

Gkwp 开发手册（正式版）

Gkwp 是首个 wordpress 中文主题开发框架,由“三吉工社”开发。它将使我们开发 wordpress 主题更加快速、简单。建议大家再阅读本手册之前,看看我们的视频介绍:

http://v.youku.com/v_show/id_XNDY0OTgwMzg0.html

一, 文件结构

首先我们要将下载的主题解压后,将 Gkwp 这个目录放到“您的 wordpress 程序目录/wp-content/themes/”目录下。Gkwp 的目录结构为:

- |---child-theme/ 子主题文件夹
- |---config/ 配置文件夹
- |---languages/ 语言包
- |---lib/ 框架核心文件
- |---options/ 主题选项文件夹
- |---static/ 存放静态文件
- |---widgets/ 小工具
- |---index.php 首页模版文件
- |---functions.php 函数文件
- |---style.css 样式文件
- |---header-meta.php 头部<head>标签内容
- |---header.php 头部模版
- |---content.php 文章内容模版
- |---comments.php 评论模版

|---comment-list.php 评论列表（也就是评论循环部分）

|---searchform.php 搜索框模版

|---sidebar.php 侧边栏模版

|---footer.php 底部模版

二，建立子主题。

请不要通过修改 Gkwp 的文件来制作主题，我们建议你以子主题的方式来制作主题。建立子主题十分简单。首先将 Gkwp 目录下的 child-theme 复制到上级目录。也就是复制到 wp-content/themes 目录。我们可以更改 child-theme 的文件夹名称。然后我们进入 wordpress 的后台，可以看见 “gkwp-子主题” 这个主题，我们启用这个主题。此时在浏览前台，就可以看见简单的样式了。这个界面是用 bootstrap 简单搭建的。

子主题文件夹下的文件其实很少。连 index.php 都没有。wordpress 是先找子主题下面的文件。如果子主题下面找不到文件则使用父主题的文件（这里的父主题是指 Gkwp）。所以在子主题没有 index.php 文件的时候，wordpress 实际是用的 Gkwp 的 index.php 文件，而如果你在子主题下建立 index.php，他们 wordpress 就用你子主题下面的文件了，不会再加载 Gkwp 下的 index.php。我们如果需要修改 Gkwp 的文件，请不要直接修改，而是复制到子主题中进行修改。

更多子主题相关的知识大家可以参考 wordpress 官方文档：

http://codex.wordpress.org/Child_Themes

三，模板开发。

Gkwp 默认开启了调试，当我们以管理员身份登陆后，访问前台在网站的底部会显示调试信息，这调试信息对我们很有用处。

调试信息

当前使用模板：D:\wordpress/wp-content/themes/test/index.php
可建立模版：front-page.php,home.php,index.php

通过当前使用模版，我们能很快的找到需要修改的文件。可建立模版又是什么呢？我们访问首页时，发现可建立模板有：front-page.php, home.php, index.php，wordpress 是这样来加载模版的：如果在主题目录下存在 front-page.php，则会加载 front-page.php，如果 front-page.php 不存在，再判断是否存在 home.php，如果存在 home.php 则加载 home.php，最后，如果 front-page.php 和 home.php 都不存在时，则会加载 index.php。可建立模版从左往右显示的是加载模版的优先级。先找到谁就加载谁。

再比如我们点击菜单栏的“示例页面”，这时候底部显示的可建立模版为 page-2.php，page.php, index.php。我们如果要对“示例页面”这个单页建立一个独立模版，那么只建立 page-2.php 这个模版就行。这里数字 2，是单页的 id。我们只有访问“示例页面”才会加载独立的模版，访问其他单页还是使用原来的模版。

在 wordpress 中，index.php 是一个特殊的模版，通过上面两个例子可以发现，不管什么时候，当 wordpress 找不到对应页面的模版时最后都会加载 index.php。index.php 是一个通用模版，index.php 并不只代表是首页，所以，我们要修改首页，不要在 index.php 文件中修改。而是建立 front-page.php 或者 home.php。我们要创建单页模版，也不要直接在 index.php 中修改，可以创建 page.php 或者 page-{id}.php。

