**菜鸟学PHP之Smarty入门**

2009年4月17日[陈皓](http://coolshell.cn/articles/author/haoel)[发表评论](http://coolshell.cn/articles/559.html#respond)[阅读评论](http://coolshell.cn/articles/559.html#comments)4,729 人阅读

　　刚开始接触模版引擎的 PHP 设计师，听到 Smarty 时，都会觉得很难。其实笔者也不例外，碰都不敢碰一下。但是后来在剖析 XOOPS 的程序架构时，开始发现 Smarty 其实并不难。只要将 Smarty 基础功练好，在一般应用上就已经相当足够了。当然基础能打好，后面的进阶应用也就不用怕了。  
　　  
　　这篇文章的主要用意并非要深入探讨 Smarty 的使用，这在官方使用说明中都已经写得很完整了。笔者仅在此写下一些自己使用上的心得，让想要了解 Smarty 却不得其门而入的朋友，可以从中得到一些启示。就因为这篇文章的内容不是非常深入，会使用 Smarty 的朋友们可能会觉得简单了点。  
　　  
　  
　　**Smarty介绍  
　　  
　　什么是模版引擎**  
　　  
　　不知道从什么时候开始，有人开始对 HTML 内嵌入 Server Script 觉得不太满意。然而不论是微软的 ASP 或是开放源码的 PHP，都是属于内嵌 Server Script 的网页伺服端语言。因此也就有人想到，如果能把程序应用逻辑 (或称商业应用逻辑) 与网页呈现 (Layout) 逻辑分离的话，是不是会比较好呢？  
　　  
　　其实这个问题早就存在已久，从交互式网页开始风行时，不论是 ASP 或是 PHP 的使用者都是身兼程序开发者与视觉设计师两种身份。可是通常这些使用者不是程序强就是美工强，如果要两者同时兼顾，那可得死掉不少脑细胞…  
　　  
　　所以模版引擎就应运而生啦！模版引擎的目的，就是要达到上述提到的逻辑分离的功能。它能让程序开发者专注于资料的控制或是功能的达成；而视觉设计师则可专注于网页排版，让网页看起来更具有专业感！因此模版引擎很适合公司的网站开发团队使用，使每个人都能发挥其专长！  
　　  
　　就笔者接触过的模版引擎来说，依资料呈现方式大概分成：需搭配程序处理的模版引擎和完全由模版本身自行决定的模版引擎两种形式。  
　　  
　　在需搭配程序处理的模版引擎中，程序开发者必须要负责变量的呈现逻辑，也就是说他必须把变量的内容在输出到模版前先处理好，才能做 assign 的工作。换句话说，程序开发者还是得多写一些程序来决定变量呈现的风貌。而完全由模版本身自行决定的模版引擎，它允许变量直接 assign 到模版中，让视觉设计师在设计模版时再决定变量要如何呈现。因此它就可能会有另一套属于自己的模版程序语法 (如 Smarty) ，以方便控制变量的呈现。但这样一来，视觉设计师也得学习如何使用模版语言。  
　　  
　　模版引擎的运作原理，首先我们先看看以下的运行图：  
　　 　  
　　一般的模版引擎 (如 PHPLib) 都是在建立模版对象时取得要解析的模版，然后把变量套入后，透过 parse() 这个方法来解析模版，最后再将网页输出。  
　　 　  
　　对 Smarty 的使用者来说，程序里也不需要做任何 parse 的动作了，这些 Smarty 自动会帮我们做。而且已经编译过的网页，如果模版没有变动的话， Smarty 就自动跳过编译的动作，直接执行编译过的网页，以节省编译的时间。  
　　  
　　**使用Smarty的一些概念**  
　　  
　　在一般模版引擎中，我们常看到区域的观念，所谓区块大概都会长成这样：  
　　<!– START : Block name –>  
　　区域内容  
　　<!– END : Block name –>  
　　  
　　这些区块大部份都会在 PHP 程序中以 if 或 for, while 来控制它们的显示状态，虽然模版看起来简洁多了，但只要一换了显示方式不同的模版， PHP 程序势必要再改一次！  
　　  
