Interested in functions, hooks, classes, or methods? Check out the new WordPress Code Reference!

zh-cn:主题开发

Languages: বাংলা • English • Español • 日本語 • 한국어 • Português do Brasil • Русский • ไทย • 中文(繁體) • (Add your language)

本文介绍如何开发设计你自己的 WordPress 主题。如果你希望了解更多如何安装和应用主题的内容，请参阅应用主题文档。本文的内容不同于应用主题，因为所讨论的是编写代码去构建你自己的主题的技术内容，而非怎样去激活主题或者是哪里可以获得新主题。

Contents

为什么要使用 WordPress 主题

WordPress 主题由一系列文件和 CSS 样式表构成，构成了一个美丽的 WordPress 网站。每个主题都是不同的， 这样WordPress用户就可以随时更改 WordPress 网站的外观。

你也许想为自己开发 WordPress 主题，或者制作公开发行的的主题。但是除了这个为什么要自己制作主题呢?

- 创建自己独特的 WordPress 主题外观.
- 利用模板， 模板标签， 和 WordPress循环 来产生不同的效果.
- 为了产生不同的效果，比如在category pages页面和搜索结果页面产生个性的效果.
- 为了迅速从两个主题改变你的博客外观，可以充分利用 Theme or style switcher这个插件迅速改变外观.
- 设计 WordPress 主题，这样大家就可以通过网络更好的使用你的作品.

WordPress 主题有很多优点.

- WordPress 主题把CSS样式表和模板文件 从系统中独立出来，所以这样升级博客的时候就不会破坏你的主题样式.
- 允许你自由的定制主题样式.
- 允许你迅速改变主题.
- 你甚至都不需要学习HTML，CSS，PHP等，即可拥有一个美观的主题.

为什么要自己制作主题呢? 这才是问题的关键.

- 这是一个学习 CSS，HTML，和PHP的好机会.
- 这是一个积累你的 CSS，HTML，PHP实践经验的的机会.
- 制作主题的过程中充满创造力.
- 这非常的有趣（大多数情况下）.
- 如果你 设计公共主题， 你会感觉非常好，因为你为 WordPress 社区做出了自己的贡献 (好吧，可以吹牛了～)

主题开发标准

WordPress 主题应该按照如下标准开发:

- 使用结构化的、没有错误的PHP和有效的HTML代码。请看 WordPress代码规范。
- 使用简洁的、有效的CSS。参见 CSS Coding Standards。
- 遵循设计指南。

主题的剖析

WordPress主题目录位于 `wp-content/themes/`。主题的子目录拥有所有样式文件、模板文件、可选的函数文件 (`functions.php`)、JavaScript 文件、图片等。比如说一个叫做 "test" 的主题就会放在 `wp-content/themes/test/`目录里。请避免使用数字名字，这会导致无法在主题列表中正常显示出来。

WordPress每一个发行版都会有一个默认的主题。请认真查看默认的主题，这样可能会对制作你自己的主题有帮助。

WordPress 主题除了图片和JavaScript，经常由三种文件构成。

1. 样式表文件 `style.css`， 控制着页面的外观
2. 函数文件 (`functions.php`)。
3. 模板文件，它控制着从数据库中调出的数据所呈现的外观。

让我们单独看一下。

子主题

最简单的主题可能是子主题，其仅仅包含一个`style.css`文件，也可以加上一些图片。之所以它能工作是因为它是以另一个主题为基础进行工作的。

更多关于子主题的信息，请看子主题或者由op111写的不错的教材。

主题样式表

CSS文件不仅定义了你的主题样式，*style.css* ***必须*** 以注释的形式列出主题的详细信息。***两个不同的主题是不允许拥有相同的表述的***，因为这样会导致主题选择出错.如果你通过拷贝一个你已经制作的主题来制作你新的主题，请确保先更改这些头部注释.

下面是样式表头部注释的例子，被称作样式表头注释。比如主题"Twenty Thirteen":

```
/*
Theme Name: Twenty Thirteen
Theme URI: http://wordpress.org/themes/twentythirteen
Author: the WordPress team
Author URI: http://wordpress.org/
Description: The 2013 theme for WordPress takes us back to the blog, featuring a full range of post
formats, each displayed beautifully in their own unique way. Design details abound, starting with a
vibrant color scheme and matching header images, beautiful typography and icons, and a flexible layout
that looks great on any device, big or small.
Version: 1.0
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Tags: black, brown, orange, tan, white, yellow, light, one-column, two-columns, right-sidebar,
flexible-width, custom-header, custom-menu, editor-style, featured-images, microformats, post-formats,
rtl-language-support, sticky-post, translation-ready
Text Domain: twentythirteen

This theme, like WordPress, is licensed under the GPL.
Use it to make something cool, have fun, and share what you've learned with others.
*/
```

提醒：主题作者的名字建议与该作者在wordpress.org上注册的用户名一致，当然填写作者的真实姓名也没有问题。请自行选择。

留意用来描述主题的标签(Tags)。它们使用户们可以通过标签筛选器找到你的主题。这里有一份完整的 可用标签列表。

这些位于style.css里的文件是必须要写的，这是用来区分安装的主题.

