value)"> <option value="">éæ©ä¸ä¸ªç½ç«:</option> <option value="1">Google</option> <option value="2">æ·å®</option> <option value="3">èé¸æç¨</option> <option value="4">å¾®å</option> <option value="5">Facebook</option> </select> </form> <br> <div id="txtHint"><b>ç½ç«ä¿¡æ¯æ¾ç¤ºå¨è¿éâ¦</b></div> </body> </html>
```

showSite() å½æ°ä¼æ§è¡ä»¥ä¸æ¥éª¤ï¼

- æ£æ¥æ¯å¦æç½ç«è¢«éæ©
- åå»º XMLHttpRequest å¯¹è±¡
- åå»ºå¨æå¡å¨ååºå°±ç»ªæ¶æ§è¡çå½æ°
- åæå¡å¨ä¸çæä»¶åéè¯·æ±
- è¯·æ³¨æ·»å å° URL æ«ç«¯çåæ°ï¼qï¼ï¼åå«ä¸æåè¡¨çåå®¹ï¼

PHP æä»¶

ä¸é¢è¿æ®µéè¿ JavaScript è°ç¨çæå¡å¨é¡µé¢æ¯åä¸º "getsite_mysql.php" ç PHP æä»¶ã

"getsite_mysql.php" ä¸çæºä»£ç ä¼è¿è¡ä¸æ¬¡éå¯¹ MySQL æ°æ®åºçæ¥è¯¢ï¼ç¶åå¨ HTML è¡¨æ ¼ä¸è¿åç»æï¼

getsite_mysql.php æä»¶ä»£ç :

```
<?php $q = isset($_GET["q"]) ? intval($_GET["q"]) : ''; if(empty($q)) { echo 'è¯·éæ©ä¸ä¸ªç½ç«'; exit; } $con = mysqli_connect('localhost','root','123456'); if (!$con) { die('Could not connect: ' . mysqli_error($con)); } // éæ©æ°æ®åº mysqli_select_db($con,"test"); // è®¾ç½®ç¼ç ï¼é²æ¢ä¸æä¹±ç  mysqli_set_charset($con, "utf8"); $sql="SELECT * FROM Websites WHERE id = '".$q."'"; $result = mysqli_query($con,$sql); echo "<table
border='1'> <tr> <th>ID</th> <th>ç½ç«å</th> <th>ç½ç« URL</th> <th>Alexa æå</th> <th>å½å®¶</th> </tr>"; while($row = mysqli_fetch_array($result)) { echo "<tr>"; echo "<td>" . $row['id'] . "</td>"; echo "<td>" . $row['name'] . "</td>"; echo "<td>" . $row['url'] . "</td>"; echo "<td>" . $row['alexa'] . "</td>"; echo "<td>" . $row['country'] . "</td>"; echo "</tr>"; } echo "</table>"; mysqli_close($con); ?>
```

è§£éï¼å½æ¥è¯¢ä» JavaScript åéå° PHP æä»¶æ¶ï¼å°åçï¼

1. PHP æå¼ä¸ä¸ªå° MySQL æ°æ®åºçè¿æ¥
2. æ¾å°éä¸çç¨æ·
3. åå»º HTML è¡¨æ ¼ï¼å¡«åæ°æ®ï¼å¹¶åé "txtHint" å ä½ç¬¦

PHP 实例 - AJAX 与 XML

AJAX 可用来与 XML 文件进行交互式通信。

AJAX XML 实例

下面的实例将演示网页如何通过 AJAX 从 XML 文件读取信息：

实例

 Select a CD:

CD info will be listed here...

实例解释 - HTML 页面

当用户在上面的下拉列表中选择某张 CD 时，会执行名为 "showCD()" 的函数。该函数由 "onchange" 事件触发：

```
<html>
<head>
<script>
function showCD(str)
{
 if (str=="")
 {
  document.getElementById("txtHint").innerHTML="";
  return;
 }
 if (window.XMLHttpRequest)
 {
  // IE7+, Firefox, Chrome, Opera, Safari 浏览器执行
  xmlhttp=new XMLHttpRequest();
 }
 else
 {
  // IE6, IE5 浏览器执行
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
 xmlhttp.onreadystatechange=function()
 {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {
   document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
  }
 }
 xmlhttp.open("GET","getcd.php?q="+str,true);
 xmlhttp.send();
}
</script>
</head>
<body>

<form>
Select a CD:
<select name="cds" onchange="showCD(this.value)">
<option value="">Select a CD:</option>
<option value="Bob Dylan">Bob Dylan</option>
<option value="Bonnie Tyler">Bonnie Tyler</option>
<option value="Dolly Parton">Dolly Parton</option>
</select>
</form>
<div id="txtHint"><b>CD info will be listed here...</b></div>

</body>
</html>
```

showCD() 函数会执行以下步骤：

- 检查是否有 CD 被选择
- 创建 XMLHttpRequest 对象
- 创建在服务器响应就绪时执行的函数
- 向服务器上的文件发送请求
- 请注意添加到 URL 末端的参数（q）（包含下拉列表的内容）

PHP 文件

上面这段通过 JavaScript 调用的服务器页面是名为 "getcd.php" 的 PHP 文件。

PHP 脚本加载 XML 文档，"cd_catalog.xml"，运行针对 XML 文件的查询，并以 HTML 返回结果：

```
<?php
$q=$_GET["q"];

$xmlDoc = new DOMDocument();
$xmlDoc->load("cd_catalog.xml");

$x=$xmlDoc->getElementsByTagName('ARTIST');

for ($i=0; $i<=$x->length-1; $i++)
{
 // 处理元素节点
 if ($x->item($i)->nodeType==1)
 {
  if ($x->item($i)->childNodes->item(0)->nodeValue == $q)
  {
   $y=($x->item($i)->parentNode);
  }
 }
}

