functions.php文件是一个所有主题都需要配备的文件，很多时候我们可以在里面添加各种函数以实现我们对WordPress的不同需求。不过大多数主题中的functions.php文件的编码都不尽如人意，有待改善。

这些functions.php文件的主要问题是：

* 代码在文件中随意分布
* 不使用WordPress提供的hooks

通过下面这篇文章，主题开发者可以学习如何创建一个代码简洁优美、富有逻辑性的functions.php文件，普通WordPress用户**则可以了解一下functions.php文件的大致结构**。

什么是功能文件？

在WordPress里，所有主题都会有一个叫做functions.php的文件，这个文件负责为主题调用各种PHP函数。functions.php与插件功能类似。此外，每当某个主题被激活，WordPress都会自动加载functions.php文件。使用父/子主题时设置时，则会同时加载父/子主题de functions.php文件。

只要有足够的想象力和PHP技巧，我们可以通过functions.php文件实现自己需要的所有WordPress效果。

*虽然功能文件的工作机制类似于插件，但它并不是插件。*

创建主题功能文件

首先在主题里新建一个空白的functions.php文件。所有的自定义代码都会装入这个文件。新建文件后，在文件里添加下面的代码：

<?php

/\* 自定义代码 \*/

/\* 自定义代码\*/

?>

给函数名称添加前缀

在进行下一步前有一件事需要我们注意：给所有函数添加前缀。

这是一个简单而实际的规则，虽然有很多主题制作者并没有这样的习惯。假设你有一个自定义主题函数comment\_output()，它的常见形式通常是：

function comment\_output() {

/\* 自定义代码 \*/

}

这段代码存在不安全因素。如果某天WordPress或者某个插件添加了一个同样名为comment\_output()的函数，而它们没有为自己使用的函数添加前缀的习惯，那么我们就会遇到致命错误。

假设你的主题名是Browncoats，你可以在所有自定义函数前加上bowncoats\_。那么上面那段代码应该是：

function browncoats\_comment\_output() {

/\* 自定义代码 \*/

}

这是一个简单的变化，也是预防将来出现问题的有效措施。

*这个规则不仅适用于函数名称，对全局变量、常量以及类名称也同样适用。*

新建一个主题设置功能

这里我们想要达到下面几个目标：

* 正确执行某个WordPress hook的功能
* 更好地规划功能文件

WordPress有一个叫做after\_setup\_theme的动作hook。一旦父/子主题的功能文件加载完毕，这个hook也会被执行。after\_setup\_theme是一个非常实用的hook，WordPress默认主题TwentyTen也使用了这个hook。

add\_action( 'after\_setup\_theme', 'browncoats\_theme\_setup' );

function browncoats\_theme\_setup() {

/\* 添加各种过滤器（filter），动作(action)，以及主题支持的功能 \*/

}

这样我们就开始了对functions.php文件的整理。上面这段代码可以用来：

* 为动作hook添加自定义动作
* 为过滤器hook添加自定义过滤器
* 登记对主题支持功能的支持

下面我们来看一个比较规范的functions.php文件：

<?php

add\_action( 'after\_setup\_theme', 'browncoats\_theme\_setup' );

function browncoats\_theme\_setup() {

/\* 添加主题支持的给你功能 \*/

add\_theme\_support( 'automatic-feed-links' );

/\* 添加自定义动作 \*/

add\_action( 'widgets\_init', 'browncoats\_register\_sidebars' );

/\* 添加自定义过滤器 \*/

add\_filter( 'the\_content', 'browncoats\_highlight\_words' );

}

function browncoats\_register\_sidebars() {

/\* 示例 \*/

}

function browncoats\_highlight\_words( $content ) {

return $content;

}

?>

上面的代码其实很简单，涉及三个操作：

* 为after\_setup\_theme动作hook添加主题设置功能browncoats\_theme\_setup()
* 主题设置功能为主题支持的功能登记支持，为一个动作hook添加动作，为过滤器hook添加过滤器
* browncoats\_register\_sidebars()与browncoats\_highlight\_words()在特定hooks上执行

规范功能文件小贴士

**1. 给所有必要内容添加前缀**

总是给自定义的函数添加唯一标识符可以避免很多潜在问题。这篇文章里给函数添加的前缀Browncoats是一个主题的名称，其他主题可以改成对应名称。

**2. 使用hook**

WordPress提供了一种可以使开发者在WordPress工作流程中某些特定点执行操作的hook（钩子）。

functions.php文件中的所有代码都应该被一个自定义函数围绕，而这个自定义函数应该添加在一个动作hook或过滤器hook上。

通过从hook上添加各种内容，功能文件结构会更有条理，用户也可以通过remove\_action()或remove\_filter()函数删除文件里的内容。同时可以保证各个函数在WordPress工作过程中的适当时机执行。

**3. 保持函数简单清爽**

让一个函数执行复杂的操作没有错，可是没有必要只通过一个函数来实现多个功能。如果要执行多次操作，请使用多个函数。

**4. 尽可能使用WordPress内置函数**

编写代码时尽量使用WordPress核心代码函数，好处有二：

* 减少维护和除BUG的代码
* 很多插件依赖于WordPress函数提供的hooks。如果你不使用正确的函数，插件就会出错。（这也是插件作者有时背了主题作者黑锅的原因。）

functions.php文件示例

下面是一个符合WordPress主题检测规范的functions.php文件示范，文件中使用了hooks并编写了主题的一些常用功能。

<?php

add\_action( 'after\_setup\_theme', 'browncoats\_theme\_setup' );

function browncoats\_theme\_setup() {

global $content\_width;

/\* 为嵌入的视频等设置$content\_width \*/

if ( !isset( $content\_width ) )

$content\_width = 600;

/\* 使主题支持自动feed链接 \*/

add\_theme\_support( 'automatic-feed-links' );

/\* 使主题支持文章缩略图（推荐图片） \*/

add\_theme\_support( 'post-thumbnails' );

/\* 使主题支持自定义背景 \*/

add\_custom\_background();

/\* 向 'init' 动作hook添加导航菜单功能 \*/

add\_action( 'init', 'browncoats\_register\_menus' );

/\* 向 'widgets\_init' 动作hook添加侧栏功能 \*/

add\_action( 'widgets\_init', 'browncoats\_register\_sidebars' );

/\* 在 'template\_rediret' 动作hook上加载JS文件 \*/

add\_action( 'template\_redirect', 'browncoats\_load\_scripts' );

}

function browncoats\_register\_menus() {

/\* 通过register\_nav\_menu()

或register\_nav\_menus() 登记导航菜单 \*/

}

function browncoats\_register\_sidebars() {

/\* 通过register\_sidebar()登记动态侧栏 \*/

}

function browncoats\_load\_scripts() {

/\* 通过wp\_enqueue\_script()存储自定义JAVASCRIPT \*/

/\* 加载评论回复javascript \*/

if ( is\_singular() && get\_option( 'thread\_comments' ) && comments\_open() )

wp\_enqueue\_script( 'comment-reply' );

}

?>