**WordPress模板基本文件**

style.css 样式表文件

index.php 主页文件

single.php 日志单页文件

page.php 页面文件

archvie.php 分类和日期存档页文件

searchform.php 搜索表单文件

search.php 搜索页面文件

comments.php 留言区域文件(包括留言列表和留言框)

404.php 404错误页面

header.php 网页头部文件

sidebar.php 网页侧边栏文件

footer.php 网页底部文件

**WordPress Header头部 PHP代码** (注: 也就是位于<head>和</head>之间的PHP代码)

<?php **bloginfo**(’name’); ?> 网站标题

<?php **wp\_title**(); ?> 日志或页面标题

<?php **bloginfo**(’stylesheet\_url’); ?> WordPress主题样式表文件style.css的相对地址

<?php **bloginfo**(’pingback\_url’); ?> WordPress博客的Pingback地址

<?php **bloginfo**(’template\_url’); ?> WordPress主题文件的相对地址

<?php **bloginfo**(’version’); ?> 博客的WordPress版本

<?php **bloginfo**(’atom\_url’); ?> WordPress博客的Atom地址

<?php **bloginfo**(’rss2\_url’); ?> WordPress博客的RSS2地址

<?php **bloginfo**(’url’); ?> WordPress博客的绝对地址

<?php **bloginfo**(’name’); ?> WordPress博客的名称

<?php **bloginfo**(’html\_type’); ?> 网站的HTML版本

<?php **bloginfo**(’charset’); ?> 网站的字符编码格式

**WordPress 主体模板 PHP代码**

<?php **the\_content**(); ?> 日志内容

<?php if(**have\_posts**()) : ?> 确认是否有日志

<?php while(**have\_posts**()) : **the\_post**(); ?> 如果有，则显示全部日志

<?php endwhile; ?> 结束PHP函数”while”

<?php endif; ?> 结束PHP函数”if”

<?php **get\_header**(); ?> header.php文件的内容

<?php **get\_sidebar**(); ?> sidebar.php文件的内容

<?php **get\_footer**(); ?> footer.php文件的内容

<?php **the\_time**(’m-d-y’) ?> 显示格式为”02-19-08″的日期

<?php **comments\_popup\_link**(); ?> 显示一篇日志的留言链接

<?php **the\_title**(); ?> 显示一篇日志或页面的标题

<?php **the\_permalink**() ?> 显示一篇日志或页面的永久链接/URL地址

<?php **the\_category**(’, ‘) ?> 显示一篇日志或页面的所属分类

<?php **the\_author**(); ?> 显示一篇日志或页面的作者

<?php **the\_ID**(); ?> 显示一篇日志或页面的ID

<?php **edit\_post\_link**(); ?> 显示一篇日志或页面的编辑链接

<?php **get\_links\_list**(); ?> 显示Blogroll中的链接

<?php **comments\_template**(); ?> comments.php文件的内容

<?php **wp\_list\_pages**(); ?> 显示一份博客的页面列表

<?php **wp\_list\_cats**(); ?> 显示一份博客的分类列表

<?php **next\_post\_link**(’ %link ‘) ?> 下一篇日志的URL地址

<?php **previous\_post\_link**(’%link’) ?> 上一篇日志的URL地址

<?php **get\_calendar**(); ?> 调用日历

<?php **wp\_get\_archives**() ?> 显示一份博客的日期存档列表

<?php **posts\_nav\_link**(); ?> 显示较新日志链接(上一页)和较旧日志链接（下一页）

<?php **bloginfo**(’description’); ?> 显示博客的描述信息

**其它的一些WordPress模板代码**

/%postname%/ 显示博客的自定义永久链接

<?php **the\_search\_query**(); ?> 搜索表单的值

<?php **\_e**(’Message’); ?> 打印输出信息

<?php **wp\_register**(); ?> 显示注册链接

<?php **wp\_loginout**(); ?> 显示登入/登出链接

<!–next page–> 在日志或页面中插入分页

<!–more–> 截断日志

<?php **wp\_meta**(); ?> 显示管理员的相关控制信息

<?php **timer\_stop**(1); ?> 显示载入页面的时间

<?php echo **get\_num\_queries**(); ?> 显示载入页面查询

**1. wordpress调用最新文章**

WordPress最新文章的调用可以使用一行很简单的模板标签wp\_get\_archvies来实现. 代码如下：

<?php get\_archives(‘postbypost’, 10); ?> (显示10篇最新更新文章)

或者

<?php wp\_get\_archives(‘type=postbypost&limit=20&format=custom’); ?>

后面这个代码显示你博客中最新的20篇文章，其中format=custom这里主要用来自定义这份文章列表的显示样式。具体的参数和使用方法你可 以参考官方的使用说明- wp\_get\_archvies。(fromat=custom也可以不要，默认以UL列表显示文章标题。)

补充: 通过WP的query\_posts()函数也能调用最新文章列表， 虽然代码会比较多一点，但可以更好的控制Loop的显示，比如你可以设置是否显示摘要。具体的使用方法也可以查看官方的说明。

