**第5章 函数编写**

**1. 代码重用的好处：成本节约、可靠性、一致性**

**2. 使用require()和include()函数, 类似于c语言中的 #include .**

**A. 文件扩展名和require()函数**

**reusable.php代码如下：**

**———————————————————————————————**

**<?php**

**echo "Here is a very simple PHP statement.<br/>";**

**?>**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**main.php代码如下：**

**———————————————————————————————**

**<?php**

**echo "This is the main file.<br/>";**

**require('reuseable.php');**

**echo 'The script will end now.<br/>";**

**?>**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**require('reuseable.php');运行时，被请求的文件内容代替，然后再执行脚本。**

**可以使用任何扩展名来命名包含文件，通常，命名为 .inc(代表include) .php**

**并且，将被包含文件保存在文档树之外。**

**B. 使用require()制作web站点的模板**



**home.html——TLA咨询公司主页的HTML脚本如下：**

**————————————————————————————**

<html>

<head>

<title>TLA Consulting Pty Ltd</title>

<style type="text/css">

h1 {color:white; font-size:24pt; text-align:center;

font-family:arial,sans-serif}

.menu {color:white; font-size:12pt; text-align:center;

font-family:arial,sans-serif; font-weight:bold}

td {background:black}

p {color:black; font-size:12pt; text-align:justify;

font-family:arial,sans-serif}

p.foot {color:white; font-size:9pt; text-align:center;

font-family:arial,sans-serif; font-weight:bold}

a:link,a:visited,a:active {color:white}

</style>

</head>

<body>

<!-- page header -->

<table width="100%" cellpadding="12" cellspacing="0" border="0">

<tr bgcolor="black">

<td align="left"><img src="logo.gif" alt="TLA logo" height="70" width="70" /></td>

<td>

<h1>TLA Consulting</h1>

</td>

<td align="right"><img src="logo.gif" alt="TLA logo" height="70" width="70" /></td>

</tr>

</table>

<!-- menu -->

<table width="100%" bgcolor="white" cellpadding="4" cellspacing="4">

<tr >

<td width="25%">

<img src="s-logo.gif" alt="" height="20" width="20" />

<span class="menu">Home</span></td>

<td width="25%">

<img src="s-logo.gif" alt="" height="20" width="20" />

<span class="menu">Contact</span></td>

<td width="25%">

<img src="s-logo.gif" alt="" height="20" width="20" />

<span class="menu">Services</span></td>

<td width="25%">

<img src="s-logo.gif" alt="" height="20" width="20" />

<span class="menu">Site Map</span></td>

</tr>

</table>

<!-- page content -->

<p>Welcome to the home of TLA Consulting.