$cd=($y->childNodes);

for ($i=0;$i<$cd->length;$i++)
{
 // 处理元素节点
 if ($cd->item($i)->nodeType==1)
 {
  echo("<b>" . $cd->item($i)->nodeName . ":</b> ");
  echo($cd->item($i)->childNodes->item(0)->nodeValue);
  echo("<br>");
 }
}
?>
```

当 CD 查询从 JavaScript 发送到 PHP 页面时，将发生：

1. PHP 创建 XML DOM 对象
2. 查找所有 <artist> 元素中与 JavaScript 所传数据相匹配的名字
3. 输出 album 的信息，并发送回 "txtHint" 占位符

PHP 实例 - AJAX 实时搜索

AJAX 可为用户提供更友好、交互性更强的搜索体验。

AJAX Live Search

在下面的实例中，我们将演示一个实时的搜索，在您键入数据的同时即可得到搜索结果。

实时的搜索与传统的搜索相比，具有很多优势：

- 当键入数据时，就会显示出匹配的结果
- 当继续键入数据时，对结果进行过滤
- 如果结果太少，删除字符就可以获得更宽的范围

在下面的文本框中输入 "HTML"，搜索包含 HTML 的页面：

上面实例中的结果在一个 XML 文件（links.xml）中进行查找。为了让这个例子小而简单，我们只提供 6 个结果。

实例解释 - HTML 页面

当用户在上面的输入框中键入字符时，会执行 "showResult()" 函数。该函数由 "onkeyup" 事件触发：

```
<html>
<head>
<script>
function showResult(str)
{
 if (str.length==0)
 {
  document.getElementById("livesearch").innerHTML="";
  document.getElementById("livesearch").style.border="0px";
  return;
 }
 if (window.XMLHttpRequest)
 {// IE7+, Firefox, Chrome, Opera, Safari 浏览器执行
  xmlhttp=new XMLHttpRequest();
 }
 else
 {// IE6, IE5 浏览器执行
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
 xmlhttp.onreadystatechange=function()
 {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {
   document.getElementById("livesearch").innerHTML=xmlhttp.responseText;
   document.getElementById("livesearch").style.border="1px solid #A5ACB2";
  }
 }
 xmlhttp.open("GET","livesearch.php?q="+str,true);
 xmlhttp.send();
}
</script>
</head>
<body>

<form>
<input type="text" size="30" onkeyup="showResult(this.value)">
<div id="livesearch"></div>
</form>

</body>
</html>
```

源代码解释：

如果输入框是空的（str.length==0），该函数会清空 livesearch 占位符的内容，并退出该函数。

如果输入框不是空的，那么 showResult() 会执行以下步骤：

- 创建 XMLHttpRequest 对象
- 创建在服务器响应就绪时执行的函数
- 向服务器上的文件发送请求
- 请注意添加到 URL 末端的参数（q）（包含输入框的内容）

PHP 文件

上面这段通过 JavaScript 调用的服务器页面是名为 "livesearch.php" 的 PHP 文件。

"livesearch.php" 中的源代码会搜索 XML 文件中匹配搜索字符串的标题，并返回结果：

```
<?php
$xmlDoc=new DOMDocument();
$xmlDoc->load("links.xml");

$x=$xmlDoc->getElementsByTagName('link');

// 从 URL 中获取参数 q 的值
$q=$_GET["q"];

// 如果 q 参数存在则从 xml 文件中查找数据
if (strlen($q)>0)
{
 $hint="";
 for($i=0; $i<($x->length); $i++)
 {
  $y=$x->item($i)->getElementsByTagName('title');
  $z=$x->item($i)->getElementsByTagName('url');
  if ($y->item(0)->nodeType==1)
  {
   // 找到匹配搜索的链接
   if (stristr($y->item(0)->childNodes->item(0)->nodeValue,$q))
   {
    if ($hint=="")
    {
     $hint="<a href='" .
     $z->item(0)->childNodes->item(0)->nodeValue .
     "' target='_blank'>" .
     $y->item(0)->childNodes->item(0)->nodeValue . "</a>";
    }
    else
    {
     $hint=$hint . "<br /><a href='" .
     $z->item(0)->childNodes->item(0)->nodeValue .
     "' target='_blank'>" .
     $y->item(0)->childNodes->item(0)->nodeValue . "</a>";
    }
   }
  }
 }
}

// 如果没找到则返回 "no suggestion"
if ($hint=="")
{
 $response="no suggestion";
}
else
{
 $response=$hint;
}

// 输出结果
echo $response;
?>
```

如果 JavaScript 发送了任何文本（即 strlen($q) > 0），则会发生：

- 加载 XML 文件到新的 XML DOM 对象
- 遍历所有的 <title> 元素，以便找到匹配 JavaScript 所传文本
- 在 "$response" 变量中设置正确的 URL 和标题。如果找到多于一个匹配，所有的匹配都会添加到变量。
- 如果没有找到匹配，则把 $response 变量设置为 "no suggestion"。

PHP å®ä¾ - AJAX RSS éè¯»å¨

RSS éè¯»å¨ç¨äºéè¯» RSS Feedã

AJAX RSS éè¯»å¨

å¨ä¸é¢çå®ä¾ä¸ï¼æä»¬å°æ¼ç¤ºä¸ä¸ª RSS éè¯»å¨ï¼éè¿å®ï¼æ¥èª RSS çåå®¹å¨ç½é¡µä¸è¿è¡å·æ°çæåµä¸è¢«è½½å¥ï¼

éæ©ä¸ä¸ª RSS-feed:

RSS-feed æ°æ®åè¡¨...

å®ä¾è§£é - HTML é¡µé¢

å½ç¨æ·å¨ä¸é¢çä¸æåè¡¨ä¸éæ©æä¸ª RSS-feed æ¶ï¼ä¼æ§è¡åä¸º "showRSS()" çå½æ°ãè¯¥å½æ°ç± "onchange" äºä»¶è§¦åï¼