**2. wordpress调用随机文章**

<?php  
$rand\_posts = get\_posts(‘numberposts=10&orderby=rand’);  
foreach( $rand\_posts as $post ) :  
?>  
<!–下面是你想自定义的Loop–>  
<li><a href=”<?php the\_permalink(); ?>”><?php the\_title(); ?></a></li>  
<?php endforeach; ?>

**3. wordpress调用最新留言**

下面是我之前在一个WordPress主题中代到的最新留言代码，具体也记不得是哪个主题了。该代码直接调用数据库显示一份最新留言。其中 LIMIT 10限制留言显示数量。绿色部份则是每条留言的输出样式。

<?php  
global $wpdb;  
$sql = “SELECT DISTINCT ID, post\_title, post\_password, comment\_ID,  
comment\_post\_ID, comment\_author, comment\_date\_gmt, comment\_approved,  
comment\_type,comment\_author\_url,  
SUBSTRING(comment\_content,1,30) AS com\_excerpt  
FROM $wpdb->comments  
LEFT OUTER JOIN $wpdb->posts ON ($wpdb->comments.comment\_post\_ID =  
$wpdb->posts.ID)  
WHERE comment\_approved = ’1′ AND comment\_type = ” AND  
post\_password = ”  
ORDER BY comment\_date\_gmt DESC  
LIMIT 10″;  
$comments = $wpdb->get\_results($sql);  
$output = $pre\_HTML; foreach ($comments as $comment) {  
$output .= “n<li>”.strip\_tags($comment->comment\_author)  
.”:” . ” <a href=”" . get\_permalink($comment->ID) .  
“#comment-” . $comment->comment\_ID . “” title=”on ” .  
$comment->post\_title . “”>” . strip\_tags($comment->com\_excerpt)  
.”</a></li>”;  
} $output .= $post\_HTML;  
echo $output;?>

**4.wordpress调用相关文章**

在文章页显示相关文章

<?php  
  
$tags = wp\_get\_post\_tags($post->ID);  
  
if ($tags) {  
  
$first\_tag = $tags[0]->term\_id;  
  
$args=array(  
  
‘tag\_\_in’ => array($first\_tag),  
  
‘post\_\_not\_in’ => array($post->ID),  
  
‘showposts’=>10,  
  
‘caller\_get\_posts’=>1  
  
);  
  
$my\_query = new WP\_Query($args);  
  
if( $my\_query->have\_posts() ) {  
  
while ($my\_query->have\_posts()) : $my\_query->the\_post(); ?>  
  
<li><a href=”<?php the\_permalink() ?>” rel=”bookmark” title=”<?php the\_title\_attribute(); ?>”><?php the\_title();?> <?php comments\_number(‘ ‘,’(1)’,'(%)’); ?></a></li>  
  
<?php  
  
endwhile;  
  
}  
  
}  
  
wp\_reset\_query();  
  
?>

**5.wordpress调用指定分类的文章**

<?php $posts = get\_posts( “category=4&numberposts=10″ ); ?>  
<?php if( $posts ) : ?>  
<ul><?php foreach( $posts as $post ) : setup\_postdata( $post ); ?>  
<li>  
<a href=”<?php the\_permalink() ?>” rel=”bookmark” title=”<?php the\_title(); ?>”><?php the\_title(); ?></a>  
</li>  
<?php endforeach; ?>  
</ul>  
<?php endif; ?>

**6.wordpress去评论者链接的评论输出**

<?php  
  
global $wpdb;  
  
$sql = “SELECT DISTINCT ID, post\_title, post\_password, comment\_ID,  
  
comment\_post\_ID, comment\_author, comment\_date\_gmt, comment\_approved,  
  
comment\_type,comment\_author\_url,  
  
SUBSTRING(comment\_content,1,14) AS com\_excerpt  
  
FROM $wpdb->comments  
  
LEFT OUTER JOIN $wpdb->posts ON ($wpdb->comments.comment\_post\_ID =  
  
$wpdb->posts.ID)  
  
WHERE comment\_approved = ’1′ AND comment\_type = ” AND  
  
post\_password = ”  
  
ORDER BY comment\_date\_gmt DESC  
  
LIMIT 10″;  
  
$comments = $wpdb->get\_results($sql);  
  
$output = $pre\_HTML;  
  
foreach ($comments as $comment) {  
  
$output .= “\n<li>”.strip\_tags($comment->comment\_author)  
  
.”:” . ” <a href=\”" . get\_permalink($comment->ID) .  
  
“#comment-” . $comment->comment\_ID . “\” title=\”on ” .  
  