Please take some time to get to know us.</p>

<p>We specialize in serving your business needs

and hope to hear from you soon.</p>

**<!-- page footer -->**

**<table width="100%" bgcolor="black" cellpadding="12" border="0">**

**<tr>**

**<td>**

**<p class="foot">&copy; TLA Consulting Pty Ltd.</p>**

**<p class="foot">Please see our**

**<a href="legal.php">legal information page</a></p>**

**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

**—————————————————————————————**

**通常，可以将这个文件分割，然后分别命名为 header.php home.php footer.php**

**home.php 可以代替home.html，它包含页面内容和两个require()语句，代码如下：**

**—————————————————————————————**

**<?php**

**require('header.php');**

**?>**

**<!-- page content -->**

**<p>Welcome to the home of TLA Consulting.**

**Please take some time to get to know us.</p>**

**<p>We specialize in serving your business needs**

**and hope to hear from you soon.</p>**

**<?php**

**require('footer.php');**

**?>**

**——————————————————————————————**

**安全起见，通常，在web文档树之外，建一个可以被访问的目录，将被包含的文件保存为 .inc 。**

**因为，如果扩展名是 .php,而又是不完整的脚本，一旦读入会出错，如果是其他扩展名，则有可能被别人读取源码。**

**C. 使用auto\_prepend\_file 和auto\_append\_file**

**修改php.ini中的两个选项，自动载入页眉和页脚。相当于用include()语句包含的文件一样的效果。**

**3. 在PHP中使用函数**

**A. 调用函数**

**function\_name(); 如 phpinfo();**

**function\_name(parameter); function\_name(2); function\_name(7.99);**

**function\_name($variable); 参数可以是任何数据类型，但是特定的函数通常会要求特定的数据类型。**

**查看fopen()函数原型, 理解含义。调用代码如下：**

**$name = 'myfile.txt';**

**$openmode = 'r';**

**$fp = fopen($name,$openmode);**

**B. 调用未定义的函数**

**如报告错误信息，需要检查：函数名称是否正确、这个函数是否存在于所用的PHP版本中。**

**C. 理解字母大小写和函数名称**

**函数名称不区分大小写。变量名区分。**

**4. 理解为啥要定义自己的函数**

**5. 了解基本的函数结构**

**<?php**

**function my\_function(){**

**echo 'My function was called';**

**}**

**?>**

**函数声明在脚本中，将只能在这个脚本中使用。通常，把用到的函数包含在一个文件中，然后在所有脚本中调用 require()语句，将其包含进来。**

**上例也可以改写为：**

**<?php**

**function my\_function(){**

**?>**

**My function was called**

**<?php**

**}**

**?>**

**函数命名：page\_header() pageheader()都是不错的名称。**

**不能与已有的函数同名**

**名称只能是字母、数字、下划线，数字不开头。**

**PHP不支持函数重载。**

**5. 使用参数**

**参数是一维数组，函数功能是将传入的一维数组以表格的形式显示出来**

**function create\_table($data){**

**echo '<table border=\"1\">';**

**reset($data);**

**$value = current($data);**

**while($value) {**

**echo "<tr><td>" . $value . "</td></tr>\n";**

**$value = next($data);**

**}**

**echo "</table>";**

**}**

**调用：**

**$my\_array = array( 'Line one','Line two','Line three');**

**create\_table($my\_array);**



**改进create\_table(）函数，允许调用时可选地设置表格的边界、宽度、单元大小、空白填充：**

function create\_table2($data, $border=1,$cellpadding=4,$cellspacing=4){

echo '<table border="'.$border. '" cellpadding="'.$cellpadding. '" cellspacing="' . $cellspacing . '">';

reset($data);

$value = current($data);

while($value) {

echo "<tr><td>" . $value . "</td></tr>\n";

$value = next($data);

}

echo "</table>";

}

**调用：**

**$my\_array = array( 'Line one','Line two','Line three');**

**create\_table2($my\_array,3,8,8);**



**注意：不能漏掉一个可选参数而给出参数列表的后一个参数。**

**create\_table2($my\_array,3,8); 是合法的。**

**通过3个帮助器函数，可以确定已经传递了多少个参数，以及这些参数的值。**

function var\_args( )

{

echo "Number of parameters:";

echo func\_num\_args();

echo '<br/>';

$args = func\_get\_args();

foreach ($args as $arg) {

echo $arg . '<br/>';