　　在 Smarty 中，一切以变量为主，所有的呈现逻辑都让模版自行控制。因为 Smarty 会有自己的模版语言，所以不管是区块是否要显示还是要重复，都是用 Smarty 的模版语法 (if, foreach, section) 搭配变量内容作呈现。这样一来感觉上好象模版变得有点复杂，但好处是只要规划得当， PHP 程序一行都不必改。  
　　  
　　由上面的说明，我们可以知道使用Smarty 要掌握一个原则：将程序应用逻辑与网页呈现逻辑明确地分离。就是说 PHP 程序里不要有太多的 HTML 码。程序中只要决定好那些变量要塞到模版里，让模版自己决定该如何呈现这些变量 (甚至不出现也行) 。  
　　  
　　**Smarty的基础  
　　  
　　安装Smarty**  
　　  
　　首先，我们先决定程序放置的位置。  
　　  
　　[Windows](http://windows.chinaitlab.com/)下可能会类似这样的位置：「 d:\appserv\web\demo\ 」。  
　　  
　　Linux下可能会类似这样的位置：「 /home/jaceju/public\_html/ 」。  
　　  
　　到Smarty的官方网站[下载](http://download.chinaitlab.com/)最新的Smarty套件：[http://smarty.php.net](http://smarty.php.net/)。  
　　  
　　解开 Smarty 2.6.0 后，会看到很多档案，其中有个 libs 资料夹。在 libs 中应该会有 3 个 class.php 檔 + 1 个 debug.tpl + 1 个 plugin 资料夹 + 1 个 core 资料夹。然后直接将 libs 复制到您的程序主资料夹下，再更名为 class 就可以了。就这样？没错！这种安装法比较简单，适合一般没有自己主机的使用者。  
　　  
　　至于 Smarty 官方手册中为什么要介绍一些比较复杂的安装方式呢？基本上依照官方的方式安装，可以只在主机安装一次，然后提供给该主机下所有设计者开发不同程序时直接引用，而不会重复安装太多的 Smarty 复本。而笔者所提供的方式则是适合要把程序带过来移过去的程序开发者使用，这样不用烦恼主机有没有安装 Smarty 。  
　　  
　　**程序的资料夹设定**  
　　  
　　以笔者在[Windows](http://windows.chinaitlab.com/)安装Appserv为例，程序的主资料夹是「d:\appserv\web\demo\」。安装好Smarty后，我们在主资料夹下再建立这样的资料夹：  
　　 　  
　　在 Linux 底下，请记得将 templates\_c 的权限变更为 777 。Windows 下则将其只读取消。  
　　  
　　**第一个用Smarty写的小程序**  
　　  
　　我们先设定 Smarty 的路径，请将以下这个档案命名为 main.php ，并放置到主资料夹下：  
　　  
　　main.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | <?php  include "class/Smarty.class.php";  define('\_\_SITE\_ROOT', 'd:/appserv/web/demo'); // 最后没有斜线  $tpl = new Smarty();  $tpl->template\_dir = \_\_SITE\_ROOT . "/templates/";  $tpl->compile\_dir = \_\_SITE\_ROOT . "/templates\_c/";  $tpl->config\_dir = \_\_SITE\_ROOT . "/configs/";  $tpl->cache\_dir = \_\_SITE\_ROOT . "/cache/";  $tpl->left\_delimiter = '<{';  $tpl->right\_delimiter = '}>';  ?> |