$comment->post\_title . “\”>” . strip\_tags($comment->com\_excerpt)  
  
.”</a></li>”;  
  
}  
  
$output .= $post\_HTML;  
  
echo $output;?>

**7.wordpress调用含gravatar头像的评论输出**

<?php  
  
global $wpdb;  
  
$sql = “SELECT DISTINCT ID, post\_title, post\_password, comment\_ID, comment\_post\_ID, comment\_author, comment\_date\_gmt, comment\_approved,comment\_author\_email, comment\_type,comment\_author\_url, SUBSTRING(comment\_content,1,10) AS com\_excerpt FROM $wpdb->comments LEFT OUTER JOIN $wpdb->posts ON ($wpdb->comments.comment\_post\_ID = $wpdb->posts.ID) WHERE comment\_approved = ’1′ AND comment\_type = ” AND comment\_author != ‘郑 永’ AND post\_password = ” ORDER BY comment\_date\_gmt DESC LIMIT 10″;  
  
$comments = $wpdb->get\_results($sql);  
  
$output = $pre\_HTML;  
  
foreach ($comments as $comment) {  
  
$output .= “\n<li>”.get\_avatar(get\_comment\_author\_email(‘comment\_author\_email’), 18). ” <a href=\”" . get\_permalink($comment->ID) . “#comment-” . $comment->comment\_ID . “\” title=\”" . $comment->post\_title . ” 上的评论\”>”. strip\_tags($comment->comment\_author) .”： “. strip\_tags($comment->com\_excerpt) .”</a></li>”;  
  
}  
  
$output .= $post\_HTML;  
  
$output = convert\_smilies($output);  
  
echo $output;  
  
?>

上面代码把comment\_author的值改成你的ID，18是头像大小，10是评论数量。

**8.wordpress调用网站统计大全**

1、日志总数：

<?php $count\_posts = wp\_count\_posts(); echo $published\_posts = $count\_posts->publish;?>

2、草稿数目：

<?php $count\_posts = wp\_count\_posts(); echo $draft\_posts = $count\_posts->draft; ?>

3、评论总数：

<?php echo $wpdb->get\_var(“SELECT COUNT(\*) FROM $wpdb->comments”);?>

4、成立时间：

<?php echo floor((time()-strtotime(“2008-8-18″))/86400); ?>

5、标签总数：

<?php echo $count\_tags = wp\_count\_terms(‘post\_tag’); ?>

6、页面总数：

<?php $count\_pages = wp\_count\_posts(‘page’); echo $page\_posts = $count\_pages->publish; ?>

7、分类总数：

<?php echo $count\_categories = wp\_count\_terms(‘category’); ?>

8、链接总数：

<?php $link = $wpdb->get\_var(“SELECT COUNT(\*) FROM $wpdb->links WHERE link\_visible = ‘Y’”); echo $link; ?>

9、用户总数：

<?php $users = $wpdb->get\_var(“SELECT COUNT(ID) FROM $wpdb->users”); echo $users; ?>

10、最后更新：

<?php $last = $wpdb->get\_results(“SELECT MAX(post\_modified) AS MAX\_m FROM $wpdb->posts WHERE (post\_type = ‘post’ OR post\_type = ‘page’) AND (post\_status = ‘publish’ OR post\_status = ‘private’)”);$last = date(‘Y-n-j’, strtotime($last[0]->MAX\_m));echo $last; ?>

**9.wordpress判断语句**

is\_single()

判断是否是具体文章的页面

is\_single(’2′)

判断是否是具体文章（id=2）的页面

is\_single(’Beef Stew’)

判断是否是具体文章（标题判断）的页面

is\_single(’beef-stew’)

判断是否是具体文章（slug判断）的页面

comments\_open()

是否留言开启

pings\_open()

是否开启ping

is\_page()

是否是页面

is\_page(’42′)

id判断，即是否是id为42的页面

is\_page(’About Me’)

判断标题

is\_page(’about-me’)

slug判断

is\_category()

是否是分类

is\_category(’6′)

id判断，即是否是id为6的分类

is\_category(’Cheeses’)

分类title判断

is\_category(’cheeses’)

分类 slug判断

in\_category(’5′)

判断当前的文章是否属于分类5

is\_author()

将所有的作者的页面显示出来

is\_author(’1337′)

显示author number为1337的页面

is\_author(’Elite Hacker’)

通过昵称来显示当前作者的页面

is\_author(’elite-hacker’)

下面是通过不同的判断实现以年、月、日、时间等方式来显示归档

is\_date()

is\_year()

is\_month()

is\_day()

is\_time()

判断当前是否是归档页面

is\_archive()

判断是否是搜索

is\_search()

判断页面是否404

is\_404()

判断是否翻页，比如你当前的blog是http://domain.com 显示http://domain.com?paged=2的时候，这个判断将返 回真，通过这个函数可以配合is\_home来控制某些只能在首页显示的界面，

例如：

<?php if(**is\_single**()):?>

//这里写你想显示的内容，包括函数

<?php endif;?>

或者：

<?php if(**is\_home**() && !**is\_paged**() ):?>

//这里写你想显示的内容，包括函数

<?php endif;?>

10.wordpress 非插件调用评论表情

<!–smilies–>

<?php

function **wp\_smilies**() {

global $wpsmiliestrans;

if ( !**get\_option**(‘use\_smilies’) **or** (**empty**($wpsmiliestrans))) return;

$smilies = **array\_unique**($wpsmiliestrans);

$link=”;

**foreach** ($smilies as $key => $smile) {

$file = **get\_bloginfo**(‘wpurl’).’/wp-includes/images/smilies/’.$smile;

$value = ” “.$key.” “;

