

Table of Contents

本书介绍	1.1
1 导读	1.2
2 部署 WordPress	1.3
3 WP 的环境配置	1.4
WordPress 使用	1.5
1 WP 基本使用	1.5.1
2 WP 基本设置	1.5.2
3 WP 常用插件使用说明（一）	1.5.3
4 WP 常用插件使用说明（二）	1.5.4
5 WP 相关资源站点	1.5.5
6 WP 站点性能优化（一）：动静分离	1.5.6
7 WP 站点性能优化（二）：服务器优化	1.5.7
8 WP 站点性能优化（三）：WordPress 缓存	1.5.8
9 WP 站点性能优化（四）：程序优化	1.5.9
10 WP 安全加固	1.5.10
WordPress 主题	1.6
1 WP 主题使用	1.6.1
2 WP 主题开发：快速入门	1.6.2
3 WP 主题开发：文件结构	1.6.3
4 WP 主题开发：一些特殊的页面	1.6.4
5 WP 主题开发：接入 Options Framework	1.6.5
6 WP 主题开发：一些 WordPress 开发的小技巧	1.6.6
7 WP 主题开发：提交主题到 WordPress 官方仓库	1.6.7
WordPress 插件	1.7
1 WP 插件使用	1.7.1
2 WP 插件的运行机制	1.7.2
3 WP 插件的创建	1.7.3
4 WP 插件后台设计与开发	1.7.4
5 开发一个 WordPress Widget	1.7.5
6 开发一个短代码插件	1.7.6
7 提交你的插件到 WordPress 官方仓库	1.7.7
多语言	1.8
1 WP 主题开发：为你的主题/插件实现国际化	1.8.1
2 使用 WPML 插件建设一个多语言站点	1.8.2
3 使用 Polylang 插件建设一个多语言站点	1.8.3
实战	1.9
1 WordPress 数据库操作 WPDB 指南	1.9.1

2 动手开发插件：Custom Author 插件开发实战	1.9.2
3 轻松玩转 WP：如何使用 WordPress 的邮件发文功能	1.9.3
答疑	1.10
答疑 20171130	1.10.1
答疑 20171201	1.10.2
答疑 20171206	1.10.3
答疑 20171216	1.10.4

人人都能学会的 WordPress 实战课

build passing

电子书介绍

每一个程序员都需要一个博客，记录自己的所见所闻、分享自己的所思所想。与其使用一个受限制的第三方博客服务，不如花点小钱建一个不受限制的 WordPress 博客。

WordPress 作为世界上使用最多的 CMS（内容管理系统），成为大多数新手程序员的选择。实际上，WordPress 并不只是做一个博客的选择，小到博客，社区解决方案，对于 WordPress 来说都不是问题，海量的拓展库给了 WordPress 无限的可能。

作为开发者，除了能够独立开发产品，也应该能够使用现有的产品来简化自己的工作，将精力放在更加重要的部分。

这本电子书将会教你最基础的 WordPress 使用、WordPress 优化、WordPress 主题开发、WordPress 插件开发、为你的插件/主题加入多语言支持、为 WordPress 加入商城功能、以及最终，我们将实践如何在其他应用中接入 WordPress。

整个课程分为多个章节，你可以根据自己的情况，选择合适的章节阅读。

如果你有兴趣，也欢迎你到 GitChat 上去[订阅达人课](#)支持我。

目录

- 1 导读
- 2 部署 WordPress
- 3 WP 的环境配置
- WordPress 使用
 - 1 WP 基本使用
 - 2 WP 基本设置
 - 3 WP 常用插件使用说明（一）
 - 4 WP 常用插件使用说明（二）
 - 5 WP 相关资源站点
 - 6 WP 站点性能优化（一）：动静分离
 - 7 WP 站点性能优化（二）：服务器优化
 - 8 WP 站点性能优化（三）：WordPress 缓存
 - 9 WP 站点性能优化（四）：程序优化
 - 10 WP 安全加固
- WordPress 主题
 - 1 WP 主题使用
 - 2 WP 主题开发：快速入门
 - 3 WP 主题开发：文件结构
 - 4 WP 主题开发：一些特殊的页面
 - 5 WP 主题开发：接入 Options Framework
 - 6 WP 主题开发：一些 WordPress 开发的小技巧
 - 7 WP 主题开发：提交主题到 WordPress 官方仓库
- WordPress 插件
 - 1 WP 插件使用
 - 2 WP 插件的运行机制