　　照上面方式设定的用意在于，程序如果要移植到其它地方，只要改 \_\_SITE\_ROOT 就可以啦。 (这里是参考 XOOPS 的 )  
　　  
　　Smarty 的模版路径设定好后，程序会依照这个路径来抓所有模版的相对位置 (范例中是 ‘d:/appserv/web/demo/templates/’ ) 。然后我们用 display() 这个 Smarty 方法来显示我们的模版。  
　　  
　　接下来我们在 templates 资料夹下放置一个 test.htm：(扩展名叫什么都无所谓，但便于视觉设计师开发，笔者都还是以 .htm 为主。)  
　　  
　　templates/test.htm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=big5">  <title><{$title}></title>  </head>  <body>  <{$content}>  </body>  </html> |

　　现在我们要将上面的模版显示出来，并将网页标题 ($title) 与内容 ($content) 更换，请将以下档案内容命名为 test.php ，并放置在主资料夹下：  
　　  
　　test.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <?php  require "main.php";  $tpl->assign("title", "测试用的网页标题");  $tpl->assign("content", "测试用的网页内容");  // 上面两行也可以用这行代替  // $tpl->assign(array("title" => "测试用的网页标题", "content" => "测试用的网页内容"));  $tpl->display('test.htm');  ?> |

　　请打开浏览器，输入 http://localhost/demo/test.php 试试看(依您的环境决定网址)，应该会看到以下的画面：  
　　 　  
　　再到 templates\_c 底下，我们会看到一个奇怪的资料夹 (%%179) ，再点选下去也是一个奇怪的资料夹 (%%1798044067) ，而其中有一个档案：  
　　  
　　templates\_c/%%179/%%1798044067/test.htm.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <?php /\* Smarty version 2.6.0, created on 2003-12-15 22:19:45 compiled from test.htm \*/ ?>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=big5">  <title><?php echo $this->\_tpl\_vars['title']; ?></title>  </head>  <body>  <?php echo $this->\_tpl\_vars['content']; ?>  </body>  </html> |

　　没错，这就是 Smarty 编译过的档案。它将我们在模版中的变量转换成了 PHP 的语法来执行，下次再读取同样的内容时， Smarty 就会直接抓取这个档案来执行了。  
　　  
　　最后我们整理一下整个 Smarty 程序撰写步骤：  
　　  
　　Step 1. 加载 Smarty 模版引擎。  
　　  
　　Step 2. 建立 Smarty 对象。  
　　  
　　Step 3. 设定 Smarty 对象的参数。  
　　  
　　Step 4. 在程序中处理变量后，再用 Smarty 的 assign 方法将变量置入模版里。  
　　  
　　Step 5. 利用 Smarty 的 display 方法将网页秀出。  
　　  
　　**如何安排你的程序架构**  
　　  
　　上面我们看到除了 Smarty 所需要的资料夹外 (class 、 configs 、 templates 、 templates\_c) ，还有两个资料夹： includes 、 modules 。其实这是笔者模仿 XOOPS 的架构所建立出来的，因为 XOOPS 是笔者所接触到的程序中，少数使用 Smarty 模版引擎的架站程序。所谓西瓜偎大边，笔者这样的程序架构虽没有 XOOPS 的百分之一强，但至少给人看时还有 XOOPS 撑腰。  
　　  
　　includes 这个资料夹主要是用来放置一些 function 、 sql 檔，这样在 main.php 就可以将它们引入了，如下：  
　　  
　　main.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <?php  include "class/Smarty.class.php";  define('\_\_SITE\_ROOT', 'd:/appserv/web/demo'); // 最后没有斜线  // 以 main.php 的位置为基准  require\_once "includes/functions.php";  require\_once "includes/include.php";  $tpl = new Smarty();  $tpl->template\_dir = \_\_SITE\_ROOT . "/templates/";  $tpl->compile\_dir = \_\_SITE\_ROOT . "/templates\_c/";  $tpl->config\_dir = \_\_SITE\_ROOT . "/configs/";  $tpl->cache\_dir = \_\_SITE\_ROOT . "/cache/";  $tpl->left\_delimiter = '<{';  $tpl->right\_delimiter = '}>';  ?> |