$img = “<img src=\”{$file}\” alt=\”{$smile}\” />”;

$imglink = **htmlspecialchars**($img);

$link .= “<a href=\”#commentform\” title=\”{$smile}\” onclick=\”document.**getElementById**(‘comment’).value += ‘{$value}’\”>{$img}</a> ”;

}

echo ‘<div>’.$link.’</div>’;

}

?>

<?php **wp\_smilies**();?>

<!–smilies—>

将以上代码复制到 comments.php 中合适的位置。

WordPress是目前十分流行的独立博客程序，因傻瓜化安装和使用，其在网民中的应用已近乎普及。但也因为很多新入门的用户几乎对WordPress 程序没有任何了解，造成使用中碰到问题无法解决，求助也十分不易。而且，根据本站的经验，WordPress用户学习了解并掌握一些基本的WordPress知识尤其是WordPress中功能强大使用方便的函数会极大地方便自己的应用，从而定制一个自己心仪的独立博客。

      在这篇文章中，本站试着将自己在长期的WordPress生涯中摸索积累的WordPress函数以自己的语言与朋友们分享，希望这一系列文章能够为朋友们使用WordPress程序带来方便。

      在讲述下面的WordPress函数之前，我们需要明确这样一点，所有的WordPress函数都是已经定义好的PHP函数，它们都需要写在PHP 语句中才能执行；而且，所有这些函数在PHP语句中都应以分号”;”结尾。其实，为了用好这些函数，为你的WordPress服务，你最好具有 PHP语言的基础。

**1.bloginfo()**  
顾名思义，该函数主要用来显示博客信息；而且根据参数的不同，可以用来显示博客信息中的不同部分。常用的有以下几种：  
bloginfo(’name’) 显示博客题名，如“本站”；默认（不写参数）输出该项；  
bloginfo(’description’) 显示博客描述部分，如“路漫漫其修远兮，吾将上下而求索”；  
bloginfo(’url’) 输出博客URL地址，如[http://www.qiusuoge.com](http://www.qiusuoge.com/)；  
bloginfo(’rss2\_url’) 显示博客的RSS2.0 feed地址，如<http://www.qiusuoge.com/feed>；  
bloginfo(’template\_url’) 用来获取WordPress博客的模板地址；  
bloginfo(’charset’) 显示博客的编码方式，如“UTF-8”；  
一种常见的使用bloginfo()函数的组合如下：  
” title=”” >  
这种形式通常会用来添加博客的底部信息，如Copyright @ 本站，经常在主题模板中使用。  
需要注意的是bloginfo()函数只能输出显示这些参量，如果你想在PHP语句中使用得到的这些值，则需使用get\_bloginfo()函数，该函数和bloginfo()使用相同的参数，获得相同的结果。

**2.wp\_title()**  
该函数用来显示页面的标题，如在文章页面，则显示文章标题；在分类页面，则显示分类名称；等等。  
wp\_title()函数可以跟三个参数，即wp\_title(’separator’,echo,seplocation)，其中 separator是title和其余部分之间的分割符号，默认是>>；echo是个bool变量，取true显示标题，取false则将标题作为一个PHP参量返回；seplocation定义分隔符的位置，取right定义分隔符在标题后面，取其他任何值，都表示将分隔符放在标题前面。

**3.wp\_get\_archives()**  
该函数用来获取博客的文章存档，通过设置函数的参数，可以按各种方式获取，如按月，按年等等。  
wp\_get\_archives()函数后面同样可以跟多种参数，只不过所有参数都需要使用&连接，并放在单引号(’)中以字符串方式传递给函数，形如wp\_get\_archives(’type=monthly&format=html& show\_post\_count=1&limit=10′)。  
如上的参数意义描述如下：  
type=monthly表示按月显示文章存档，可以使用yearly、daily、weekly等代替monthly表示按年、日、以及周显示文章存档；  
# format=html表示使用通常的HTML中格式化文章列表；  
show\_post\_count=1表示在文章存档后面显示属于该类别（年、月等）的文章数量，该参量是个bool值；  
limit=10表示显示的文章存档的最大数量为10，超过次数，则超出部分不显示；  
尽管参数稍多，显得略为复杂，但其实只需注意type、show\_post\_count以及limit等三个参量即可。

**4.wp\_list\_categories()**  
和wp\_get\_archives()函数类似，wp\_list\_categories()函数用来获取博客文章的分类信息，并可以通过设置适当的函数参数，将其显示出来。该函数的参数和wp\_get\_archives()函数类似，都需要使用&连接，放在单引号 (’)中以字符串方式传递。形如wp\_get\_archives(’orderby=name&order=ASC& show\_count=1&use\_desc\_for\_title=1&feed=订阅&exclude=2,5& number=10′)。  
如上示例中，函数各参数的意义如下：  
orderby=name表示按照分类名称的字母先后顺序显示分类信息，可以将name换为ID等；  
order=ASC表示按照分类名称的字母的升序显示分类信息，将ASC改为DESC表示按降序；  
show\_count=1在每个分类名称后面显示属于该分类的文章数；  
use\_desc\_for\_title=1使用该分类的描述信息为每个分类名称超链接添加一个title属性；  
feed=订阅：在每个分类信息旁边添加一个名为“订阅”的超链接，提供该分类的RSS订阅；  
exlude=2,5：在显示的分类中去除ID为2和5的分类；也可以用include=2,5表示只显示ID为2和5的分类；  
number=10：表示只显示最先的10个分类。