样式表指南

- 当创作你的CSS的时候请参考 CSS 编码标准。
- 尽可能使用有效的 CSS。作为例外，也可以使用一些前缀，遵循CSS3标准。
- 尽量减少使用CSS hacks。明显浏览器支持(如IE)是个例外，如果可能的话，将CSS hack文件区分开来或者使用独立的文件。
- 所有的 HTML 元素应该有样式声明，无论是文章页面还是评论部分。
  - 表格，标题，图片，列表，块引用，等等。

- 强烈建议添加打印友好的字体
  - 你能通过使用`media="print"`属性来包含一个适用于打印的样式表文件，或者在你的主样式表文件中增加一部分专为打印提供的设置。

函数文件

一个主题可以使用一个函数文件，位于主题的子目录，叫做 `functions.php`。这个文件就像一个插件， 如果它位于你正在使用的主题里的话，他在你的主题初始化的时候就会自动加载(后台页面和前台页面都一样加载)。对于这个文件的建议：

- 启用主题功能，例如：侧边栏，菜单，文章缩略图，文章格式，自定义标题栏。
- 定义用于模板文件中的函数。
- 设置一个选项菜单，让网站拥有者可以自定义颜色，样式，和你的主题的其他特性。

默认的WordPress的主题包含一个`functions.php`文件，它定义这些功能很多，所以你可能会把它当做参考.既然`functions.php`基本上可以作为一个插件，所以Function_Reference可以让你更多的了解这个函数，以及你可以怎么利用这些函数.

关于添加函数到 *functions.php* 还是一个单独插件中的提醒： 你有时会发现需要在多个主题中使用相同的函数，在这种情形下，建议开发主题插件 plugin。*functions.php*仅对特定的主题生效，而插件视所有的主题一致。

模板文件

模板 是一些PHP文件，他可以输出HTML代码呈献给浏览器，决定着主题的外观.下面让我们来看一下主题的模板.

WordPress允许为你的网站定义不同的模板.他虽然不是必需的，但是这些不同的模板为你的网站添上一笔. 模板是根据Template Hierarchy的，由一个具体的主题决定.

作为一个主题开发者，你可以自由决定如何定制你的模板.比如说，极端情况下， 你甚至可以仅仅使用一个文件`index.php`作为模板文件，*所有* 页面都会使用这个模板.更多的情况是，使用不同的模板文件产生不同的结果，以达到最大定制.

模板文件列表

这里是被WordPress确认的主题文件列表.当然，你的主题可以包含任何样式表，图像或者文件.记住 下面列出的文件对WordPress有特殊的意义-- 点击模板层次 查看具体情况.

`style.css`
主样式表，这个文件 **必须** 位于你的主题里面，而且必须在头部注释处写清楚你的主题的信息.

`rtl.css`
rtl 样式表。如果网站的阅读方向是自右向左的，他会**自动**被包含进来。你可以使用 the RTLer 插件来生成这个文件.

index.php

主模板.如果你的主题使用自己的模板，index.php 是必须要有的.

comments.php

评论模板.

front-page.php

首页模板，仅用于开启 静态首页 时。

home.php

主页模板，默认的首页。如果你开启了 静态首页 这是展现最新的文章的模板页面。

single.php

单独页面模板。显示单独的一篇文章时被调用。对于这个以及其他的请求模板，如果模板不存在会使用 index.php。

single-{post-type}.php

自定义单独页面模板。例如， single-books.php 展示自定义文章类型为 books的文章. 如果文章类型未被设置则使用index.php。

page.php

页面模板. 独立页面调用。

category.php

分类模板。 分类页面调用。

tag.php

标签模板。标签页面调用。

taxonomy.php

术语模板。请求自定义分类法的术语时使用。

author.php

作者模板。作者页面调用。

date.php

日期/时间模板，按时间查询时使用的模板。

archive.php

存档模板。查询分类，作者或日期时使用的模板。需要注意的是，该模板将会分别被category.php， author.php， date.php 所覆盖（如果存在的话）。