- [3 WP 插件的创建](#)
- [4 WP 插件后台设计与开发](#)
- [5 开发一个 WordPress Widget](#)
- [6 开发一个短代码插件](#)
- [7 提交你的插件到 WordPress 官方仓库](#)
- 多语言
 - [1 WP 主题开发：为你的主题/插件实现国际化](#)
 - [2 使用 WPML 插件建设一个多语言站点](#)
 - [3 使用 Polylang 插件建设一个多语言站点](#)
- 实战
 - [1 WordPress 数据库操作 WPDB 指南](#)
 - [2 动手开发插件：Custom Author 插件开发实战](#)
 - [3 轻松玩转 WP：如何使用 WordPress 的邮件发文功能](#)
- 答疑
 - [答疑 20171130](#)
 - [答疑 20171201](#)
 - [答疑 20171206](#)
 - [答疑 20171216](#)

协议



本作品采用[知识共享署名-非商业性使用-禁止演绎 4.0 国际许可协议](#)进行许可。

导读：课前须知

WordPress 是一个独立博客系统，说简单也很简单，著名的“五分钟安装”让不少人心仪；说复杂也复杂，上千万的插件和主题让人眼花缭乱。

我们在创建一个 WordPress 独立博客时，会接触到诸如虚拟主机、VPS、独立服务器、云服务器、域名、SEO 等很多新鲜的词汇。

如果不明白这些词汇，那么要好好学习了。希望在进行后续的课程时，掌握一些小的技能，具体如下。

善用搜索引擎

很多问题都可以通过搜索引擎来找到。该达人课除了能够分享我自己踩坑之后的心得，对读者来说，最大的价值便是减少搜集信息的成本。

但人力有时穷，我无法保证能够覆盖到读者所有的需求，即使遇见了问题来找我问，我也可能无法在最快的时间答复。所以，如果能够在学习前掌握搜索引擎的使用技巧，当遇到问题时，可以优先使用搜索引擎搜索，在无法获得解答时，可以选择到文章下或读者圈中提问，等待我的回复。

如果可以，尽可能的选择 Google 作为主用搜索引擎，至少在搜索技术文章上，Google 要比百度更加好用。如果无法使用 Google，Bing 也是一个不错的选择。

掌握好 `site:` 的用法，这个搜索技巧真的非常有用。

有基础的 HTML、CSS 知识

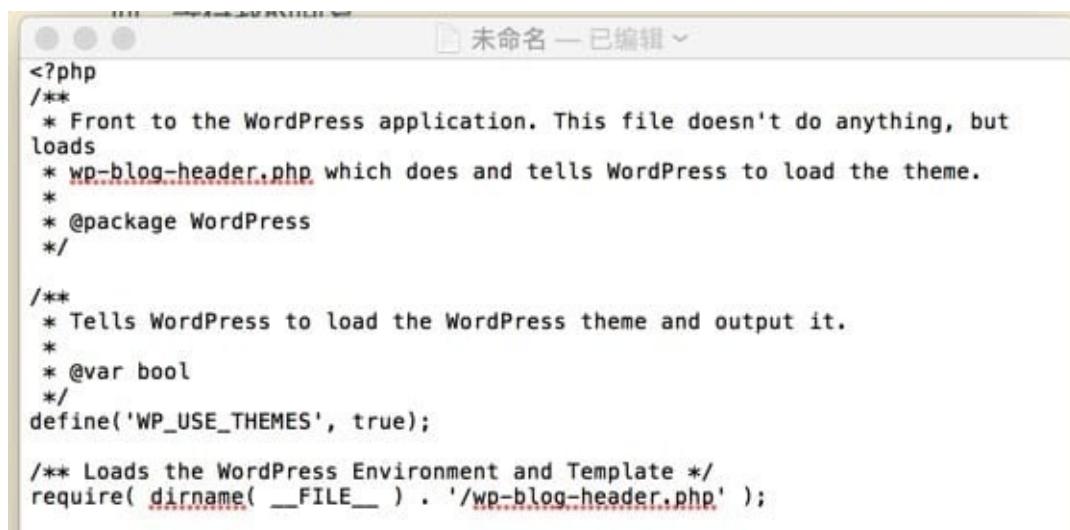
我在课程中设计了 WordPress 主题开发的内容，但是毕竟这门课程主要是在说 WordPress 开发，所以虽然会涉及到 HTML、CSS，但我并不会做具体的讲解，读者可以选择到 [MDN Web 文档](#) 中去学习相关内容。