}

}

**7. 理解作用域**

**PHP的规则：**

1）在函数内部声明的变量，是局部变量，作用域从声明他们的那条语句开始，到函数末尾。

2）在函数外部声明的变量，是全局变量，作用域从声明他们的那条语句开始，到文件末尾。

3）特殊的超级全局变量，在函数内部和外部都是可见的。

使用require() include()不影响作用域。

4）global关键字，可以用来手动指定一个在函数中定义或使用的变量具有全局作用域。

5）用unset($variable\_name)可以手动删除变量。如果变量被删除，它就不在参数所指定的作用域中了。

**function fn(){**

**$var = 'contents';**

**}**

**fn();**

**echo $var;**



**注意：在函数的内部和外部，重复声明一个变量，是合法的，但是不可取。**

**<?php**

**function fn(){**

**$var = "contents 2";**

**echo 'inside the function , $var = '. $var . '<br/>';**

**}**

**$var = "contents 1";**

**fn();**

**echo 'outside the function , $var = '. $var . '<br/>';**

**?>**



**如果希望在函数内部创建的变量具有全局作用域，可以用global关键字：**

**function fn(){**

**global $var;**

**$var = "contents";**

**echo 'inside the function , $var = '. $var . '<br/>';**

**}**

**fn();**

**echo 'outside the function , $var = '. $var . '<br/>';**

**?>**



**如果没有global $var;这一行声明，outside 报错。**

**8. 参数的引用传递和值传递**

**function increment($value,$amount =1){**

**$value = $value + $amount;**

**}**

**$value =10;**

**increment($value);**

**echo $value;**



**function increment(&$value,$amount =1){**

**$value = $value + $amount;**

**}**

**$value =10;**

**echo $value . '<br/>';**

**increment($value);**

**echo $value;**



**9. 使用return关键字**

**function test\_return(){**

**echo 'This statement will be executed';**

**return;**

**echo 'This statement will never be executed';**

**}**

**function larger($x,$y){**

**if(!isset($x) || (!isset($y))){**

**echo 'This function requires two numbers.' ;**

**return;**

**}**

**if($x>$y) {**

**echo $x . '<br/>';**

**}else{**

**echo $y . '<br/>';**

**}**

**}**

**$a = 1;**

**$b = 2.5;**

**$c =1.9;**

**larger($a,$b);**

**larger($c,$a);**

**larger($d,$a);**







**@ larger($d,$a);**



**从函数返回一个值：**

**function larger($x,$y){**

**if(!isset($x) || (!isset($y))){**

**return false;**

**}**

**if($x>$y) {**

**return $x . '<br/>';**

**}else{**

**return $y . '<br/>';**

**}**

**}**

**$a = 1;**

**$b = 2.5;**

**$c =1.9;**

**echo larger($a,$b);**

**echo larger($c,$a);**

**@ $larger = larger($d,$a);**

**var\_dump($larger);**

**if($larger === false) {**

**echo "error<br/>" ;**

**}**

**if($larger == 0){**

**echo "yes";**

**}**

**注意：测试false必须用 “===”，确保false不会跟0混淆。**

**false对应值0，true对应值1.**



**10. 实现递归**

**recursion.php ——使用递归将一个字符串颠倒**

**—————————————————————————————**

**<?php**

**function reverse\_r($str){**

**if(strlen($str) > 0){**

**reverse\_r(substr($str,1));**

**}**

**echo substr($str, 0,1);**

**return;**

**}**

**function reverse\_i($str){**

**for ($i=1; $i <= strlen($str) ; $i++) {**

**echo substr($str, -$i,1);**

**}**

**return;**

**}**

**reverse\_r('Hello');**

**echo '<br/>';**

**reverse\_i('Hello');**

**?>**

**————————————————————————————**



**reverse\_r() 是递归的**

**reverse\_i() 是循环的**

**reverse\_r()带有一个字符串作为参数，调用它时，它会用下面的参数多次调用自己：**

**reverse\_r('ello');**

**reverse\_r('llo');**

**reverse\_r('lo');**

**reverse\_r('o');**

**reverse\_r('');**

**每次调用，都会在服务器的内存中生成一段该函数代码的新副本，参数不同。**

**每次调用，都会测试传入的字符串的长度，当strlen()==0时，条件失败。**

**在应用领域，不推荐使用递归。**

**程序员常常忘记了给出递归的终止条件，这意味着函数会一直重复下去，直到服务器内存耗尽。**

**名称空间**

名称空间是一个抽象的容器，可以包含你所定义的函数、常量、类。

一个名称空间中的所有函数、类、常量，都将自动冠名为名称空间前缀。

非全路径的类、函数、常量名称，将在运行时解析，在查看全局空间之前，将先查看名称空间。