```
<html>
<head>
<meta charset="utf-8">
<title>èé¸æç¨(runoob.com)</title>
<script>
function showRSS(str)
{
 if (str.length==0)
 {
  document.getElementById("rssOutput").innerHTML="";
  return;
  }
 if (window.XMLHttpRequest)
 {
  // IE7+, Firefox, Chrome, Opera, Safari æµè§å¨æ§è¡ä»£ç
  xmlhttp=new XMLHttpRequest();
 }
 else
 {
  // IE6, IE5 æµè§å¨æ§è¡ä»£ç
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
 xmlhttp.onreadystatechange=function()
 {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {
   document.getElementById("rssOutput").innerHTML=xmlhttp.responseText;
  }
 }
 xmlhttp.open("GET","getrss.php?q="+str,true);
 xmlhttp.send();
}
</script>
</head>
<body>

<form>
<select onchange="showRSS(this.value)">
<option value="">éæ©ä¸ä¸ª RSS-feed:</option>
<option value="rss">è¯»å RSS æ°æ®</option>
</select>
</form>
<br>
<div id="rssOutput">RSS-feed æ°æ®åè¡¨...</div>
</body>
</html>
```

showRSS() å½æ°ä¼æ§è¡ä»¥ä¸æ¥éª¤ï¼

- æ£æ¥æ¯å¦æ RSS-feed è¢«éæ©
- åå»º XMLHttpRequest å¯¹è±¡
- åå»ºå¨æå¡å¨ååºå°±ç»ªæ¶æ§è¡çå½æ°
- åæå¡å¨ä¸çæä»¶åéè¯·æ±
- è¯·æ³¨ææ·»å å° URL æ«ç«¯çåæ°ï¼qï¼ï¼åå«ä¸æåè¡¨çåå®¹ï¼

PHP æä»¶

æä»¶ rss_demo.xmlã

ä¸é¢è¿æ®µéè¿ JavaScript è°ç¨çæå¡å¨é¡µé¢æ¯åä¸º "getrss.php" ç PHP æä»¶ï¼

```
<?php
// rss æä»¶
$xml="rss_demo.xml";

$xmlDoc = new DOMDocument();
$xmlDoc->load($xml);

// ä» "<channel>" ä¸è¯»åå ç´ 
$channel=$xmlDoc->getElementsByTagName('channel')->item(0);
$channel_title = $channel->getElementsByTagName('title')
->item(0)->childNodes->item(0)->nodeValue;
$channel_link = $channel->getElementsByTagName('link')
->item(0)->childNodes->item(0)->nodeValue;
```

```
$channel_desc = $channel->getElementsByTagName('description')
->item(0)->childNodes->item(0)->nodeValue;

// è¾å º "<channel>" ä¸çå ç´
echo("<p><a href='" . $channel_link
  . "'>" . $channel_title . "</a>");
echo("<br>");
echo($channel_desc . "</p>");

// è¾å º "<item>" ä¸çå ç´
$x=$xmlDoc->getElementsByTagName('item');
for ($i=0; $i<=1; $i++) {
 $item_title=$x->item($i)->getElementsByTagName('title')
 ->item(0)->childNodes->item(0)->nodeValue;
 $item_link=$x->item($i)->getElementsByTagName('link')
 ->item(0)->childNodes->item(0)->nodeValue;
 $item_desc=$x->item($i)->getElementsByTagName('description')
 ->item(0)->childNodes->item(0)->nodeValue;
 echo ("<p><a href='" . $item_link
 . "'>" . $item_title . "</a>");
 echo ("<br>");
 echo ($item_desc . "</p>");
}
?>
```

å½ RSS feed çè¯·æ±ä» JavaScript åéå° PHP æä»¶æ¶ï¼å°åçï¼

- æ£æ¥åªä¸ª RSS feed è¢«éä¸
- åå»ºä¸ä¸ªæ°ç XML DOM å¯¹è±¡
- å¨ xml åéä¸å è½½ RSS ææ¡£
- ä» channel åç´ ä¸æåå¹¶è¾åºæ°æ®
- ä» item åç´ ä¸æåå¹¶è¾åºæ°æ®

PHP å®ä¾ - AJAX æç¥¨

AJAX æç¥¨

å¨ä¸é¢çå®ä¾ä¸ï¼æä»¬å°æ¼ç¤ºä¸ä¸ªæç¥¨ç¨åºï¼éè¿å®ï¼æç¥¨ç»æå¨ä¸è¿è¡é¡µé¢å·æ°çæåµä¸è¢«æ¾ç¤ºã

ä½ åæ¬¢ PHP å AJAX å?

æ¯: ○
å¦: ○

å®ä¾è§£é - HTML é¡µé¢

å½ç¨æ·å¨ä¸é¢éæ©ä¸ä¸ªéé¡¹æ¶ï¼ä¼æ§è¡åä¸º "getVote()" çå½æ°ãè¯¥å½æ°ç± "onclick" äºä»¶è§¦åã

poll.html æä»¶ä»£ç å¦ä¸ï¼

```
<html>
<head>
<meta charset="utf-8">
<title>èé¸æ (runoob.com)</title>
<script>
function getVote(int) {
  if (window.XMLHttpRequest) {
    // IE7+, Firefox, Chrome, Opera, Safari æ§è¡ä»£ç 
    xmlhttp=new XMLHttpRequest();
  } else {
    // IE6, IE5 æ§è¡ä»£ç 
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
  xmlhttp.onreadystatechange=function() {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
      document.getElementById("poll").innerHTML=xmlhttp.responseText;
    }
  }
  xmlhttp.open("GET","poll_vote.php?vote="+int,true);
  xmlhttp.send();
}
</script>
</head>
<body>

<div id="poll">
<h3>ä½ åæ¬¢ PHP å AJAX å?</h3>
<form>
æ¯:
<input type="radio" name="vote" value="0" onclick="getVote(this.value)">
<br>å¦:
<input type="radio" name="vote" value="1" onclick="getVote(this.value)">
</form>
</div>