Gkwp 的调试信息，给我们显示出了可以建立哪个模版。是不是很方便？

现在大家可以打开 Gkwp 的模版文件，我先对每个模版中使用的函数做简单的说明。

index.php（通用模版）：

get_header('meta'): 加载 header-meta.php 模版，这个模版中定义了<head>标签中的内容。这个模版我们不用修改，<head>标签中的内容可以通过设置配置来控制，具体怎么设置配置，在第四部分讲解。

get_header(); 加载 header.php。header.php 是头部样式模版，如果你要修改头部的样式，将 Gkwp 中的 header.php 文件复制到子主题中进行修改即可。

have_posts(): 判断是否有文章。

the_post(): 获取当前文章，并指针下移，在 wordpress 中，文章的读取已经封装了一系列函数，一般我们循环读取文章的代码类似：

```
<?php while ( have_posts() ) : the_post(); ?>
```

```
//...读取文章的内容
```

```
<?php endwhile; ?>
```

通过代码我们更加容易理解 the_post 有指针下移的功能。

get_template_part() : 加载模版，get_template_part('content') 会加载 content.php，get_template_part('content','page') 会加载 content-page.php。

get_post_format() : 获得当前主题的形式，要支持主题形式，首先需要设置配置项 theme_support 支持 post-formats。然后在后台发布文章的时候就会有形式的选择。



当选择是 标准是 `get_post_format()` 返回是空 ,
`get_template_part('content',get_post_format())` 就会加载 `content.php`
当选择是日志是 , `get_post_format()`返回是 `aside`
`get_template_part('content',get_post_format())` 就会加载 `content-aside.php`
这样使不同类型的文章可以使用不同的模版。 关于 `theme_support` 配置项如何设置
会在后面第四部分讲解。

`gk_page()` : 分页。 这个分页样式是经过优化的 , 不是 wordpress 默认的分页。

`get_sidebar()` : 加载边栏模版(`sidebar.php`)

`get_footer()` : 加载底部模版(`footer.php`)

header.php (头部样式模版) :

`bloginfo()`: 获得博客的一些信息 , 如网站名 `bloginfo('name')` , 描述
`bloginfo('description')`等.

`wp_nav_menu()`: 输出菜单

`get_search_form()`: 输出搜索框。

`header_image()` : 输出头部图片的地址

content.php(文章内容部分的模版):

the_ID(): 会输出当前文章的 ID 。

post_class(): 会根据文章的属性（如分类，置顶等）输出不一样的 classname。可以方便我们对不同分类文章设置不同样式。

comments_open(); 判断当前主题是否允许评论。

post_password_required() :当前主题是否需要输入密码才能访问。在发布文章的时候可以设置文章需要密码才能访问，post_password_required()是判断当前文章是否需要密码。



公开度： 公开

☐ 公开

☒ 密码保护

密码：

☐ 私密

comments_popup_link(): 显示文件的评论数，点击评论数可以跳转到评论的地方。

the_permalink(): 访问当前文章的内容页的链接地址。

the_title(): 输出文章标题。

get_post_type(): 获得当前文章类型，类型可能有 post，page 等。post 是正常发布的文章，page 是一个单页。

get_the_category_list(): 输出当前文章的分类

get_the_tag_list(): 输出当前文章的标签。

the_content() :输出文章内容。如果是在列表页，会显示<!--more--> 标签之前的内容，如果是在文章内容页，会输出全部文章内容。在编辑文章时，编辑器的下图按钮可以添加<!--more-->标签



the_excerpt() : 显示文章的摘要，如果你没有设置文章摘要，会去掉文章内容中的
html 标签后显示前面 55 个字。

edit_post_link(): 编辑文章的连接

comments.php (评论部分的模版):