此外，[菜鸟教程](#)也是一个不错的站点，假如 HTML、CSS 学的不怎么样，可以去看看它的 BootStrap 教程，能够做出一个至少还能看得过去的页面。

熟练使用一款代码编辑器

无论是 Atom 、Sublime Text or Visual Studio Code，要求不高，只要求是一个代码编辑器。

一个代码编辑器可以帮助读者很好的做代码高亮的相关工作，借助代码高亮，能够更好的分辨代码的层级，减少写错代码的可能。



```

未命名 - 已编辑
<?php
/**
 * Front to the WordPress application. This file doesn't do anything, but
 * loads
 * wp-blog-header.php which does and tells WordPress to load the theme.
 *
 * @package WordPress
 */

/**
 * Tells WordPress to load the WordPress theme and output it.
 *
 * @var bool
 */
define('WP_USE_THEMES', true);

/** Loads the WordPress Environment and Template */
require( dirname( __FILE__ ) . '/wp-blog-header.php' );

```

```

1  <?php
2 /**
3  * Front to the WordPress application. This file doesn't do anything, but
4  * loads
5  * wp-blog-header.php which does and tells WordPress to load the theme.
6  *
7  * @package WordPress
8  */
9 /**
10 * Tells WordPress to load the WordPress theme and output it.
11 *
12 * @var bool
13 */
14 define('WP_USE_THEMES', true);
15
16 /** Loads the WordPress Environment and Template */
17 require( dirname( __FILE__ ) . '/wp-blog-header.php' );
18

```

准备好钱和耐心

首先，达人课是付费的，必须要为接下来的内容和后续持续的输出支付费用。

此外，由于在这个课程中，希望读者能够跟着做完，所以在课程中会带着读者购买一些服务，通过这些服务，来建设一个属于你的博客。

注意：费用不会很多，由于只是学习，会给用户推荐价格最低廉的产品，到真正使用时，只需要选择同类型的价格更高的产品即可。

学会提问

在提问之前，请读一读：[提问的智慧](#)。

掌握提问的技巧，能够最大限度的降低我们之间沟通的成本。让我们更加愉快和高效的沟通，做到更好的交流，节省大家的时间。

此外，还可以看看 [X-Y Problem](#)。

课程介绍

每一个程序员都需要一个博客，记录自己的所见所闻、分享自己的所思所想。与其使用一个受限制的第三方博客服务，不如花点小钱建一个不受限制的 WordPress 博客。

WordPress 作为世界上使用最多的 CMS（内容管理系统），成为大多数新手程序员的选择。实际上，WordPress 并不只是做一个博客的选择，小到博客，社区解决方案，对于 WordPress 来说都不是问题，海量的拓展库给了 WordPress 无限的可能。

作为开发者，除了能够独立开发产品，也应该能够使用现有的产品来简化自己的工作，将精力放在更加重要的部分。

该达人课将会讲解最基础的 WordPress 使用、WordPress 优化、WordPress 主题开发、WordPress 插件开发、为插件/主题加入多语言支持、为 WordPress 加入商城功能，以及最终我们将实践如何在其他应用中接入 WordPress。