**5.get\_the\_category()**  
get\_the\_category()函数用来返回当前文章所属的类别的若干属性所组成的一个数组，该数组包括以下内容：  
cat\_ID：当前类别的ID(也可以写作’term\_id’)；  
cat\_name：当前类别的名称(也被写作’name’)；  
category\_description：当前分类的描述(也可以写作’description’)；  
category\_count：属于当前分类的文章数量(也被写作’count’)。  
具体的使用方法，我们通过下面的几个句子来说明：  
形如get\_the\_category()->cat\_ID的语句，返回当前文章所属分类的ID号；  
形如get\_the\_category()->description的语句，返回当前文章所属分类的描述；等等。

**6.the\_category()**  
该函数返回当前文章所属的类别名称，而且是文章类别的超链接形式。  
默认的无参数形式the\_category()直接以超链接形式显示类别名称，显示为：精品推介；  
可以在函数中跟上分隔符等参数来格式化输出，如the\_category(’-’)，若当前文章属于两个以上分类，可以显示这样的形式：精品推介-经验知识；如只属于一个分类，则显示为这样的形式：精品推介。

**7.category\_description()**  
该函数以分类的ID为输入，得到该分类的描述。常和echo、get\_the\_category()配合使用，将当前分类描述输出：  
echo category\_description(get\_the\_category()->cat\_ID);  
如上语句，get\_the\_category()得到保存有当前分类信息的一个数组；cat\_ID为该数组中该分类的ID；将该ID输入给category\_description()函数，即可得到该分类的描述；然后使用echo将其输出。  
但经本站试验，使用如下的语句可以实现和上面语句相同的功能：  
echo category\_description();  
这可能是因为该函数在默认无参数输入的情况下会输出当前分类描述的结果。

**8.is\_home()**  
is\_home()用以判断当前显示的博客页面是否是博客首页，返回的是一个Bool值。如果是在首页，则返回TRUE；否则返回FALSE。  
该函数常用来控制博客侧边栏的显示方式，经常使用如下代码段：  
 **9.is\_archive()**  
is\_archive()用以判断当前显示的内容是否是博客存档页面，比如按日期的存档，或者按分类的存档，等等；其和is\_home()函数一样，返回一个Bool值。

**10.is\_page()**  
is\_page()函数判断当前显示的内容是否是博客的独立页面（page），如“本站WordPress”、“关于本站”等页面；它也返回一个Bool值。  
我们可以在模板中通过该函数判断当前是否是一个独立页面，从而决定是否为当前显示的文章显示发布时间等等。

**11.is\_paged()**  
该函数用以判断当前文章是否因为内容过多而分页显示；需要注意的是，如果你在写文章时手动添加了标签，来强制分页的话，该函数并不会因此而返回TRUE。

**12.is\_page\_template()**  
is\_page\_template()函数需要跟一个参数，通常以如下方式使用：  
is\_page\_template(’guestbook.php’);  
藉此判断当前显示的独立页面（page）是否使用了参数所示的模板guestbook.php；如果不跟参数，函数返回当前独立页面是否使用了模板。

**13.is\_single()**  
is\_single()用以判断当前显示的页面内容是否是一篇单独的文章。其后面可以跟三种参数，一种是文章ID；一种是文章题目（title）；一种是文章名称（slug，文章题目的一种简短说明形式）；或者可以将三种参数组合使用，藉此来判断当前页面内容是否是具体的某篇文章。

**一个简单的例子如下，我们可以通过如下几种方式判断当前显示的内容是否是本文：**

is\_single(’808′ );is\_single(’用好WordPress不可不知的函数（二）’);is\_single(’functions-must-known- using-wordpress-second’);is\_single(’808′,’用好WordPress不可不知的函数（二）’,’functions-must-known-using-wordpress-second’);

在这里，本站需要对上述最后一种方式做些说明：该函数后跟三个参数时，有优先级，如果第一个参数符合条件，则返回TRUE；否则，则返回FALSE；貌似后面的参数并没有什么意义。