get_comments_number(): 评论条数。

wp_list_comments(): 输出评论循环部分。

previous_comments_link(): 上一页

next_comments_link(): 下一页

comment_form(): 显示提交的 form 表单

comment-list.php(评论循环部分的模版)

comment_author_link(): 显示评论作者。

edit_comment_link():评论编辑链接。

comment_class(): 会根据评论不同的属性，输出不同的 classname 方便我们定义不同的样式。

get_avatar(): 获得用户头像。

comment_date(): 评论日期， 单位只精确到天的时间。

comment_time(): 评论时间， 单位精确到秒的时间，不包括年月日。

sidebar.php (侧边栏模版)

dynamic_sidebar(\$name): 显示侧边栏的小工具，\$name 参数是定义的侧边的标识符。Gkwp 默认只有一个右侧边栏，名称为 sidebar-1，如果要设置这个右侧边栏的内容，在后台“外观-->小工具”处设置。我们可以自己添加很多侧边，会在后面边栏设置中详细讲解。

自定义函数

如果你需要在模版中使用自己定义的一些函数，你先把函数定义在子主题的 common.php 文件中，然后就可以在模版中使用自定义函数了。不建议在子主题中建立 functions.php 文件来定义自定义函数。

路径相关函数

获得和子主题相关的路径的函数有：get_stylesheet_directory() 获得子主题的文件夹路径，get_stylesheet_directory_uri() 获得子主题浏览器访问地址。注意请不要在子主题中使用 get_template_directory() 和 get_template_directory_uri() 因为他们获得的路径是父主题的，并不是子主题的。为了方便大家使用路径，Gkwp 定了常量

__ROOT__，功能同 get_stylesheet_directory()，获得子主题的文件夹路径。常量的前后各是两个下划线。

__URL__ 功能同 get_stylesheet_directory_uri()，获得子主题的浏览器访问地址。

本手册对函数讲得比较粗略。在模版中使用的函数，除了以 gk_ 开头的函数是 Gkwp 新增函数，其余函数均是 wordpress 自带函数，如果大家对这些函数的用法不了解，请参考 wordpress 官方文档 http://codex.wordpress.org/Template_Tags。

四，设置配置。

Gkwp 能使很多功能的开发不用写代码，只需要设置配置即可。配置文件是子主题的 config/config.php。它返回一个数组形式，数组的下标为配置项名称，数组的值为配置项的值。能设置哪些配置项，大家可以参考 Gkwp 的 config/config.php 文件。

需要说明的是，配置文件并不像其他文件一样，并不是子主题存在配置文件就不加载父主题的配置项。它是先加载父主题的配置项，如果子主题有配置文件则再加载子主题的配置项。子主题中设置的配置项可以覆盖父主题的相应配置项。所以大家可以吧父主题的配置项当作是配置项的默认值，你如果不对默认值进行修改则不用在子主题定义相关配置项，如果你要对默认值进行修改，复制相关配置项到子主题的配置文件中进行修改即可。

下面对配置项进行说明：

debug：是否开启调试，开启设置为 true，关闭设置为 false，开启调试后，会在前台的网站底部看见调试信息。这个调试信息只有管理员可见。所以在开启调试后，需要以管理员身份登陆后才能看见调试信息。

sidebar_debug：是否开启对边栏数据的调试。开启调试后，可以时时看见边栏设置的默认值。此配置项将在后面“边栏设置”处详细讲解。

content_width：内容宽度，你用的主题，文章内容都会有宽度的限制。如果用户在文章中上传一张图片，而这张图片的宽度超出了文章内容的宽度就很容易破坏主题的布局。设置这个配置项后，用户上传的图片，不会超出这个宽度，不会破坏主题的布局。