　　modules 这个资料夹则是用来放置程序模块的，如此一来便不会把程序丢得到处都是，整体架构一目了然。  
　　  
　　上面我们也提到 main.php ，这是整个程序的主要核心，不论是常数定义、外部程序加载、共享变量建立等，都是在这里开始的。所以之后的模块都只要将这个档案包含进来就可以啦。因此在程序流程规划期间，就必须好好构思 main.php 中应该要放那些东西；当然利用 include 或 require 指令，把每个环节清楚分离是再好不过了。  
　　 　  
　　在上节提到的 Smarty 程序 5 步骤， main.php 就会帮我们先将前 3 个步骤做好，后面的模块程序只要做后面两个步骤就可以了。  
　　  
　　**从变量开始**  
　　  
　　如何使用变量  
　　  
　　从上一章范例中，我们可以清楚地看到我们利用 <{ 及 }> 这两个标示符号将变量包起来。预设的标示符号为 { 及 } ，但为了中文冲码及 [Java](http://java.chinaitlab.com/)script 的关系，因此笔者还是模仿 XOOPS ，将标示符号换掉。变量的命名方式和 PHP 的变量命名方式是一模一样的，前面也有个 $ 字号 (这和一般的模版引擎不同)。标示符号就有点像是 PHP 中的 <?php 及 ?> (事实上它们的确会被替换成这个) ，所以以下的模版变量写法都是可行的：  
　　  
　　1. <{$var}>  
　　  
　　2. <{ $var }> <!– 和变量之间有空格 –>  
　　  
　　3. <{$var  
　　  
　　}> <!– 启始的标示符号和结束的标示符号不在同一行 –>  
　　在 Smarty 里，变量预设是全域的，也就是说你只要指定一次就好了。指定两次以上的话，变量内容会以最后指定的为主。就算我们在主模版中加载了外部的子模版，子模版中同样的变量一样也会被替代，这样我们就不用再针对子模版再做一次解析的动作。  
　　  
　　而在 PHP 程序中，我们用 Smarty 的 assign 来将变量置放到模版中。 assign 的用法官方手册中已经写得很多了，用法就如同上一节的范例所示。不过在重复区块时，我们就必须将变量做一些手脚后，才能将变量 assign 到模版中，这在下一章再提。  
　　  
　　**修饰你的变量**  
　　  
　　上面我们提到 Smarty 变量呈现的风貌是由模版自行决定的，所以 Smarty 提供了许多修饰变量的函式。使用的方法如下：  
　　  
　　<{变量|修饰函式}> <!– 当修饰函式没有参数时 –>  
　　  
　　<{变量|修饰函式:”参数(非必要，视函式而定)”}> <!– 当修饰函式有参数时 –>  
　　范例如下：  
　　  
　　<{$var|nl2br}> <!– 将变量中的换行字符换成 <br /> –>  
　　  
　　<{$var|string\_format:”%02d”}> <!– 将变量格式化 –>  
　　好，那为什么要让模版自行决定变量呈现的风貌？先看看底下的 HTML ，这是某个购物车结帐的部份画面。  
　　  
　　<input name=”total” type=”hidden” value=”21000″ />  
　　  
　　总金额：21,000 元  
　　一般模版引擎的模版可能会这样写：  
　　  
　　<input name=”total” type=”hidden” value=”{total}” />  
　　  
　　总金额：{format\_total} 元  
　　它们的 PHP 程序中要这样写：

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <?php  $total = 21000;  $tpl->assign("total", $total);  $tpl->assign("format\_total", number\_format($total));  ?> |

　　而 Smarty 的模版就可以这样写： (number\_format 修饰函式请到Smarty 官方网页[下载](http://download.chinaitlab.com/))  
　　  
　　<input name=”total” type=”hidden” value=”<{$total}>” />  
　　  
　　总金额：<{$total|number\_format:””}> 元  
　　Smarty 的 PHP 程序中只要这样写：

|  |  |
| --- | --- |
| 1  2  3  4  5 | <?php  $total = 21000;  $tpl->assign("total", $total);  ?> |

