# wordpress主题开发

# 一、创立文件

/wp-content/themes/主题

新建index.php和style.css文件

# 二、设置主题

1、框架会自动识别主题

***2、主题缩略图***

/wp-content/themes/主题/

创建screenshot.png，主题会自动识别

1. ***加载顺序***

*模板-模板加载、别名-设置别名*

首页：front-page.php/page.php/index.php

页面(page)：模板/别名/page.php

文章：

分类：

# 函数备用

## 1、杂项

wp\_trim\_words( $content, 66 ); //截取字符窜

get\_template\_directory() //获取主题路径，不直接输出

require\_once( );       // 引入文件

//按照这种方式所写的调用函数会判断模板文件夹下是否有 diy-one.php 如果没有则调用diy.php。

**get\_template\_part('diy','one')。**

## page属性/get\_page()

$page\_id = 12;

$page\_data = get\_page( $page\_id ); //通过

echo $page\_data->post\_title;// 标题

echo apply\_filters('the\_content', $page\_data->post\_content); //内容

## 执行一条sql / global $wpdb

$name = 'about-us'; //page别名

global $wpdb;

$page\_id = $wpdb->get\_var("SELECT ID FROM $wpdb->posts WHERE post\_name = '$name'");

echo $page\_data = get\_page( $page\_id )->post\_content;

## 4、内容查询

### pre\_get\_posts 动作

Wordpress创建了一个全局变量 $query，通过这个方法修改了指定页面的循环，指定页面，使用have\_post() the\_title

### query\_posts()

<?php// 查询

query\_posts( $args );

while ( have\_posts() ) : the\_post();

the\_title();

endwhile;

wp\_reset\_query(); // 重置查询

?>

### WP\_Query 类 \*\*

$the\_query = new WP\_Query( $args );

// 循环开始

if ( $the\_query->have\_posts() ) :

while ( $the\_query->have\_posts() ) : $the\_query->the\_post();

the\_post()/the\_title()/等等，主循环方法

endwhile;

endif;

wp\_reset\_postdata(); // 重置文章数据

==================

$args = array(

‘author’=>123, //作者id

‘cat’=>1 , //分类id

‘category\_name’=>’111’, //别称

‘category\_\_in’=>’4’,//

posts\_per\_page=> 查询几篇文章

)

# BUG

## 开启debug

/wp-config.php

define( 'WP\_DEBUG', true ); //开启debug

## 后台设置文章居中，前端页面没有效果

添加CSS样式，前端

.aligncenter img{

display: block;

margin-left: auto;

margin-right: auto;

}

.alignright img{

float: right;

margin-left: 10px;

}

.alignleft img{

float: left;

margin-right: 10px;

}

# 五、功能开发

## 1、基本功能

### （1）页面

/wp-content/themes/主题目录/page.php

页面模板默认为page.php ，

1. 创建自己的模板文件page-my.php,以page开头，可以放在主题目录下的任何位置
2. 添加注释，这个注释会解析成**模板名称**

/\*\*

\* Template Name: 成功案例

\*

\*\*/

1. 后台添加新页面，选择对应模板

### （2）文章

#### 首选：使用模板区分

/wp-content/themes/主题目录/single.php

文章会使用single.php做为默认模板

1. 创建自己的文章模板single-my.php
2. 添加注释，可以解析成模板名称

/\*

\* Template Name: 案例模板

\* Template Post Type: post

\*/

1. 后台添加新文章，选择对应为文章模板

#### 方案一：使用标签区分文章

/wp-content/themes/innofit/single.php

文章内容默认模板为content.php

1. 找到single.php中代码

//get\_post\_format()会返回使用的标签

//最终拼接的文件名为content-my.php

get\_template\_part( 'content', get\_post\_format());

2、复制content.php，重命名为content-my.php

3、修改content-my.php内容为想要的内容

4、**指定文章标签为指定标签**

**---------------------**

**通过在single.php文件中调用不同的模板，实现加载不同的模板**

#### 方案二：使用标签区分

/wp-content/themes/主题目录/funcion.php

1. 在文件最后添加

add\_action('template\_include', 'load\_single\_template');

function load\_single\_template($template) {

    $new\_template = '';

    // single post template 5

    if( is\_single() ) {

        //global $post;

        // 'wordpress' is category slugs 8

        // if( has\_term('wordpress', 'category', $post) ) {

        if( has\_post\_format('gallery')) { **//这里添加的是标签名**

            // 这里决定要加载的文件

            $new\_template = locate\_template(array('single-mylist.php' ));

        }

    }

    return ('' != $new\_template) ? $new\_template : $template;

}

1. 创建指定文件的标签

#### 启动文章标签

/wp-content/themes/主题目录/funcion.php

add\_theme\_support( 'post-formats',

array( 'aside', 'chat','gallery','image','link', 'quote', 'status', 'video', 'audio' )