注意这是文章内容的宽度，不是网页内容的宽度。这个宽度不包括侧边栏。

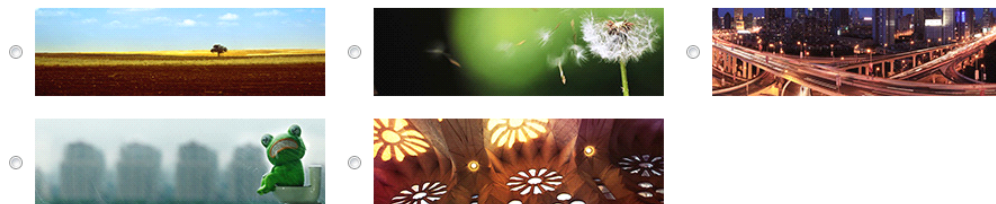
theme_support : 设置主题支持。可设置 post-thumbnails , custom-header , post-formats 等 , 大家可以参考 wordpress 官方文档进行设置 :

http://codex.wordpress.org/Function_Reference/add_theme_support

在 Gkwp 中不用调用 add_theme_support 函数 , 只需要设置 theme_support 配置项就行 , 配置项为一个数组 , 数组的下标为 add_theme_support 的第一个参数。数组的值为 add_theme_support 的第二个参数。大家参考 wordpress 官方文档 , 知道了 add_theme_support 能传什么参数 就知道了 theme_support 配置项如何设置。

register_default_headers : 设置头部图片。当主题支持设置了 custom-header 时 , 后台 外观菜单 下会出现 “顶部” 这个菜单。用户可以定制顶部。而顶部可以选择的图片 , 是通过设置 register_default_headers 这个配置项。

☒ 随机 : 在每个页面上显示不同的图像。



thumbnails : 定义缩略图名称 , 他是一个数组 , 可以定义多个缩略图。数组的格式为 : "名称"=>array("宽","高","是否截取")。如定义 :

```
'thumbnails'=>array(
    'size90'=>array(90,90,true),
    'size120'=>array(120,120,false)
)
```

则在模版中 , 可以用 the_post_thumbnail('size90') 和 the_post_thumbnail('size120'); 显示不同尺寸的缩略图。文章要能设置缩略图必须先设置主题支持特色图片。所以 theme_support 这个配置项中要定义 post-thumbnails。

第三个参数，是否截取，如果 false，不截取 则是等比例缩放。 如果为 true，截取，这样缩略图就是固定设置的宽高了，只是有时候有部分图片会被截取。

wordpress 系统默认已经有了 thumbnail(150px x 150px),medium(300px x 300px),large(640px x 640px),full(原图大小) 这各种缩略图。 我们如果在配置项中定义这些名称会覆盖系统默认的尺寸。

register_nav_menus : 注册菜单。注册的菜单将会在后台 “外观-->菜单” 中的 “主题位置” 中显示。 我们可以对某个菜单设置菜单项。 Gkwp 默认支持了一个菜单，名称为 “主菜单”，它是显示在头部的菜单。如果你的主题需要更多的菜单。如底部也需要菜单。则可以先配置如下

```
'register_nav_menus'=>array(

    'primary'=>__('主菜单','gkwp'),

    'bottom'=>'底部菜单'

)
```

然后再到后台 “外观-->菜单” 处可以对底部菜单进行设置。

设置完菜单后，前台需要显示菜单，调用 wp_nav_menu 函数显示。才函数的 theme_location 参数要为 bottom。wp_nav_menu 的用法详情见 wordpress 官方文档

http://codex.wordpress.org/Function_Reference/wp_nav_menu

languages : 定义自己的语言包，它是一个数组，数组的下标是语言包的名称，数组的值是语言包的文件夹路径。 如定义

```
'languages'=>array(

    'mylang'=>__ROOT__.'/languages'
```

)

`__ROOT__` 是一个常量，表示当前主题（子主题）的主题目录。上面例子表示语言包为当前主题目录下的 `languages` 文件夹。

你可以用一些工具来制作语言包的 `.po` 文件和 `.mo` 文件。推荐大家使用 `poedit`。

定义好语言包后，我们就可以在模版中使用，如：

```
__( 'some word','mylang')
```

register_js：注册 js。Wordpress 为了防止静态文件被重复加载，有 js 和 css 的队列机制。比如 在 wordpress 中使用 jquery，调用函数：`wp_enqueue_script('jquery')` 就可以加载 jquery。可能我们安装的插件很多可能都需要 jquery，他们重复调用很多次 `wp_enqueue_script('jquery')`。但是最终 jquery 只会被加载一次。这就是 js 列队的功能。