**14.is\_category()**  
该函数用以判断当前显示的页面内容是否是一个分类页面，如网站相关；其中无需参数。函数返回一个Bool值。

**15.is\_tag()**  
is\_tag()用以判断当前显示的页面是否是一个标签页面，比如WordPress；其后也不需要跟参数。该函数同样返回一个Bool值。

**16.is\_date()**  
此函数用以判断当前显示的内容是否为按时间归档的页面，比如2009年四月，或者2009年4月8日，等等。

**17.is\_day()、is\_month()、is\_year()**  
这些函数用以判断当前显示的内容是否为按天、按月、按年份归档的页面。它们和is\_date()类似，只不过将归档时间更具体化而已。

**18.is\_author()**  
该函数用以判断当前显示的内容是否为以作者名归档的页面，比如本站的admin作者页面。

**19.is\_admin()**  
is\_admin()函数用以判断当前是否在控制面板页面，或者管理员面板页面。

**20.get\_bloginfo()**  
该函数和我们前面的文章用好WordPress不可不知的函数（一）中介绍的bloginfo()函数实现近乎相同的功能。主要用来显示博客的信息；而且根据后跟参数的不同，会输出博客的不同信息。  
其后不跟参数时，get\_bloginfo()可以显示博客名称，形如“本站”；  
后跟其他参数时，可以显示对应的信息，比如get\_bloginfo ( ‘description’ )用以显示博客描述信息；  
其他还可以使用的参数包括name、url、wpurl以及admin\_email等等。但因为其与bloginfo()函数实现相同的结果，所以，在bloginfo能够实现的情况下，本站不推荐使用get\_bloginfo()函数。

**21.query\_posts()**  
query\_posts()函数结合适当的参数用来控制哪些文章会在页面上显示。  
形如query\_posts(”cat=3,6&cat=-5,-10″)表示取分类ID为3和6的文章显示，不取分类ID为5和10的文章显示；  
形如query\_posts(”order=ASC&showposts=10&offset=1&orderby=date&posts\_per\_page=5″)意义如下：  
order=ASC表示按照升序排列，取为DESC则表示按降序；  
showposts=10则表示获取10篇文章；  
offset=1表示取最新的文章；  
orderby=date表示将文章按照日期排序；  
posts\_per\_page=5表示每页显示5篇文章。  
需要注意的是该函数只是将文章内容从MySQL数据库中查询出来，要将其显示，还需要与其他语句配合，比如一个经常在侧边栏中使用的形式如下：  
如上的这段代码用以在侧边栏的指定位置上显示最新的5篇文章。

query\_posts()函数后面可以跟众多种类的参数，功能十分强大，在此，我们不进行过多介绍。如果时间和精力允许，本站会在以后的文章中专门撰文详细地为朋友们做一介绍。  
 **22.get\_posts()**  
该函数和query\_posts()函数功能大体相同，都是用来从数据库中查询并得到符合某条件的文章。不过get\_posts()函数的使用有一个固定的形式，如下：

” id=”post-”>

即首先使用get\_posts()函数查询得到文章数据，然后使用形如foreach($lastposts as $post) : setup\_postdata($post);的循环将查询得到的文章内容显示出来。  
get\_posts()函数使用和query\_posts()函数相同的参数，在此我们也不再作详细介绍。

**23.wp\_list\_cats()**  
该函数和前文用好WordPress不可不知的函数（一）中wp\_list\_categories()函数实现相同的功能，不过在最新的WordPress版本中，此函数已经被弃用，其功能为wp\_list\_categories()函数完全取代。

**24.get\_calendar()**  
get\_calendar()函数用以在WordPress上显示日历，日历样式和使用widget显示的效果相同，如下图所示：

其后可以跟一个BOOL参数，用以控制日历上方星期的显示样式。但经本站测试，在中文状态下，使用TRUE或FALSE参数，日历的显示效果并没有什么区别。

**25.wp\_list\_bookmarks()**  
该函数用来显示博客的友情链接，并可以使用各种参数来控制显示的数量、种类以及样式等等。  
形如wp\_list\_bookmarks(’title\_li=&categorize=0&orderby=rand&include=41,40,37,54′);的形式，意义解释如下：  
title\_li=&categorize=0是一种通常使用的组合，意为不显示WordPress后台控制面板中设置的友情链接标题，但所有友链都按照设置的分类显示出来；单独的一个title\_li=还可以用来设置被显示友链的类别名称；  
orderby=rand设置友链的显示方式为随机顺序，当然，还可以设置为其他的方式，比如id、url、name等等；  
include=41,40,37,54表示只显示ID为这四个数字的四个友链；与此对应，还支持使用exclude，表示不显示ID为多少的友链。  
# 此外，该函数还经常用到的参数有before和after，用来设置每个链接前后的文字，默认的是和  
标记。

**26.get\_links()、wp\_get\_links()**  
这两个函数实现和25.wp\_list\_bookmarks()相同的功能，不过此二函数已在WordPress升级的过程中为25.wp\_list\_bookmarks()所取代。

**27.wp\_list\_pages()**  
该函数以页面名称的超链接形式显示WordPress博客内的所有页面，经常用来建立顶端导航页面，或用来修饰侧边栏。  
形如wp\_list\_pages(’title\_li=&sort\_column=menu\_order&include=12,25,38,57&depth=1&’);的函数调用，各参数意义如下：  
title\_li=用来设置所有显示页面的一个总名称；后面没有参数值时，表示不显示名称；  
sort\_column=menu\_order用来设置页面的显示顺序，表示按照WP后台设置的各页面顺序显示，其他的常用顺序设置可能还包括post\_title、post\_date、ID等等；  
include=12,25,38,57表示只显示ID为这四个数值的四个页面；同样，可以使用exclude来排除相应ID的页面；  
depth=1表示只显示父页面，对所有子页面不予显示；其他数值还包括默认的0，表示显示所有页面（子页面有缩进）；-1显示所有页面（子页面无缩进）；等等。  
此外，该函数可能会用到的属性还包括link\_before和link\_after，用于设置显示的页面链接前后的字符。