　　所以在 Smarty 中我们只要指定一次变量，剩下的交给模版自行决定即可。这样了解了吗？这就是让模版自行决定变量呈现风貌的好处！  
　　  
　　**控制模版的内容  
　　  
　　重复的区块**  
　　  
　　在 Smarty 样板中，我们要重复一个区块有两种方式： foreach 及 section 。而在程序中我们则要 assign 一个数组，这个数组中可以包含数组数组。就像下面这个例子：  
　　  
　　首先我们来看 PHP 程序是如何写的：  
　　  
　　test2.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | <?php  require "main.php";  $array1 = array(1 => "苹果", 2 => "菠萝", 3 => "香蕉", 4 => "芭乐");  $tpl->assign("array1", $array1);  $array2 = array(  array("index1" => "data1-1", "index2" => "data1-2", "index3" => "data1-3"),  array("index1" => "data2-1", "index2" => "data2-2", "index3" => "data2-3"),  array("index1" => "data3-1", "index2" => "data3-2", "index3" => "data3-3"),  array("index1" => "data4-1", "index2" => "data4-2", "index3" => "data4-3"),  array("index1" => "data5-1", "index2" => "data5-2", "index3" => "data5-3"));  $tpl->assign("array2", $array2);  $tpl->display("test2.htm");  ?> |

　　而模版的写法如下：  
　　  
　　templates/test2.htm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | <html>  　　<head>  　　<meta http-equiv="Content-Type" content="text/html; charset=big5">  　　<title>测试重复区块</title>  　　</head>  　　<body>    <pre>  　　利用 foreach 来呈现 array1  　　<{foreach item=item1 from=$array1}>  　　<{$item1}>  　　<{/foreach}>  　　利用 section 来呈现 array1  　　<{section name=sec1 loop=$array1}>  　　<{$array1[sec1]}>  　　<{/section}>  　　利用 foreach 来呈现 array2  　　<{foreach item=index2 from=$array2}>  　　<{foreach key=key2 item=item2 from=$index2}>  　　<{$key2}>: <{$item2}>  　　<{/foreach}>  　　<{/foreach}>  　　利用 section 来呈现 array1  　　<{section name=sec2 loop=$array2}>  　　index1: <{$array2[sec2].index1}>  　　index2: <{$array2[sec2].index2}>  　　index3: <{$array2[sec2].index3}>  　　<{/section}>  　　</pre>  　　</body>  　　</html> |

　　执行上例后，我们发现不管是 foreach 或 section 两个执行结果是一样的。那么两者到底有何不同呢？  
　　  
　　第一个差别很明显，就是foreach 要以巢状处理的方式来呈现我们所 assign 的两层数组变量，而 section 则以「主数组[循环名称].子数组索引」即可将整个数组呈现出来。由此可知， Smarty 在模版中的 foreach 和 PHP 中的 foreach 是一样的；而 section 则是 Smarty 为了处理如上列的数组变量所发展出来的叙述。当然 section 的功能还不只如此，除了下一节所谈到的巢状资料呈现外，官方手册中也提供了好几个 section 的应用范例。  
　　  
　　不过要注意的是，丢给 section 的数组索引必须是从 0 开始的正整数，即 0, 1, 2, 3, …。如果您的数组索引不是从 0 开始的正整数，那么就得改用 foreach 来呈现您的资料。您可以参考官方讨论区中的此篇讨论，其中探讨了 section 和 foreach 的用法。  
　　  
　　**巢状资料的呈现**  
　　  
　　模版引擎里最令人伤脑筋的大概就是巢状资料的呈现吧，许多著名的模版引擎都会特意强调这点，不过这对 Smarty 来说却是小儿科。  
　　  
　　最常见到的巢状资料，就算论譠程序中的讨论主题区吧。假设要呈现的结果如下：  
　　  
　　公告区  
　　  
　　站务公告  
　　  
　　文学专区  
　　  
　　好书介绍  
　　  
　　奇文共赏  
　　  
　　计算机专区  
　　  
　　硬件外围  
　　  
　　软件讨论  
　　  
　　程序中我们先以静态资料为例：  
　　  
　　test3.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | <?php  require "main.php";  $forum = array(  array("category\_id" => 1, "category\_name" => "公告区",  "topic" => array(  array("topic\_id" => 1, "topic\_name" => "站务公告")  )  ),  array("category\_id" => 2, "category\_name" => "文学专区",  "topic" => array(  array("topic\_id" => 2, "topic\_name" => "好书介绍"),  array("topic\_id" => 3, "topic\_name" => "奇文共赏")  )  ),  array("category\_id" => 3, "category\_name" => "计算机专区",  "topic" => array(  array("topic\_id" => 4, "topic\_name" => "硬件外围"),  array("topic\_id" => 5, "topic\_name" => "软件讨论")  )  )  );  $tpl->assign("forum", $forum);  $tpl->display("test3.htm");  ?> |