);

显示所有的tab

### （3）分类目录

**使用别名区分分类**

/wp-content/themes/主题目录/category.php

分类目录模板默认为category.php，没有自己创建。

1、创建自己的模板，复制category.php，命名为category-my.php

2、修改category-my.php代码

//这里是加载同级目录下的content-addus.php文件

get\_template\_part( 'content-addus','' )

3、复制content.php，重命名为content-addus.php

4、修改content-addus.php内容为想要的内容

5、创建分类目录mylist(自定义)

6、**指定为目录别名为my(这里是明确使用哪一个分类模板，my名称需要与category-my.php的my一样**

**-------------**

**通过这个方法可以重新定义分类页面，直接重写categort-xxxx.php页面中的代码即可。**

### （4）查找主题加载的模板

/wp-includ/template-loader.php

if ( $template = apply\_filters( 'template\_include', $template ) ) {

    echo $template;exit(); //打印模板路径

    //include( $template );

} elseif ( current\_user\_can( 'switch\_themes' ) ) {

    $theme = wp\_get\_theme();

    if ( $theme->errors() ) {

        wp\_die( $theme->errors() );

    }

}

### 加载主题目录下为文件

//加载 /主题/tmplate/index-news.php

get\_template\_part("template/index-news");

/主题/index.php

get\_template\_part(“index”)

/主题/index-one.php

get\_template\_part(“index”,”one”) //没有加载index,有加载index-one.php

## 2、使用Ajax

### 后台注册

/wp-content/themes/主题目录/function.php

/\*\*

\* 处理请求数据

\*/

function content\_info() {

$post\_id = $\_POST['id'];

// $data = get\_post($post\_id, OBJECT)->post\_content; //根据ID获取文章

// 输出响应

header( "Content-Type: application/json" );

$data = array(

'info'=>'this is my test',

'success' => true,

);

echo json\_encode($data);

exit;//这个停止一定要写

}

//wp\_ajax\_nopriv\_ 是在没有登录的处理流程;wp\_ajax\_是登录后的处理流程  
//add\_action( ‘wp\_ajax\_nopriv\_\*\*\*’, ‘dosome’ );  
//add\_action( ‘wp\_ajax\_\*\*\*’, ‘dosome’ );

//函数名对应添加上，第一个表示用户没有登录时，这里全部都一样处理

add\_action( 'wp\_ajax\_nopriv\_content\_info', 'content\_info' ); //用户没登陆

add\_action( 'wp\_ajax\_content\_info', 'content\_info' ); //用户已登录

### 前台请求

//获取ajax处理地址，统一

$ajaxurl = "<?php echo admin\_url( 'admin-ajax.php' ); ?>"

var data={

"id": $id,

"action" : 'content\_info', //必填，action的参数要和请求的函数名一致

}

//正常的ajax

$.post($ajaxurl, data, function(response) {

console.log(response['info'])

});

## 3、导航栏自定义

### 注册导航栏

这里作用是在后台显示，**外观-菜单** 按钮，对菜单输出没有影响

if(function\_exists('register\_nav\_menus')){

register\_nav\_menus( //注册菜单栏

array(

//这里的配置项是显示在 **外观-菜单-管理位置**

'header-menu' => \_\_( '导航自定义菜单' ), //显示的名称

// 'footer-menu' => \_\_( '页角自定义菜单' ),

// 'sider-menu' => \_\_('侧边栏菜单')

)

);

}

---------------------

***通过后台可修改功能***

1. 注册register\_nav\_menus方法，填写菜单标示
2. wp\_nav\_menu配置

theme\_location与register\_nav\_menus中的配置项关联

menu 配置项为空

3、外观-菜单-管理位置 进行修改菜单

### 在生成导航栏的位置调用方法

<?php

echo wp\_nav\_menu(

array(

'theme\_location' => 'header-menu', //使用注册的菜单，function中注册

'menu' => 'two', //加载不同的菜单，默认为空

'container' => 'div', //ul最外面的div，父类节点 container => false去掉父类节点

'container\_class' => 'con', //导航栏最外面

'container\_id' => '', //导航栏ID

'menu\_class' => 'head-nav', //ul的class

'menu\_id' => '', //ul的id

'echo' => true, //直接显示

'fallback\_cb' => 'wp\_page\_menu',

'before' => '', //在li里输出内容

'after' => '', //在li里输出内容

'link\_before' => '', //在li里输出内容

'link\_after' => '', //在li里输出内容

'items\_wrap' => '<a class="logo"></a><nav class="%2$s">%3$s</nav>',

// 'items\_wrap' => '%3$s', //ul

'depth' => 0, //显示深度

'walker' => new wp\_bootstrap\_navwalker //自定义标题栏

)

);