</body>
</html>
```

getVote() å½æ°åäºä¸é¢çæ¥éª¤ï¼

- åå»º XMLHttpRequest å¯¹è±¡
- åå»ºå¨æå¡å¨ååºå°±ç»ªæ¶æ§è¡çå½æ°
- åæå¡å¨ä¸çæä»¶åéè¯·æ±
- è¯·æ³¨æå¨ URL æ«å°¾æ·»å çåæ°ï¼qï¼ï¼åå«ä¸ææ¡çåå®¹ï¼

PHP æä»¶

ä¸é¢éè¿ JavaScript ä»£ç è°ç¨çæå¡å¨é¡µé¢æ¯åä¸º "**poll_vote.php**" ç PHP æä»¶ï¼

```
<?php
$vote = htmlspecialchars($_REQUEST['vote']);

// è·åæä»¶ä¸åå¨çæ°æ®
$filename = "poll_result.txt";
$content = file($filename);

// å°æ°æ®åå²å°æ°ç»ä¸
$array = explode("||", $content[0]);
$yes = $array[0];
$no = $array[1];

if ($vote == 0)
{
  $yes = $yes + 1;
}

if ($vote == 1)
{
  $no = $no + 1;
}

// æå¥æç¥¨æ°æ®
$insertvote = $yes."||".$no;
$fp = fopen($filename,"w");
fputs($fp,$insertvote);
fclose($fp);
?>