首先我们必须得告诉 jquery 的地址在哪儿，`wp_enqueue_script('jquery')` 才会加载到真正的 jquery 文件。这时候就需要注册 js。在不使用框架的情况下，注册 js 是用 wordpress 的函数 `wp_register_script`。而在 Gkwp 中，注册 js 只需要设置配置项 `register_js` 即可。

`register_js` 配置项就是自定义一个名称，和名称对应的 js 文件在哪儿。比如定义了

```
'register'=>array(  
  
    'myjs'=>'static/js/my.js'  
  
)
```

然后我们就可以调用 `wp_enqueue_script('myjs')` 加载 `my.js` 了，同样的，程序如果调用了多次 `wp_enqueue_script('myjs')`，`my.js` 也只会加载一次。wordpress 系统已经有很多注册好了的 js，可以直接用，详情见官方文档：

http://codex.wordpress.org/Function_Reference/wp_register_script

注：js 地址可以是相对地址，也可以是绝对地址，是相对地址表示是相对于主题目录下的地址。如 static/js/my.js' 这个相对地址，是表示主题文件夹下的 static/js/my.js；如果是绝对地址，要以 http://开头。

js：定义主题要加载的 js 文件，可以是已注册 js 的名称，可以是相对地址，可以是绝对地址，可以按条件加载。如定义：

```
'js'=>array(
    'jquery',
    'http://3gk4.com/b.js',
    array('lt IE 9','static/js/html5.js'),
    array(is_singular(),'static/js/singular.js')
);
```

此时 浏览前台，查看前台的 html 源代码，可以看见加载 js 部分的代码如下，

```
<scripttype="text/javascript"
src="http://wp/wp-content/themes/gkwp/static/js/jquery-1.8.2.min.js"></script>
<script type="text/javascript" src="http://3gk4.com/b.js"></script>
<!--[if lt IE 9]>
<scripttype="text/javascript"
src="http://wp/wp-content/themes/gkwp/static/js/html5.js"></script>
<![endif]-->
```

浏览首页并没有加载 singular.js 而如果浏览一个内容页可以看见 singular.js 也被加载了。

下面重点说明一下按条件加载，按条件加载需要定义一个数组，数组的第一个值是条件，第二个值是要加载的文件地址。如果条件是一个字符串。则表示是 html 的注释条件，这种条件代码 ie 能识别，其他浏览器只是会把他们当作注释，如 `array('lte IE 6','static/js/ie6.min.js')`，前台会生成这样的代码：

```
<!--[if lte IE 6]>
```

```
<script type="text/javascript" src="加载文件地址"></script>
```

```
<![endif]-->
```

其中我标记为红色的部分，其实是我们定义的配置项中，数组的第一个值。我们定义什么，红色部分就会显示为什么。

如果数组的第一个值是一个布尔值，则表示为 true 时加载，为 false 时不加载。比如上面举例中 `is_singular()` 这个函数其实会返回一个布尔值。访问内容页是它的返回值才会为 true，访问首页时他的返回值为 false，所以我们在首页时看见没有加载 `singular.js` 而在内容页看见加载了 `singular.js`。同样的我们还可以用 `is_home()`, `is_page()`, `is_category()`, `is_single()` 等函数来进行条件加载，让不同页面加载不同的 js 文件，甚至你还可以定义自己的函数。只要这个函数的返回值是布尔值就行。

register_css：注册 css，这个配置项的作用和 `register_js` 类似，它是防止 css 文件重复加载，我们使用 `wq_enqueue_style('您定义的名称')` 来加载 css，即使调用了多次，也只会加载一次 css。

css：定义主题要加载的 css 文件，此配置项的功能和配置项“js”类似。它同样也可以定义注册 css 的名称，相对地址，绝对地址，按条件加载。用法都和 js 一样，不再重复讲解。