　　模版的写法如下：  
　　  
　　templates/test3.htm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | <html>  　　<head>  　　<title>巢状循环测试</title>  　　</head>  　　<body>    <table width="200" border="0" align="center" cellpadding="3" cellspacing="0">  　　<{section name=sec1 loop=$forum}>    <tr>    <td colspan="2"><{$forum[sec1].category\_name}></td>  　　</tr>  　　<{section name=sec2 loop=$forum[sec1].topic}>    <tr>    <td width="25"></td>    <td width="164"><{$forum[sec1].topic[sec2].topic\_name}></td>  　　</tr>  　　<{/section}>  　　<{/section}>  　　</table>  　　</body>  　　</html> |

　　执行的结果就像笔者举的例子一样。  
　　  
　　因此呢，在程序中我们只要想办法把所要重复值一层一层的塞到数组中，再利用 <{第一层数组[循环1].第二层数组[循环2].第三层数组[循环3]. … .数组索引}> 这样的方式来显示每一个巢状循环中的值。至于用什么方法呢？下一节使用数据库时我们再提。  
　　  
　　**转换数据库中的资料**  
　　  
　　上面提到如何显示巢状循环，而实际上应用时我们的资料可能是从数据库中抓取出来的，所以我们就得想办法把数据库的资料变成上述的多重数组的形式。这里笔者用一个 DB 类别来抓取数据库中的资料，您可以自行用您喜欢的方法。  
　　  
　　我们只修改 PHP 程序，模版还是上面那个 (这就是模版引擎的好处~)，其中 $db 这个对象假设已经在 main.php 中建立好了，而且抓出来的资料就是上面的例子。  
　　  
　　test3.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | <?php  require "main.php";  // 先建立第一层数组  $category = array();  $db->set<span class="t\_tag">SQL</span>($SQL1, 'CATEGORY');  if (!$db->query('CATEGORY')) die($db->error());  // 抓取第一层循环的资料  while ($item\_category = $db->fetchAssoc('CATEGORY'))  {  // 建立第二层数组  $topic = array();  $db->setSQL(sprintf($SQL2, $item\_category['category\_id']), 'TOPIC');  if (!$db->query('TOPIC')) die($db->error());  // 抓取第二层循环的资料  while ($item\_topic = $db->fetchAssoc('TOPIC'))  {  // 把抓取的数据推入第二层数组中  array\_push($topic, $item\_topic);  }  // 把第二层数组指定为第一层数组所抓取的数据中的一个成员  $item\_category['topic'] = $topic;  // 把第一层数据推入第一层数组中  array\_push($category, $item\_category);  }  $tpl->assign("forum", $category);  $tpl->display("test3.htm");  ?> |