**28.wp\_tag\_cloud()**  
顾名思义，wp\_tag\_cloud()函数用来显示WordPress博客的标签云。  
一种形如wp\_tag\_cloud(’smallest=8&largest=22&number=30&orderby=count’);的函数调用，各参数的意义如下：  
smallest=8用来设置标签云中显示出来的所有标签中，计数最少（最少文章使用）的标签的字体大小为8；  
largest=22用来设置标签云的所有标签中，计数最多（最多文章使用）的标签的字体大小为22；  
number=30设置标签云中显示的最多标签数量为30；  
orderby=count设置标签云中标签的排序方式为计数（默认），而不是名称（相应参数为name，widget调用时的默认值）。  
其他常用的参数还包括include和exclude，用来设置在标签云中是否包含或去除ID为某数字的标签。

**29.wp\_register()**  
wp\_register()函数用以向管理员显示“站点管理”超链接；或者当WP博客开放了注册时，向未登陆的用户显示“注册”超链接。  
该函数不需要什么参数，唯一可能用到的参数形式如wp\_register(’前’,’后’)，可以在如上显示的超链接文字的前后分别显示一个“前”字和一个“后”字。当然，你可以据此发挥想象力来个性化自己网站的管理或注册链接。

**30.wp\_loginout()**  
该函数用来在指定位置显示一个“登录”链接；当然，如果你已经登录过了，则会相应地显示一个“退出”链接。此函数后面不使用任何参数，所以无法进行灵活的自定义。  
不过如果你想自定义自己的WP博客的登录或退出链接文字的话，还是可以使用下面31中 本站 介绍的函数wp\_logout\_url()和wp\_login\_url()。

**31.wp\_logout\_url()、wp\_login\_url()**  
使用如上30中的函数尽管可以方便地为WP博客设置登录、退出链接，但自定义不够灵活。所以，从WordPress 2.7版本开始，提供了这里的两个函数。它们分别用来获取WP博客退出或登录超链接，然后，我们使用获得的超链接即可编写如下代码，来对WP博客的登录和退出链接进行灵活设置：  
”>点击这里退出  
”>点击这里登录  
当然，要想实现完美的效果，还需要对访客的登录状态进行判断，使用一个if语句，根据登录状态显示相应的菜单项。

**32.wp\_meta()**  
该函数通常会紧跟如上29、30中的函数后面，其具体在直观显示上没有什么异样，貌似是WP主题为WP插件留下的API Hook，本站建议朋友们在如上函数后面跟上这一函数。

**33.get\_recent\_posts()**  
该函数只有当你安装了中文WordPress工具箱之后，才能使用。其作用是用来获取最新日志，函数原型如下：  
# get\_recent\_posts($no\_posts = 5, $before = ‘+ ‘, $after = ‘  
’, $show\_pass\_post = false, $skip\_posts = 0)  
可以使用$no\_posts控制显示文章数量，$before和$after的意义和前面函数中相同；至于后两个参数，一般不必设置，直接取默认值即可。  
不过因为该函数与WordPress内置的get\_posts()和query\_posts()函数功能重复，所以通常情况下很少使用。

**34.get\_recent\_comments()**  
其实安装了如上的中文WordPress工具箱之后，最常使用的是这个函数，因为WordPress程序本身没有内置获取最新评论的函数。该函数原型如下：  
# get\_recent\_comments($no\_comments = 5, $before = ‘ ‘, $after = ‘  
’, $show\_pass\_post = false)  
意义显然，和上面函数类似，本站此处不再赘言。

**35.get\_recentcomments()**  
该函数是在安装了WP-RecentComments插件之后才具有的功能，与如上34中的函数类似。  
该函数原型如下：  
get\_recentcomments(int num, int size)

**36.wp\_get\_post\_tags()**  
该函数用来在某个文章页面或者根据某篇文章的ID来获取该文章的tag，获取的结果被放置到一个tag数组中。一个常见的使用方式如下：  
if (is\_single()){$keywords = “”;$tags = wp\_get\_post\_tags($post->ID);foreach ($tags as $tag ) {$keywords = $keywords . $tag->name . “,”;}echo $keywords;}  
首先判断是否是单文章页面，如果是，则据当前文章的ID（$post->ID）来获取当前文章的tag，然后取得其name（$tag->name），并将其组合输出。

**37.single\_cat\_title()、single\_tag\_title()**  
如名所言，这两个函数用来获取分类页面和tag页面的title，其通常的使用方式如：

然而，除此之外，single\_cat\_title()还可以用来在tag页面上获取当前页面的title；但single\_tag\_title()却不可用于获取分类页面的title。

**38.get\_settings()、get\_option()**  
此二函数与前文用好WordPress不可不知的函数（三）中函数20.get\_bloginfo()类似，使用方法也相同，可以通过后跟各种参数来获取WordPress博客的相关信息。  
比如如下的调用方式：  
get\_settings(’name’)或get\_option(’name’)  
可以用来获取当前WordPress博客的标题。