<h2>ç»æ:</h2>
<table>
  <tr>
  <td>æ¯:</td>
  <td>
  <span style="display: inline-block; background-color:green;
      width:<?php echo(100*round($yes/($no+$yes),2)); ?>px;
      height:20px;" ></span>
  <?php echo(100*round($yes/($no+$yes),2)); ?>%
  </td>
  </tr>
  <tr>
  <td>å¦:</td>
  <td>
  <span style="display: inline-block; background-color:red;
      width:<?php echo(100*round($no/($no+$yes),2)); ?>px;
      height:20px;"></span>
  <?php echo(100*round($no/($no+$yes),2)); ?>%
  </td>
  </tr>
</table>
```

å½æéçå¼ä» JavaScript åéå° PHP æä»¶æ¶ï¼åçï¼

1. è·å "poll_result.txt" æä»¶çåå®¹
2. æä»¶åå®¹æ¾å¥åéï¼å¹¶ç»è¢«éæ©çåéç´¯å 1
3. æç»æåå¥ "poll_result.txt" æä»¶
4. è¾åºå¾å½¢åçæç¥¨ç»æ

ææ¬æä»¶

ææ¬æä»¶ï¼**poll_result.txt**ï¼ä¸åå¨æç¥¨ç¨åºçæ°æ®ã

å®åå¨çæ°æ®å¦ä¸æç¤ºï¼

3||4

ç¬¬ä¸ä¸ªæ°åè¡¨ç¤º "Yes" çæç¥¨æ°ï¼ç¬¬äºä¸ªæ°åè¡¨ç¤º "No" çæç¥¨æ°ã

æ³¨éï¼è®°å¾åªåè®¸æ¨ç Web æå¡å¨æ¥ç¼è¾è¯¥ææ¬æä»¶ãä¸è¦è®©å¶ä»äººè·å¾è®¿é®æï¼é¤äº Web æå¡å¨ (PHP)ã

PHP 5 时区

PHP 支持的时区

下面是 PHP 支持的时区的完整列表，这些对一些 PHP 日期函数很有用。

- 非洲
- 美洲
- 南极洲
- 北冰洋
- 亚洲
- 大西洋
- 大洋洲
- 欧洲
- 印度洋
- 太平洋

非洲

Africa/Abidjan	Africa/Accra	Africa/Addis_Ababa	Africa/Algiers	Africa/Asmara
Africa/Asmera	Africa/Bamako	Africa/Bangui	Africa/Banjul	Africa/Bissau
Africa/Blantyre	Africa/Brazzaville	Africa/Bujumbura	Africa/Cairo	Africa/Casablanca
Africa/Ceuta	Africa/Conakry	Africa/Dakar	Africa/Dar_es_Salaam	Africa/Djibouti
Africa/Douala	Africa/El_Aaiun	Africa/Freetown	Africa/Gaborone	Africa/Harare
Africa/Johannesburg	Africa/Juba	Africa/Kampala	Africa/Khartoum	Africa/Kigali
Africa/Kinshasa	Africa/Lagos	Africa/Libreville	Africa/Lome	Africa/Luanda
Africa/Lubumbashi	Africa/Lusaka	Africa/Malabo	Africa/Maputo	Africa/Maseru
Africa/Mbabane	Africa/Mogadishu	Africa/Monrovia	Africa/Nairobi	Africa/Ndjamena
Africa/Niamey	Africa/Nouakchott	Africa/Ouagadougou	Africa/Porto-Novo	Africa/Sao_Tome
Africa/Timbuktu	Africa/Tripoli	Africa/Tunis	Africa/Windhoek	

美洲

America/Adak	America/Anchorage	America/Anguilla
America/Antigua	America/Araguaina	America/Argentina/Buenos_Aires
America/Argentina/Catamarca	America/Argentina/ComodRivadavia	America/Argentina/Cordoba
America/Argentina/Jujuy	America/Argentina/La_Rioja	America/Argentina/Mendoza
America/Argentina/Rio_Gallegos	America/Argentina/Salta	America/Argentina/San_Juan
America/Argentina/San_Luis	America/Argentina/Tucuman	America/Argentina/Ushuaia
America/Aruba	America/Asuncion	America/Atikokan
America/Atka	America/Bahia	America/Bahia_Banderas
America/Barbados	America/Belem	America/Belize
America/Blanc-Sablon	America/Boa_Vista	America/Bogota
America/Boise	America/Buenos_Aires	America/Cambridge_Bay
America/Campo_Grande	America/Cancun	America/Caracas
America/Catamarca	America/Cayenne	America/Cayman
America/Chicago	America/Chihuahua	America/Coral_Harbour
America/Cordoba	America/Costa_Rica	America/Creston
America/Cuiaba	America/Curacao	America/Danmarkshavn
America/Dawson	America/Dawson_Creek	America/Denver
America/Detroit	America/Dominica	America/Edmonton
America/Eirunepe	America/El_Salvador	America/Ensenada
America/Fort_Wayne	America/Fortaleza	America/Glace_Bay

America/Godthab America/Goose_Bay America/Grand_Turk
America/Grenada America/Guadeloupe America/Guatemala
America/Guayaquil America/Guyana America/Halifax
America/Havana America/Hermosillo America/Indiana/Indianapolis
America/Indiana/Knox America/Indiana/Marengo America/Indiana/Petersburg
America/Indiana/Tell_City America/Indiana/Vevay America/Indiana/Vincennes
America/Indiana/Winamac America/Indianapolis America/Inuvik
America/Iqaluit America/Jamaica America/Jujuy
America/Juneau America/Kentucky/Louisville America/Kentucky/Monticello
America/Knox_IN America/Kralendijk America/La_Paz
America/Lima America/Los_Angeles America/Louisville
America/Lower_Princes America/Maceio America/Managua
America/Manaus America/Marigot America/Martinique
America/Matamoros America/Mazatlan America/Mendoza
America/Menominee America/Merida America/Metlakatla
America/Mexico_City America/Miquelon America/Moncton
America/Monterrey America/Montevideo America/Montreal
America/Montserrat America/Nassau America/New_York
America/Nipigon America/Nome America/Noronha
America/North_Dakota/Beulah America/North_Dakota/Center America/North_Dakota/New_Salem
America/Ojinaga America/Panama America/Pangnirtung
America/Paramaribo America/Phoenix America/Port-au-Prince
America/Port_of_Spain America/Porto_Acre America/Porto_Velho
America/Puerto_Rico America/Rainy_River America/Rankin_Inlet
America/Recife America/Regina America/Resolute
America/Rio_Branco America/Rosario America/Santa_Isabel
America/Santarem America/Santiago America/Santo_Domingo
America/Sao_Paulo America/Scoresbysund America/Shiprock
America/Sitka America/St_Barthelemy America/St_Johns
America/St_Kitts America/St_Lucia America/St_Thomas
America/St_Vincent America/Swift_Current America/Tegucigalpa
America/Thule America/Thunder_Bay America/Tijuana
America/Toronto America/Tortola America/Vancouver
America/Virgin America/Whitehorse America/Winnipeg
America/Yakutat America/Yellowknife

南极洲

Antarctica/Casey Antarctica/Davis Antarctica/DumontDUrville Antarctica/Macquarie Antarctica/Mawson
Antarctica/McMurdo Antarctica/Palmer Antarctica/Rothera Antarctica/South_Pole Antarctica/Syowa
Antarctica/Vostok

北冰洋

Arctic/Longyearbyen

亚洲

Asia/Aden Asia/Almaty Asia/Amman Asia/Anadyr Asia/Aqtau
Asia/Aqtobe Asia/Ashgabat Asia/Ashkhabad Asia/Baghdad Asia/Bahrain
Asia/Baku Asia/Bangkok Asia/Beirut Asia/Bishkek Asia/Brunei
Asia/Calcutta Asia/Choibalsan Asia/Chongqing Asia/Chungking Asia/Colombo
Asia/Dacca Asia/Damascus Asia/Dhaka Asia/Dili Asia/Dubai
Asia/Dushanbe Asia/Gaza Asia/Harbin Asia/Hebron Asia/Ho_Chi_Minh
Asia/Hong_Kong Asia/Hovd Asia/Irkutsk Asia/Istanbul Asia/Jakarta
Asia/Jayapura Asia/Jerusalem Asia/Kabul Asia/Kamchatka Asia/Karachi
Asia/Kashgar Asia/Kathmandu Asia/Katmandu Asia/Khandyga Asia/Kolkata
Asia/Krasnoyarsk Asia/Kuala_Lumpur Asia/Kuching Asia/Kuwait Asia/Macao
Asia/Macau Asia/Magadan Asia/Makassar Asia/Manila Asia/Muscat
Asia/Nicosia Asia/Novokuznetsk Asia/Novosibirsk Asia/Omsk Asia/Oral
Asia/Phnom_Penh Asia/Pontianak Asia/Pyongyang Asia/Qatar Asia/Qyzylorda
Asia/Rangoon Asia/Riyadh Asia/Saigon Asia/Sakhalin Asia/Samarkand
Asia/Seoul Asia/Shanghai Asia/Singapore Asia/Taipei Asia/Tashkent
Asia/Tbilisi Asia/Tehran Asia/Tel_Aviv Asia/Thimbu Asia/Thimphu
Asia/Tokyo Asia/Ujung_Pandang Asia/Ulaanbaatar Asia/Ulan_Bator Asia/Urumqi
Asia/Ust-Nera Asia/Vientiane Asia/Vladivostok Asia/Yakutsk Asia/Yekaterinburg
Asia/Yerevan

大西洋

Atlantic/Azores Atlantic/Bermuda Atlantic/Canary Atlantic/Cape_Verde Atlantic/Faeroe
Atlantic/Faroe Atlantic/Jan_Mayen Atlantic/Madeira Atlantic/Reykjavik Atlantic/South_Georgia
Atlantic/St_Helena Atlantic/Stanley

大洋洲

Australia/ACT Australia/Adelaide Australia/Brisbane Australia/Broken_Hill Australia/Canberra
Australia/Currie Australia/Darwin Australia/Eucla Australia/Hobart Australia/LHI
Australia/Lindeman Australia/Lord_Howe Australia/Melbourne Australia/North Australia/NSW
Australia/Perth Australia/Queensland Australia/South Australia/Sydney Australia/Tasmania
Australia/Victoria Australia/West Australia/Yancowinna

欧洲

Europe/Amsterdam Europe/Andorra Europe/Athens Europe/Belfast Europe/Belgrade
Europe/Berlin Europe/Bratislava Europe/Brussels Europe/Bucharest Europe/Budapest
Europe/Busingen Europe/Chisinau Europe/Copenhagen Europe/Dublin Europe/Gibraltar
Europe/Guernsey Europe/Helsinki Europe/Isle_of_Man Europe/Istanbul Europe/Jersey
Europe/Kaliningrad Europe/Kiev Europe/Lisbon Europe/Ljubljana Europe/London
Europe/Luxembourg Europe/Madrid Europe/Malta Europe/Mariehamn Europe/Minsk
Europe/Monaco Europe/Moscow Europe/Nicosia Europe/Oslo Europe/Paris
Europe/Podgorica Europe/Prague Europe/Riga Europe/Rome Europe/Samara
Europe/San_Marino Europe/Sarajevo Europe/Simferopol Europe/Skopje Europe/Sofia
Europe/Stockholm Europe/Tallinn Europe/Tirane Europe/Tiraspol Europe/Uzhgorod
Europe/Vaduz Europe/Vatican Europe/Vienna Europe/Vilnius Europe/Volgograd
Europe/Warsaw Europe/Zagreb Europe/Zaporozhye Europe/Zurich

印度洋

Indian/Antananarivo Indian/Chagos Indian/Christmas Indian/Cocos Indian/Comoro
Indian/Kerguelen Indian/Mahe Indian/Maldives Indian/Mauritius Indian/Mayotte
Indian/Reunion

太平洋

Pacific/Apia Pacific/Auckland Pacific/Chatham Pacific/Chuuk Pacific/Easter
Pacific/Efate Pacific/Enderbury Pacific/Fakaofo Pacific/Fiji Pacific/Funafuti
Pacific/Galapagos Pacific/Gambier Pacific/Guadalcanal Pacific/Guam Pacific/Honolulu
Pacific/Johnston Pacific/Kiritimati Pacific/Kosrae Pacific/Kwajalein Pacific/Majuro
Pacific/Marquesas Pacific/Midway Pacific/Nauru Pacific/Niue Pacific/Norfolk
Pacific/Noumea Pacific/Pago_Pago Pacific/Palau Pacific/Pitcairn Pacific/Pohnpei
Pacific/Ponape Pacific/Port_Moresby Pacific/Rarotonga Pacific/Saipan Pacific/Samoa
Pacific/Tahiti Pacific/Tarawa Pacific/Tongatapu Pacific/Truk Pacific/Wake
Pacific/Wallis Pacific/Yap

PHP array() 函数

[ ] 定制的 PHP Array 参考手册

实例

创建名为 $cars 的数值数组，赋三个元素给它，并打印包含数组值的文本：

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

运行实例 »

定义和用法

array() 函数用于创建数组。

在 PHP 中，有三种类型的数组：

- **数值数组** - 带有数字 ID 键的数组
- **关联数组** - 带有指定的键的数组，每个键关联一个值
- **多维数组** - 包含一个或多个数组的数组

语法

数值数组的语法：

array(*value1,value2,value3,etc.*);

关联数组的语法：

array(*key=>value,key=>value,key=>value,etc.*);

参数	描述
key	规定键名（数值或字符串）。
value	规定键值。

技术细节

返回值：	返回参数的数组。
PHP 版本：	4+
更新日志：	自 PHP 5.4 起，可以使用短数组语法，用 [] 代替 array()。 例如，用 $cars=["Volvo","BMW"]; 代替 $cars=array("Volvo","BMW");

更多实例

实例 1

创建名为 $age 的关联数组：

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>
```