　　在数据库抓取一笔资料后，我们得到的是一个包含该笔数据的数组。透过 while 叙述及 array\_push 函式，我们将数据库中的资料一笔一笔塞到数组里。如果您只用到单层循环，就把第二层循环 (红色的部份) 去掉即可。  
　　  
　　**决定内容是否显示**  
　　  
　　要决定是否显示内容，我们可以使用 if 这个语法来做选择。例如如果使用者已经登入的话，我们的模版就可以这样写：  
　　  
　　<{if $is\_login == true}>  
　　显示使用者操作选单  
　　<{else}>  
　　显示输入帐号和密码的窗体  
　　<{/if}>  
　　  
　　要注意的是，「==」号两边一定要各留至少一个空格符，否则 Smarty 会无法解析。  
　　  
　　if 语法一般的应用可以参照官方使用说明，所以笔者在这里就不详加介绍了。不过笔者发现了一个有趣的应用：常常会看到程序里要产生这样的一个表格： (数字代表的是资料集的顺序)  
　　  
　　1 2  
　　  
　　3 4  
　　  
　　5 6  
　　  
　　7 8  
　　  
　　这个笔者称之为「横向重复表格」。它的特色和传统的纵向重复不同，前几节我们看到的重复表格都是从上而下，一列只有一笔资料。而横向重复表格则可以横向地在一列中产生 n 笔资料后，再换下一列，直到整个循环结束。要达到这样的功能，最简单的方式只需要 section 和 if 搭配即可。  
　　  
　　我们来看看下面这个例子：  
　　  
　　test4.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | <?php  require "main.php";  $my\_array = array(  array("value" => "0"),  array("value" => "1"),  array("value" => "2"),  array("value" => "3"),  array("value" => "4"),  array("value" => "5"),  array("value" => "6"),  array("value" => "7"),  array("value" => "8"),  array("value" => "9"));  $tpl->assign("my\_array", $my\_array);  $tpl->display('test4.htm');  ?> |

　　模版的写法如下：  
　　  
　　templates/test4.htm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <html>  　　<head>  　　<title>横向重复表格测试</title>  　　</head>  　　<body>    <table width="500" border="1" cellspacing="0" cellpadding="3">    <tr>  　　<{section name=sec1 loop=$my\_array}>    <td><{$my\_array[sec1].value}></td>  　　<{if $smarty.section.sec1.rownum is div by 2}>  　　</tr>    <tr>  　　<{/if}>  　　<{/section}>  　　</tr>  　　</table>  　　</body>  　　</html> |

　　重点在于 $smarty.section.sec1.rownum 这个 Smarty 变量，在 section 循环中这个变量会取得从 1 开始的索引值，所以当 rownum 能被 2 除尽时，就输出 </tr><tr> 使表格换列 (注意！是 </tr> 在前面<tr> 在后面) 。因此数字 2 就是我们在一列中想要呈现的资料笔数。各位可以由此去变化其它不同的呈现方式。  
　　  
　　**加载外部内容**  
　　  
　　我们可以在模版内加载 PHP 程序代码或是另一个子模版，分别是使用 include\_php 及 include 这两个 Smarty 模版语法； include\_php 笔者较少用，使用方式可以查询官方手册，这里不再叙述。  
　　  
　　在使用 include 时，我们可以预先加载子模版，或是动态加载子模版。预先加载通常使用在有共同的文件标头及版权宣告；而动态加载则可以用在统一的框架页，而进一步达到如 Winamp 般可换 Skin 。当然这两种我们也可以混用，视状况而定。  
　　  
　　我们来看看下面这个例子：  
　　  
　　test5.php:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <?php  require "main.php";  $tpl->assign("title", "Include 测试");  $tpl->assign("content", "这是模版 2 中的变量");  $tpl->assign("dyn\_page", "test5\_3.htm");  $tpl->display('test5\_1.htm');  ?> |

　　模版 1 的写法如下：  
　　  
　　templates/test5\_1.htm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=big5">  <title><{$title}></title>  </head>  <body>  <{include file="test5\_2.htm"}>  <{include file=$dyn\_page}>  <{include file="test5\_4.htm" custom\_var="自订变量的内容"}>  </body>  </html> |

　　模版 2 的写法如下：  
　　  
　　templates/test5\_2.htm:  
　　  
　　<{$content}>  
　　模版 3 的写法如下：  
　　  
　　templates/test5\_3.htm:  
　　  
　　这是模版 3 的内容  
　　模版 4 的写法如下：  
　　  
　　templates/test5\_4.htm:  
　　  
　　<{$custom\_var}>

　　这里注意几个重点：1. 模版的位置都是以先前定义的 template\_dir 为基准；2. 所有 include 进来的子模版中，其变量也会被解译。；3. include 中可以用「变量名称=变量内容」来指定引含进来的模版中所包含的变量，如同上面模版 4 的做法。