**39.wp\_head()**  
该函数与前文32.wp\_meta()相同，是WP主题为WP插件留下的API Hook。

**40.get\_header()、get\_footer()、get\_sidebar()和comments\_template()**  
这几个函数是用来在WordPress主题中获取并包含相应的文件的。比如：  
get\_header()用来包含当前主题文件夹下的header.php；  
get\_footer()用来包含主题文件夹下的footer.php；  
get\_sidebar()用来包含主题文件夹下的sidebar.php；  
comments\_template()用来包含comments.php。  
需要注意的一点是，如果当前主题文件夹下缺少对应的文件，则函数会使用wp-content/themes/default/文件夹下的对应文件代替。  
此外，以上函数后面都不能跟参数，只有get\_sidebar()例外，因为一个主题中可以使用多个sidebar。形如get\_sidebar(’up’)的调用方法可将sidebar-up.php侧边栏模板文件包含到主题中。  
除了以上几个函数之外，在主题中如果想包含一个具体的文件，还可以使用如下方式：  
include(TEMPLATEPATH . ‘/\*\*\*.php’)  
如上的函数形式可以将当前主题文件夹下名为\*\*\*.php的文件包含进来；其中TEMPLATEPATH是当前主题文件夹地址的一个引用（不含末尾的/，所以需要添加上）。

**41.have\_posts()、the\_post()**  
这两个函数的使用范围有限，通常在WordPress的循环中使用，用以获取所有文章。其固定使用形式如下：  
此处显示文章此处显示未找到文章时的信息，比如404相关  
另一种常见的形式是将如上代码中的前两行组合起来（其他地方不变）：

该形式通常会在模板的index.php、archive.php或者single.php等页面使用。除此之外的其他地方，我们通常不会看到此二函数的身影。

**42.the\_title()、the\_title\_attribute()**  
the\_title()函数主要用来获取当前文章的title，其后可以跟上三个参数（可全部省略，取默认值），调用形式如下：

参数before用来设置在获取的title前面显示的字符内容；after用来设置title其后显示的内容；而display是一个Bool值，用于控制获取的title是否显示出来。  
the\_title\_attribute()函数与the\_title()类似，其使用方法如下：  
the\_title\_attribute(’before=前&after=后&echo=true’)  
其中before=和after=分别用于设置title前面和后面显示的字符；echo=true或者false用户设置获取的title字符串是否显示出来。  
形如或the\_title\_attribute(’before=当前文章&after=的评论：&echo=true’)的调用形式将会显示如下的结果：  
当前文章用好WordPress不可不知的函数的评论：

**43.single\_post\_title()、single\_tag\_title()、single\_cat\_title()**

44.the\_ID()  
该函数后面不跟任何参数，使用如下所示的调用方式：

用于获取并显示当前文章页面的ID号。不过需要特别注意的一点是，该函数只能在WordPress的大循环内使用，在其他地方使用可能也会显示ID号，但显示的内容始终不会随文章而改变。  
此外，该函数通常还会在如下所示的CSS结构中使用：  
可为博客中不同的作者设置不同的title样式，以示区分。

**45.get\_the\_ID()**  
该函数与44.the\_ID()函数实现完全相同的功能，目前WordPress官方也没有提供该函数的使用说明。你可以参阅如上44中对the\_ID()函数的介绍。特别提醒一点，该函数与the\_ID()类似，也只能在WordPress的大循环中才能正确使用。

**46.the\_time()、get\_the\_time()**  
the\_time()用来获取并显示当前文章发布的时间，和上面几个函数类似，此函数也是只能在WordPress的大循环中使用。  
该函数后面可以跟控制日期或时间格式的参数，常用的参数形式如下：  
如的调用形式显示效果为：六月13, 2009（英文状态下显示June 13, 2009）；  
如的调用形式显示效果为：7:09 下午（英文状态下显示7:09 pm）；  
如的调用形式显示效果为：19:09。  
事实上，除了使用the\_time()函数之外，WordPress还提供了一个具有类似功能的get\_the\_time()函数。该函数除了不具有the\_time()函数的显示功能之外，其余功能二者完全相同。使用get\_the\_time()函数时，如欲将获取的时间显示出来，需要使用专用语句。  
下面，我们籍此机会来简单了解一下WordPress中时间的格式。在WordPress中，通常用于控制时间格式的有一下字符：l, F, j, S, Y, G, g, i, a等等，其详细意义如下：  
l（小写L）用来显示一周之中每一天的名称，比如星期六，或者在英文中显示Saturday；  
F用来显示月份名称，比如六月，或者June；  
j用来显示一月之中的某一天，比如13；  
Y用来以4位数字形式显示年份，使用y则以末两位数字显示年份，比如2009或09；  
G, g, i, a等四个字符通常组合使用，如前例子，有两种形式：  
g:i a以形如7:09 下午或7:09 pm的形式显示时间；  
G:i以形如19:09的24小时进制形式显示时间。  
S通常紧跟在j后面，表示是否在一月之中某天之后添加英文后缀（st, nd, th等）。