有了 js，css 配置项。我们开发 wordpress 主题，<head> 标签内的内容完全可以由控

制项控制。不用因为不同页面导入的 js , css 不一样 , 而导入不同的头部模版。

自定义配置 : 你还可以自己定义一些配置 , 然后再主题中使用 , 如定义

```
'logo'=>'http://3g4k.com/logo.png'
```

在主题中获得这个配置项用 `gk_config('logo')`。我们也可以在带中对配置项重新设置值如果 `gk_config('logo','http://3g4k/logo2.png');` 然后如果在用 `gk_config('logo')` 获得配置项 , 可以发现它的值已经被更改。

theme_options : 设置主题选项。这个配置项的具体语法在后面主题选项部分讲解。

ext_config : 扩展配置 , 有些时候我们如果觉得某个配置项内容太多 , 想独立为一个单独的文件来定义它 , 就可以用扩展配置。比如前面提到的配置项 `register_default_headers` , 如果你要定义的系统默认图片很多 , 那么你可以将这个配置项定义一个独立文件来配置。在子主题的配置文件中定义 `ext_config` 配置项为

```
'ext_config'=>array('sidebar','widget','register_default_headers')
```

然后可以在子主题的 `config` 目录下 建立一个 `register_default.headers.php` 。 这个文件返回一个数组。

```
<?php
```

```
return array(
```

```
'img_1' => array(
```

```
    'url' => '%s/static/img/headers/006.jpg',
```

```
    'thumbnail_url' => '%s/static/img/headers/006_s.jpg',
```

```
    'description' => __( '淡香沃土', 'gkwp' )
```

```
),
```

```
//.....省略部分代码
```

)

我们如果用 `gk_config('register_default_headers')` 获得配置项的值，其实就是 `register_default_headers.php` 中返回的整个数组。

一般情况下，扩展配置我们用得很少，如果你需要用扩展配置，请注意不要去掉默认的 `sidebar` 和 `widget` 两个扩展配置。

五，边栏设置。

Gkwp 默认只有右侧一个侧边栏，我们实际开发主题中，往往需要很多边栏，比如文章内容页的边栏可能要和首页的边栏不一样，底部可能也要边栏。

1，配置侧边栏：

将 Gkwp 中 `config/sidebar.php` 文件复制到子主题的 `config` 目录下。可以参考 `sidebar.php` 默认的数据，定义我们自己的侧边栏。

如定义：

```
return array(  
  
    array(  
  
        'name' => '网站右侧',  
  
        'id' => 'sidebar-1',  
  
        'before_widget' => '<aside id="%1$s" class="widget %2$s">',  
  
        'after_widget' => "</aside>",  
  
        'before_title' => '<h3 class="right_title">',
```



```

        'after_title' => '</h3>'

    ),

    array(

        'name' => '新增一个侧栏',

        'id' => 'your_name',

        'before_widget' => '<aside id="%1$s" class="widget %2$s">',

        'after_widget' => "</aside>",

        'before_title' => '<h3 class="right_title">',

        'after_title' => '</h3>'

    ),

);

```

此时在到后台 “ 外观-->小工具 ” 处， 可以看见多了个 “ 新增一个侧栏”， 我们就将左侧小工具， 拖动到这个侧边栏中。 在模版显示侧边栏用函数 `dynamic_sidebar('your_name')` 即可。

配置参数说明：

name: 在后台的侧栏的中文显示名称

id : 侧栏的唯一英文标识。

before_widget : 小工具开始的标签。

after_widget : 小工具结束的标签。

before_title : 标题开始的标签。

after_title : 标题结束的标签。

default：此参数可以设置边栏的默认值。

2，设置边栏的默认值。

在边栏配置参数中，有 default 这个参数，这是可以设置边栏的默认小工具的。

这是一个 Gkwp 独有的功能。以前 wordpress 主题开发，我们无法控制真正主题侧栏的默认值。传统设置默认值的方法是，判断读取侧边栏是否成功，如果读取侧边栏失败，则输出一些 html 代码写得假数据：

```
if(!dinamic_sidebar('name')){  
  
    //设置默认值显示样式，一些用 html 代码写得假数据。  
  
}
```