运行实例 »

实例 2

遍历和打印数值数组的值：

```
<?php
$cars=array("Volvo","BMW","Toyota");
$arrlength=count($cars);

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

运行实例 »

实例 3

遍历和打印关联数组的值：

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

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

运行实例 »

实例 4

创建多维数组：

```
<?php
// 一个二维数组
$cars=array
(
array("Volvo",100,96),
array("BMW",60,59),
array("Toyota",110,100)
);
?>
```

运行实例 »

完整的 PHP Array 参考手册

PHP array_change_key_case() å½æ°

å®æ´ç PHP Array åèæå

å®ä¾

å°æ°ç»çææçé®è½¬æ¢ä¸ºå¤§ååæ¯ï¼

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
print_r(array_change_key_case($age,CASE_UPPER));
?>
```

è¿è¡å®ä¾ Â»

å®ä¹åç¨æ³

array_change_key_case() å½æ°å°æ°ç»çææçé®é½è½¬æ¢ä¸ºå¤§ååæ¯æå°ååæ¯ã

è¯æ³

array_change_key_case(*array*,*case*);

åæ°	æè¿°
array	å¿éãè§å®è¦ä½¿ç¨çæ°ç»ã
case	å¯éãå¯è½çå¼ï¼ • CASE_LOWER - é»è®¤å¼ãå°æ°ç»çé®è½¬æ¢ä¸ºå°ååæ¯ã • CASE_UPPER - å°æ°ç»çé®è½¬æ¢ä¸ºå¤§ååæ¯ã

ææ¯ç»è

è¿åå¼ï¼	è¿åå¶é®ä¸ºå¤§ååæ¯çæ°ç»ï¼æè¿åå¶é®ä¸ºå°ååæ¯çæ°ç»ï¼æèå¦æ *array* ä¸æ¯ä¸ä¸ªæ°ç»åè¿å FALSEã
PHP çæ¬ï¼	4.2+