search.php

搜索结果模板，显示搜索结果时使用的模板。

attachment.php

附件模板，查看单个附件时使用的模板。

image.php

图片附件模板。当在wordpress中查看单个图片时将调用此模板，如果不存在此模板，则调用attachment.php 模板。

404.php

404 错误页面 模板。当WordPress无法查找到匹配查询的日志或页面时，使用404.php文件。

按照Template Hierarchy，这些文件在 WordPress 中有特殊的意义，即当对应的 条件标签 返回 true 的时候，他们将在这种情况下代替index.php ，例如，如果当前显示的是单一的一篇博文，那么is_single() 这个函数将返回'true'，并且如果有一个single.php 文件存在于当前主题中，该文件模板就将起作用。

基本模板

在最简单的情况下，一个WordPress主题由两个文件构成：

- `style.css`
- `index.php`

这些文件都位于主题目录. 这`index.php` 模板 是非常灵活的.他可以用来包含所有的引用 header， sidebar， footer， content，categories， archives， search， error， 和其它在WordPress产生的文件.

或者，他也可以模块化，使用单独的文件分担工作.如果你没有提供其它的模板文件，WordPress 会使用默认文件.比如说，如果你没有提供`comments.php` 文件， WordPress会自动使用 `wp-comments.php` 模板文件 Template Hierarchy. (注意: 自3.0起，那些默认的文件已经不能保证都存在或者跟以前的一样。提供你自己的模板文件会更安全。)

典型的模板文件包括：

- `comments.php`
- `footer.php`
- `header.php`
- `sidebar.php`

使用这些模板文件，你可以把这些文件嵌入到`index.php` 中，最后生成的文件里.

- 包含header， 使用`get_header()`.
- 包含sidebar， 使用 `get_sidebar()`.
- 包含 footer， 使用 `get_footer()`.
- 包含 search form， 使用 `get_search_form()`.

include 用法:

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

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

关于更多的如何利用各种模板，如何产生不同的信息， 请阅读 Templates 文档.

自定义页面模板

定义每一个页面模板的文件都放在你的 Themes 文件夹里面。为了创建一个自定义页面你需要首先创建一个文件，假设我们的第一个自定义页面叫做*snarfer.php*。在*snarfer.php*的文件顶部，你必须要这么写：

```
<?php
/*
Template Name: Snarfer
*/
?>
```

以上代码定义*snarfer.php*为*Snarfer*模板，在创建其它页面的时候*Snarfer*可以替换成其它名字。这个模板名字会作为一个链接出现在主题编辑器中，单击它就可以编辑这个文件。

你可是使用任何以*.php*为后缀的有效文件名(查看 reserved Theme filenames 中你不应该使用的文件名；WordPress保留了特定的文件名以供特殊用途)。

紧接着上面五行代码之后的内容取决于你。你随后编写的代码将控制使用 Snarfer 模板的页面会如何显示。查看 Template Tags 了解你可以使用的丰富的 WordPress 模板函数。你会发现直接从其他模板文件(*page.php* 或者 *index.php*)中拷贝代码到 *snarfer.php* 中，然后在文件顶部添加那五行代码会更加方便。如此以来仅需*替换* HTML 和 PHP 代码而不至于从零开始。展示一个例子 below。一旦你创建了页面模板并将之放在你的主题目录中，你便可以在创建和编辑页面时采用它。(**提醒：**当创建或编辑页面时，除非你按如上方法定义了至少一个模板，否则页面模板类型选择菜单不会出现。)

基于查询的模板文件

WordPress 可以根据不同的*查询*类型加载不同的模板。 有两个办法: 根据模板层次命名你的模板文件，或者在循环中使用条件标签(if 语句).

为了应用模板层次，你仅需要提供特定的模板文件，它们会分别自动替代 `index.php`。例如，如果你提供一个 `category.php` 模板文件，当文章分类页面被请求时， `category.php` 会替代 `index.php` 被加载。如果 `category.php` 文件不存在时， `index.php` 被正常加载。

你甚至可以通过给文件定义更加准确的名字从而获得更加细致的模板层次，例如 `category-6.php` —— 当请求 ID 为 6 的文章分类时该模板会优先于 `category.php` 被加载(在WordPress 2.3及以下版本中，当以管理员登陆后，你可以在Manage > Categories找到分类ID。自WordPress 2.5开始ID栏目被移除了。你可以单击'编辑分类'然后查看浏览器的URL地址找到当前分类的ID。它看上去像这样'...categories.php?action=edit&cat_ID=3'，其中3就是分类ID)。想要了解具体的流程，参考 分类模板。