这样用户看着好像是有数据，而在后台“外观-->小工具”处发现对应的边栏却没有数据。用户不知道怎么设置了，这时候你必须得为主题出教程才行。

而现在使用 Gkwp，我们可以真正设置边栏的默认数据了。设置的默认数据让用户再后台小工具设置处也能看见。

default 参数设置此侧栏默认有哪些小工具，填写小工具的英文标识。多个用逗号隔开。默认系统小工具的英文标识有：

recent-comments :最新评论 ,pages :页面 ,links :链接 , search :搜索 ,archives :
归档。meta :功能 ,calendar :日历 ,text :文本 ,categories :分类目录 ,recent-posts :
近期文章 , rss : RSS , tag_cloud : 标签云 , nav_menu : 自定义菜单。

比如我们设置 default 为：

```
default=>'search,recent-comments,recent-posts'
```

表示该边栏默认有 搜索，最新评论，最近文章 这三个小工具。

3, 设置小工具的默认值。

边栏设置完默认小工具，小工具也应该有默认值才行。不然边栏的小工具还是无法显示。

设置小工具的默认值在 config/widget.php 中设置。它返回一个数组，数组的下标为小工具的英文标识，数组的值是设置的小工具的属性。如设置：

```
array(  
  
    'search'=>array('title'=>"),  
  
    'recent-comments'=>array('title'=>'最近评论','number'=>3),  
  
    'recent-posts'=>array('title'=>'最近文章','number'=>5)  
);
```

一个小工具可以设置什么属性？

比如

```
'recent-comments'=>array('title'=>'最近评论','number'=>3),
```

其实就相对于我们在后台设置时填写的标题和显示数量。



近期评论： 最近评论

标题：
最近评论

显示评论的数量： 3

删除 | 关闭 保存

大家可以用 firebug 等工具查看到输入框的 name 属性。比如标题这个输入的 name 属性类似：

```
widget-recent-comments[n][title]
```

我们取 name 属性的最后一个下标。然后就知道了 标题这个输入框的英文

标识是 title, 那么最近评论这个小工具就可以设置 title 属性。

4, 侧栏调试

我们设置的侧栏默认数据, 只要在启用主题的时候才会设置默认数据, 这显然不利于我们的开发。所以 Gkwp 有一个配置项 sidebar_debug, 帮助我们调试侧栏数据, 当设置 sidebar_debug 为 true 时, 每次访问前台都会重新设置侧栏的默认值。这时候你会发现, 在后台设置侧栏是不起作用的, 因为程序每次都会重新设置为你配置的默认值。当你配置好侧栏的默认值后, 记得将 sidebar_debug 设置为 false。

六, 小工具开发。

Gkwp 能让你改变小工具的内部 html 标签结构, 比如最近评论这个小工具。如果你想改变内部 html 标签结构, 复制 Gkwp 父主题下的 widgets/comments.php 到子主题的 widgets 目录下, 如果子主题下还没有建立 widgets 目录, 手动建立即可。然后打开 comments.php, 说明一下小工具文件内的几个变量:

\$before_widget, \$before_title, \$after_title, \$after_widget 这些变量是注册边栏时设置的参数。请看前面讲过的注册边栏时的配置参数, 这些变量的值就是那设置的参数。像这样的变量你还可以自定义。比如 你在注册侧栏的参数设置为:

```
array(  
  
    'name' => '新增一个侧栏',  
  
    'id' => 'your_name',  
  
    'before_widget' => '<aside id="%1$s" class="widget %2$s">',
```

```
'after_widget' => "</aside>",  
  
'before_title' => '<h3 class="right_title">',  
  
'after_title' => '</h3>',  
  
'new_var'=>'my var'  
  
)
```