?>

### 自定义导航栏

walker参数自定义导航栏，详细查看wp\_navwalker.php查看

class wp\_bootstrap\_navwalker extends Walker\_Nav\_Menu {

/\*\*

\* 主要修改的方法

\*/

function start\_el( &$output, $item, $depth = 0, $args = array(), $id = 0 ) {

//$output 是输出的菜单

//$item 是当前菜单的属性

$attributes = ! empty($item->url)? ' href="' . esc\_attr( $item->url) .'"' : ''; //用于配置 a 标签的属性

$active = $item->object\_id == get\_the\_ID()? "active": ""; //给当前页添加active属性

//用于生产导航栏

$item\_output = sprintf( '<a%1$s><div class="navBox box2 %3$s" >%2$s</div></a>',

$attributes, //a标签属性

apply\_filters( 'the\_title', $item->title, $item->ID ), //

$active

);

$output .= apply\_filters( 'walker\_nav\_menu\_start\_el', $item\_output, $item, $depth, $args ); //输出导航栏，复制

}

## 4、后台添加指定菜单

/\*\*

\* 后台侧边栏添加自定义菜单

\* 在后台菜单生成时，挂载一个动作

\* add\_menu\_page菜单注册功能

\*\*/

add\_action('admin\_menu', 'register\_menu');

function register\_menu() {

add\_menu\_page(

'标题', //页面名称

'留言信息', //菜单名称

'edit\_theme\_options', //权限编辑主题选项

\_\_FILE\_\_, //打开文件的路径

'ruikeedu\_com\_toplevel\_page' //调用的方法，一般是返回一个页面

);}

# 六、WordPress默认主题模板文件说明

## 1、文件夹

css（文件夹）：存放 css 文件

genericons（文件夹）：存放字体文件

inc（文件夹）：存放 php 文件，文件内一般是自定义函数

js（文件夹）：存放 js 文件

languages（文件夹）：存放语言文件

template-parts（文件夹）：为了代码简洁，作者把一些代码（可能重复用到的）分开放到了此文件夹下

## 2、文件

404.php：出现404错误时使用的文件

footer.php：页脚

header.php：页头

index.php：首页

archive.php：分类页面

comments.php：评论

page.php：页面（page）

single.php：文章（post）页面

functions.php：定义函数

style.css：样式文件

image.php：图片

rtl.css：关于字体的一些样式

screenshot.php：主题缩略图，在后台显示

search.php：搜索页面

searchform.php：搜索框

sidebar.php：侧边栏

sidebar-content-bottom.php：侧边栏底部

screenshot.png  
从WP2.0开始，模版选择会提供缩略图，这个文件就是缩略图文件。

主题支持函数：functions.php  
自定义头部，进入后台主题设置可以看到，可扩展性相当强。

# css文件

Theme Name. (必需) 子主题的名称。

Theme URI. (可选) 子主题的主页。

Description. (可选) 子主题的描述。比如：我的第一个子主题，真棒！

Author URI. (可选) 作者主页。

Author. (optional) 作者的名字。

Template. (必需) 父主题的目录名，区别大小写。 注意： 当你更改子主题名字时，要先换成别的主题。

Version. (可选) 子主题的版本。比如：0.1，1.0，等。

/\*

Theme Name: Twenty Ten Child

Description: Child theme for the Twenty Ten theme

Author: Your name here

Template: twentyten

\*/

 @import url("../twentyten/style.css");

 #site-title a {

color: #009900;}

# 自定义主题菜单栏

## 注册

add\_action('customize\_register', 'info'); //修改about

function info($wp\_customize) //将主要函数注册

## 添加主题侧边栏

//info与函数名相同

$wp\_customize->add\_section('info', array(

'title' => 'about', //在后台显示的内容

));

## 3、添加控制器

$wp\_customize->add\_control('info\_select', array(

**'type' => 'textarea', //类型**

**'section' => 'info', //与section项相同**

**'label' => "内容",**

'description' => \_\_('说明'),

'input\_attrs' => array( //样式属性

'class' => 'my-custom-class-for-js', // 添加class

'style' => 'height:100px;width:100%;', // 添加style

'placeholder' => ‘’, // 默认输出项

)

));

## 4、添加设置项

//info\_select与control中的选项相同

$wp\_customize->add\_setting('info\_select', array(

'capability' => 'edit\_theme\_options',

'theme\_supports' => '', /

'default' => '',

'transport' => 'refresh', // or postMessage 刷新相关

'sanitize\_callback' => '',

'sanitize\_js\_callback' => '', // Basically to\_json.

));

## 5、各种其他功能控制器

### 图片上传

$wp\_customize->add\_control(

new WP\_Customize\_Image\_Control( $wp\_customize,

'about\_info\_title\_image', //声明setting

array(

'label' => \_\_('标题图片', 'spasalon'), //标题

'section' => 'about\_info\_title', //注册到section

'type' => 'upload',

)));

### 颜色选择