æ´å¤å®ä¾

å®ä¾ 1

å°æ°ç»çææçé®è½¬æ¢ä¸ºå°ååæ¯ï¼

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
print_r(array_change_key_case($age,CASE_LOWER));
?>
```

è¿è¡å®ä¾ Â»

å®ä¾ 2

å¦æè¿è¡å® array_change_key_case() ä¹åæä¸¤ä¸ªæèæ´å¤ä¸ªçé®ç¸åï¼æ¯å¦ "b" å "B"ï¼ï¼åæåçå¼å°è¦çå¶ä»çå¼ï¼

```
<?php
$pets=array("a"=>"Cat","B"=>"Dog","c"=>"Horse","b"=>"Bird");
print_r(array_change_key_case($pets,CASE_UPPER));
?>
```

è¿è¡å®ä¾ Â»

å®æ´ç PHP Array åèæå

PHP array_chunk() 函数

完整的 PHP Array 参考手册

实例

把数组分割为带有两个元素的数组块：

```
<?php
$cars=array("Volvo","BMW","Toyota","Honda","Mercedes","Opel");
print_r(array_chunk($cars,2));
?>
```

运行实例 »

定义和用法

array_chunk() 函数把一个数组分割为新的数组块。

语法

array_chunk(*array*,*size*,*preserve_keys*);

参数	描述
array	必需。规定要使用的数组。
size	必需。一个整数，规定每个新数组块包含多少个元素。
preserve_key	可选。可能的值： • true - 保留原始数组中的键名。 • false - 默认。每个新数组块使用从零开始的索引。

技术细节

返回值：	返回一个多维的数值数组，从 0 开始，每个维度都包含 *size* 元素。
PHP 版本：	4.2+

更多实例

实例 1

把数组分割为带有两个元素的数组块，并保留原始数组中的键名：

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43","Harry"=>"50");
print_r(array_chunk($age,2,true));
?>
```

运行实例 »

完整的 PHP Array 参考手册

PHP array_combine() å½æ°

å®æ´ç PHP Array åèæå

å®ä¾

éè¿åå¹¶ä¸¤ä¸ªæ°ç»æ¥åå»ºä¸ä¸ªæ°æ°ç»ï¼å¶ä¸çä¸ä¸ªæ°ç»åç´ ä¸ºé®åï¼å¦ä¸ä¸ªæ°ç»åç´ ä¸ºé®å¼ï¼

```
<?php
$fname=array("Peter","Ben","Joe");
$age=array("35","37","43");

$c=array_combine($fname,$age);
print_r($c);
```

?>

è¿è¡å®ä¾ Â»

å®ä¹åç¨æ³

array_combine() å½æ°éè¿åå¹¶ä¸¤ä¸ªæ°ç»æ¥åå»ºä¸ä¸ªæ°æ°ç»ï¼å¶ä¸çä¸ä¸ªæ°ç»åç´ ä¸ºé®åï¼å¦ä¸ä¸ªæ°ç»çåç´ ä¸ºé®å¼ã

æ³¨éï¼é®åæ°ç»åé®å¼æ°ç»çåç´ ä¸ªæ°å¿é¡»ç¸åï¼

è¯æ³

array_combine(*keys*,*values*);

åæ°	æè¿°
keys	å¿éãè§å®æ°ç»çé®åã
values	å¿éãè§å®æ°ç»çé®å¼ã

ææ¯ç»è

è¿åå¼ï¼	è¿ååå¹¶åçæ°ç»ãå¦æä¸¤ä¸ªæ°ç»çåç´ ä¸ªæ°ä¸ç¸åï¼åè¿å FALSEã
PHP çæ¬ï¼	5+
æ´æ°æ¥å¿ï¼	å¨ PHP 5.4 çæ¬ä¹åï¼å¦ææ°ç»ä¸ºç©ºï¼åä¼æ¥ E_WARNING çº§å«éè¯¯å¹¶è¿å FALSEã

å®æ´ç PHP Array åèæå

PHP array_count_values() 函数

完整的 PHP Array 参考手册

实例

统计数组中所有值出现的次数：

```
<?php
$a=array("A","Cat","Dog","A","Dog");
print_r(array_count_values($a));
?>
```

运行实例 »

定义和用法

array_count_values() 函数用于统计数组中所有值出现的次数。

语法

array_count_values(*array*)

参数	描述
array	必需。规定需要统计数组中所有值出现次数的数组。

技术细节

返回值：	返回一个关联数组，其元素的键名是原数组的值，键值是该值在原数组中出现的次数。
PHP 版本：	4+

完整的 PHP Array 参考手册

PHP array_diff() å½æ°

å®æ´ç PHP Array åèæå

å®ä¾

æ¯è¾ä¸¤ä¸ªæ°ç»**çé®å¼**ï¼å¹¶è¿åå·®éï¼

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"green","g"=>"blue");

$result=array_diff($a1,$a2);
print_r($result);
?>
```

è¿è¡å®ä¾ Â»

å®ä¹åç¨æ³

array_diff() å½æ°ç¨äºæ¯è¾ä¸¤ä¸ªï¼æèæ´å¤ä¸ªï¼æ°ç»**çé®å¼**ï¼å¹¶è¿åå·®éã

è¯¥å½æ°æ¯è¾ä¸¤ä¸ªï¼æèæ´å¤ä¸ªï¼æ°ç»çé®å¼ï¼å¹¶è¿åä¸ä¸ªå·®éæ°ç»ï¼è¯¥æ°ç»åæ¬äºææå¨è¢«æ¯è¾çæ°ç»ï¼*array1*ï¼ä¸ï¼ä½æ¯ä¸å¨ä»»ä½å¶ä»åæ°æ°ç»ï¼*array2* æ *array3* ççï¼ä¸çé®å¼ã

è¯æ³

array_diff(*array1*,*array2*,*array3*...);

åæ°	æè¿°
array1	å¿éãä¸å¶ä»æ°ç»è¿è¡æ¯è¾çç¬¬ä¸ä¸ªæ°ç»ã
array2	å¿éãä¸ç¬¬ä¸ä¸ªæ°ç»è¿è¡æ¯è¾çæ°ç»ã
array3,...	å¯éãä¸ç¬¬ä¸ä¸ªæ°ç»è¿è¡æ¯è¾çå¶ä»æ°ç»ã

ææ¯ç»è

è¿åå¼ï¼	è¿åä¸ä¸ªå·®éæ°ç»ï¼è¯¥æ°ç»åæ¬äºææå¨è¢«æ¯è¾çæ°ç»ï¼*array1*ï¼ä¸ï¼ä½æ¯ä¸å¨ä»»ä½å¶ä»åæ°æ°ç»ï¼*array2* æ *array3* ççï¼ä¸çé®å¼ã
PHP çæ¬ï¼	4.0.1+

æ´å¤å®ä¾

å®ä¾ 1

æ¯è¾ä¸ä¸ªæ°ç»**çé®å¼**ï¼å¹¶è¿åå·®éï¼

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"black","g"=>"purple");
$a3=array("a"=>"red","b"=>"black","h"=>"yellow");

$result=array_diff($a1,$a2,$a3);
print_r($result);
?>
```