然后 你在小工具中就可以使用 `$new_var` 这个变量。

\$instance 变量是后台设置的参数。比如最近评论这个小工具可以在后台设置标题和显示评论数：



那么 `$instance['title']` 获得的就是后台设置的标题。`$instance['number']` 获得的就是设置的评论数量。 `title` 和 `number` 其实也是 `input` 输入框的 `name` 属性。所以大家在后台小工具设置的地方，用 `firebug` 等工具查看一下输入框的 `name` 属性，就知道用什么 `$instance` 的下标来获得这里设置的值了。

七，主题选项开发。

wordpress 的主题选项功能是指可以在 后台“外观”菜单下增加子菜单。 这个是一个很有用的功能，比如我们网站的 logo 如果希望用户在后台可以上传。则我们可以在 外观菜单下增加一个“上传 logo”的菜单。

1，设置配置

如何增加菜单呢？需要设置配置。在子主题的配置文件中设置 theme_options 配置项。他是一个数组，数组的下标是菜单的英文标识。数组的值是菜单的中文名称。比如配置如下：

```
'theme_options'=>array(
    'logo'=>'上传 logo'
)
```

这时候我们浏览后台，外观菜单下就会新增“上传 logo”的菜单。

2，建立菜单文件

刚才菜单建立了后，点击菜单后要显示什么内容？这时候我们需要在子主题下的 options 目录新建一个文件，文件名称要为 logo.php，因为刚才我们新增菜单的英文标识为 logo，所以这里建立的文件名称也就要为 logo.php。logo.php 就是为菜单的显示内容，logo.php 应该有哪些代码，可以参考父主题下的 options/test.php 文件。

菜单文件中主要需要有的内容有：

文件中需要有一个 form 表单，form 表单的 action 属性为 "options.php"。

settings_fields(\$option_name); 必须要有这段代码，这是设置一组选项，这段代码会输出一些隐藏域。这些隐藏域就是告诉 options.php 文件，表单提交给它后怎么处理的。

而变量 \$option_name 它的值会为当前菜单的英文标识。

\$options = get_option(\$option_name); 这段代码是获得选项组的值。

你要在 form 表单中建立很多输入框，一个输入框的是用户要设置的选项。比如上传 logo 加上可以设置 logo 的宽带和高度，那么可能有以下输入框

```
<input name="logo[width]" value="<?php echo $option['width']?>" />
```

```
<input name="logo[height]" value="<?php echo $option['height']?>" />
```

输入的 name 属性要是个数组，数组的名称都是菜单的英文标识，数组的下标为你定义的选项名称。

上传 logo，当然也要能上传图片才行。Gkwp 实现上传图片十分简单，只要调用一个函数就行。

上传文件：<?php gk_upload('logo[upload'],\$options['upload']);?>

gk_upload 函数 会输出一个输入框。 他的第一个参数就是输入框的名称， 第二个参数是输入框的值， 第三个参数可选，是输入框的 class 名称（默认为 gk_upload）。 第四个参数可选，是输入框上传按钮的文字（默认为“选择文件”）。

3，模版中获得选项的中。

在模版中用 get_option 可以获得选项的值。 比如获得和 logo 相关的选项的。

```
$logo=get_option('logo');
```

```
$logo['width'];// 设置的宽度
```

```
$logo['height'];//设置的高度
```

```
$logo['upload'];//上传的图片。
```

本手册为试读版，如果你发现什么问题或有其他疑问，请联系三吉工社新浪微博
weibo.com/3g4k

八，关于升级

1.0 版本：发布的第一个版本，

1.1 版本：相比于 1.0 版

- 1, 增加了配置项 register_nav_menus , sidebar_debug。
- 2, 对系统的默认小工具都做了改造。开发者可以在 widgets 目录下找到对应小工具进行修改。
- 3, 导航菜单支持到 3 级菜单 (1.0 版时 , 只支持一级菜单)
- 4, 调整了默认的 UI , Gkwp 的默认 UI 可以当作一个简洁主题使用。