较之于自动加载模板层次中提供的模板文件，如果你的主题需要自己控制想要加载的文件，你可以使用条件标签。条件标签在 The Loop In Action 中检查指定的条件是否为真，然后你可加载特定的模板，或基于条件在屏幕上显示特定内容。

举例，针对一个指定分类的文章采用特定的样式表，代码像下面这样：

```
<?php
if ( is_category( '9' ) ) {
    get_template_part( 'single2' ); // looking for posts in category with ID of '9'
} else {
    get_template_part( 'single1' ); // put this on every other category post
}
?>
```

或者使用查询，看起来像这样:

```
<?php
$post = $wp_query->post;
if ( in_category( '9' ) ) {
    get_template_part( 'single2' );
} else {
    get_template_part( 'single1' );
}
?>
```

无论何种情况，这份示例代码会依据不同的文章分类加载显示不同的模板。然而查询条件是不仅限于分类的，查看条件标签了解所有选择。

定义模板

可以使用WordPress插件系统来定义你的个人模板。这项高级特性可以通过 `template_redirect` 动作钩子实现。想要了解关于创建插件的更多信息，参考插件API.

包含模板文件

为了加载其他模板(除了 header, sidebar, footer 这些已经被预先定义了加载命令的例如 `get_header()`)到某个模板中，你可以使用 `get_template_part()`。这利于主题的代码重用。

引用模板中的文件

当在同一个主题中引用其他文件时，避免使用硬编码的 URLs 和文件路径。请使用 `bloginfo()` 引用 URL 和文件路径：参看从模板中引用文件.

注意样式表中使用的 URLs 是相对于样式表本身的，而不是相对于引用这个样式表的文件。例如，如果你在主题中包含 `images/` 目录，你只需要在CSS中指明相对路径，像这样：

```
h1 {
    background-image: url(images/my-background.jpg);
}
```

插件 API 钩

开发主题的时候，需要注意的是你的主题最好能和用户可能安装的任何插件共存。插件可以通过“动作钩子（Action Hooks, 查看 插件API）”为wordpress增加功能。

大部分Action Hooks存在于wp的php核心中，所以你的主题不需要任何多余的特殊标签来让它可以正常运转。但是少数的Action Hooks需要在你的主题中做特殊处理，以使插件能够将头，尾，侧边栏等信息输出到页面中。如下是你需要包含到主题 中的Action Hooks列表：

wp_head()

放在`<head>`标签之内，在`header.php`文件中. 大部分插件常在这里导入javascript文件。

wp_footer()

在`footer.php`中，在`</body>`标签之前 . 使用举例：一些插件会在这里插入要在文档最后执行的PHP代码。更常见的用法是插入网页静态代码，比如Google Analytics。

wp_meta()

在主题菜单或侧边栏的`<li>Meta</li>`节之中；`sidebar.php`模板. 插件使用范例: 包含一个旋转广告或标签云。

comment_form()

在`comments.php`中，在表单闭标签出现之前 (`</form>`). 插件使用范例: 显示评论预览。在WordPress 3.0中，你需要使用默认的评论表单，查看comment_form()了解更多信息.

想看看真实应用的案例？这些插件钩子就包含在默认主题的模板中。

不可靠的数据

你应该在主题中避免使用动态生成的内容，尤其是输出到 HTML 标签属性中的内容。依照WordPress 编程规范，属性中出现的文本需要经过 `esc_attr`处理，这样单引号和双引号才不会影响属性值结束，否则会导致无效代码和安全问题。一般要检查的地方有 `title`， `alt`， 和 `value` 属性.

在少数情况下我们需要安全的输出，一个类似的例子便是当使用 `the_title()` 显示文章标题时需要用到 "title" 属性。为了避免安全问题，请使用 `the_title_attribute()`。 这里是一个正确转码的 `title` 属性的例子：

```
<a href="<?php the_permalink(); ?>" title="<?php sprintf( __( 'Permanent Link to %s', 'theme-name' ), the_title_attribute( 'echo=0' ) ); ?>"><?php the_title(); ?></a>
```

使用推荐的转码方式代替弃用的方式：

- `esc_html()` 代替 `wp_specialchars()` 和 `htmlspecialchars()`
- `esc_url()` 代替 `clean_url()`
- `esc_attr()` 代替 `attribute_escape()`

Home Page

WordPress Lessons

Getting Started

Working with WordPress

Design and Layout

Advanced Topics

Troubleshooting

Developer Docs

About WordPress

Codex Resources

Community portal

Current events

Recent changes

Random page

Help