è¿è¡å®ä¾ Â»

å®æ´ç PHP Array åèæå

PHP array_diff_assoc() 函数

完整的 PHP Array 参考手册

实例

比较两个数组的**键名和键值**，并返回差集：

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("a"=>"red","b"=>"green","c"=>"blue");

$result=array_diff_assoc($a1,$a2);
print_r($result);
?>
```

运行实例 »

定义和用法

array_diff_assoc() 函数用于比较两个（或更多个）数组的**键名和键值** ，并返回差集。

该函数比较两个（或更多个）数组的键名和键值，并返回一个差集数组，该数组包括了所有在被比较的数组（*array1*）中，但是不在任何其他参数数组（*array2* 或 *array3* 等等）中的键名和键值。

语法

array_diff_assoc(*array1*,*array2*,*array3*...);

参数	描述
array1	必需。与其他数组进行比较的第一个数组。
array2	必需。与第一个数组进行比较的数组。
array3,...	可选。与第一个数组进行比较的其他数组。

技术细节

返回值：	返回一个差集数组，该数组包括了所有在被比较的数组（*array1*）中，但是不在任何其他参数数组（*array2* 或 *array3* 等等）中的键名和键值。
PHP 版本：	4.3+

更多实例

实例 1

比较两个数组的**键名和键值**，并返回差集：

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"green","g"=>"blue");

$result=array_diff_assoc($a1,$a2);
print_r($result);
?>
```

运行实例 »

实例 2

比较三个数组的键名和键值，并返回差集：

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("a"=>"red","f"=>"green","g"=>"blue");
$a3=array("h"=>"red","b"=>"green","g"=>"blue");

$result=array_diff_assoc($a1,$a2,$a3);
print_r($result);
?>
```

运行实例 »

完整的 PHP Array 参考手册

PHP array_diff_key() å½æ°

å®æ´ç PHP Array åèæå

å®ä¾

æ¯è¾ä¸¤ä¸ªæ°ç»çé®åï¼å¹¶è¿åå·®éï¼

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue");
$a2=array("a"=>"red","c"=>"blue","d"=>"pink");

$result=array_diff_key($a1,$a2);
print_r($result);
?>
```

è¿è¡å®ä¾ Â»

å®ä¹åç¨æ³

array_diff_key() å½æ°ç¨äºæ¯è¾ä¸¤ä¸ªï¼æèæ´å¤ä¸ªï¼æ°ç»çé® ï¼å¹¶è¿åå·®éã

è¯¥å½æ°æ¯è¾ä¸¤ä¸ªï¼æèæ´å¤ä¸ªï¼æ°ç»çé®åï¼å¹¶è¿åä¸ä¸ªå·®éæ°ç»ï¼è¯¥æ°ç»åæ¬äºææå¨è¢«æ¯è¾çæ°ç»ï¼*array1*ï¼ä¸ï¼ä½æ¯ä¸å¨ä»»ä½å¶ä»åæ°æ°ç»ï¼*array2* æ *array3* ççï¼ä¸çé®åã

è¯æ³

array_diff_key(*array1,array2,array3...*);

åæ°	æè¿°
array1	å¿éãä¸å¶ä»æ°ç»è¿è¡æ¯è¾çç¬¬ä¸ä¸ªæ°ç»ã
array2	å¿éãç¬¬ä¸ä¸ªæ°ç»è¿è¡æ¯è¾çæ°ç»ã
array3,...	å¯éãç¬¬ä¸ä¸ªæ°ç»è¿è¡æ¯è¾çå¶ä»æ°ç»ã

ææ¯ç»è

è¿åå¼ï¼	è¿åä¸ä¸ªå·®éæ°ç»ï¼è¯¥æ°ç»åæ¬äºææå¨è¢«æ¯è¾çæ°ç»ï¼*array1*ï¼ä¸ï¼ä½æ¯ä¸å¨ä»»ä½å¶ä»åæ°æ°ç»ï¼*array2* æ *array3* ççï¼ä¸çé®åã
PHP çæ¬ï¼	5.1+

æ´å¤å®ä¾

å®ä¾ 1

æ¯è¾ä¸¤ä¸ªæ°å¼æ°ç»çé®åï¼å¹¶è¿åå·®éï¼

```
<?php
$a1=array("red","green","blue","yellow");
$a2=array("red","green","blue");

$result=array_diff_key($a1,$a2);
print_r($result);
?>
```

è¿è¡å®ä¾ Â»

å®ä¾ 2

æ¯è¾ä¸ä¸ªæ°ç»çé®åï¼å¹¶è¿åå·®éï¼

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue");
$a2=array("c"=>"yellow","d"=>"black","e"=>"brown");
$a3=array("f"=>"green","c"=>"purple","g"=>"red");

$result=array_intersect_key($a1,$a2,$a3);
print_r($result);
?>
```

è¿è¡å®ä¾ Â»

å®æ´ç PHP Array åèæå