整个课程分为多个章节，读者可以根据自己的情况，选择合适的章节阅读。

如果觉得有哪些内容需要学习，也可以在文章下方评论，收到反馈后，我会适当补充相关内容。

最后

都准备好之后，我们就开始下一节课的学习吧。

购买虚拟主机 && 设置域名解析

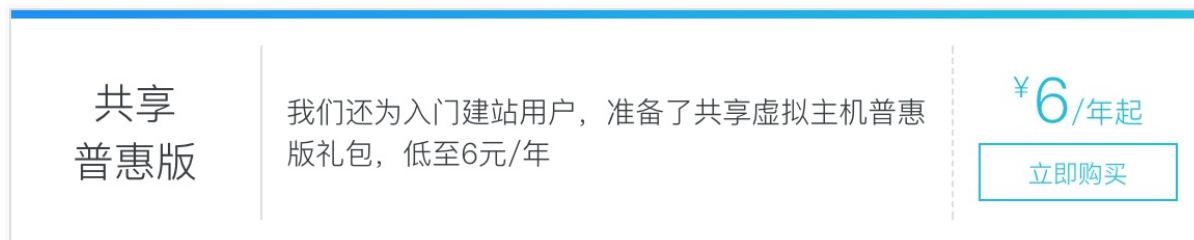
课程实践目标：一步一步完成 WordPress 站点的建立。

购买基础设施

在课程的一开始，要完成课程购买的内容，当跟着这个流程完成了购买后，也就基本掌握了虚拟主机和域名的设置，对于后续的课程、WordPress 的使用，都提供了帮助。

购买虚拟主机

进入[阿里云的虚拟主机页面](#)，找到其中的共享普惠版虚拟主机。



进入购买页面后，有多种可选方案，个人建议购买云解析的版本会比较实惠。

因为这个虚拟主机只是测试机，域名解析不和我们的主机绑定，另外两项和我们的主机绑定。

计划	存储	带宽	数据库	价格
共享虚拟主机普惠版	200M 网页空间	10G 月流量	20M 数据库	0元/年
网站木马查杀(安骑士)	防webshell 顶级后门查杀	防网站挂马 一键检测修复	清除提醒 恶意文件	6元/年
	6元			立即购买
共享虚拟主机普惠版	200M 网页空间	10G 月流量	20M 数据库	0元/年
网站日志分析(QBI)	流量趋势 PV、UV等	访问分析 地域来源	资源分析 洞悉兴趣点	6元/年
	6元			立即购买
共享虚拟主机普惠版	200M 网页空间	10G 月流量	20M 数据库	0元/年
云解析(DNS)	2G DNS防护流量	2W/QS DNS防护QPS	联通(分省) 智能解析线路	9.9元/2年
	9.9元			立即购买

购买时注意操作系统选择 Linux，不然基础环境就是 Asp.net 了。

云解析(DNS)

2G 2W/QS 联通 (分省)
 DNS防护流量 DNS防护QPS 智能解析线路

9.9元/2年

操作系统 Linux Windows

9.9元

立即购买

确认订单并支付。

购买域名

接下来购买域名，这里我们选择可以备案的后缀`.xyz`。

如何知道一个后缀是否可以备案？进入 <http://www.miitbeian.gov.cn/publish/query/indexFirst.action> 在左侧选择域名类型进行搜索查询，能够查到的就能备案。

在万网首页输入要注册的域名，将后缀选择为`.xyz`，会进入到查询页面。

gitchat-test

.xyz

查域名

.top ¥2 | .xyz ¥6 | .cc ¥16 | .ltd ¥14 | .shop ¥12 | 价格总览>

公告 域名抢注 预释放域名 域名续费

如果查询到该域名未注册，就可以将其加入清单中，并单击去结算按钮。

The screenshot shows the AliCloud Domain Purchase interface. It lists several domain names with their prices and options:

- gitchattest.xyz(未注册)**: 6 元/首年 更多价格 > 加入清单
- gitchattest.com(未注册)**: 55 元/首年 更多价格 > 加入清单
+邮件推送资源包 (1元/年) 立即抢购
+独享云虚拟主机经济版 (年) 套餐价353元 加入清单
- gitchattest.net(未注册)**: 62 元/首年 更多价格 > 加入清单
- gitchattest.cn(未注册)**: 29 元/首年 更多价格 > 加入清单
+邮件推送资源包 (1元/年) 立即抢购
- gitchattest.top(未注册)**: 2 元/首年 更多价格 > 加入清单

To the right, there's a summary cart section with a total of 1 item: gitchattest.xyz. It includes a "去结算" (Go to Settlement) button.

在新的页面中，确认订单信息，并选择域名的所有者。

The screenshot shows the Order Confirmation page. It displays the following information:

- 确认订单**: Product name: .xyz 域名, Product content: gitchattest.xyz, Selection: 1年, Price: ￥6.00, Operation: 删除 (Delete).
- 支付**: Price: ￥6.00.
- 支付成功**.
- 推荐购买** section:
 - 轻量应用服务器**: 拥有服务器完整root权限, 支持各开发语言网站, 一键HTTPS配置, 灵活建站首选。立即购买 45元/月 加入购物车
 - 独享主机经济版**: 预装网站环境, 赠送正版数据库, 独享服务器资源和IP, 网站快速易推广! ￥149元/半年, 低至0.8元/天! 加入购物车
 - 邮件推送**: 企业办公、营销必备, 邮件发送量: 1000封 加入购物车
 - 云解析经济版**: 域名必配安全服务, 流量攻击防护: 2G, 域名攻击防护QPS: 20,000 加入购物车
- 如何选择?** (How to choose?)
- 收起** (Collapse) button.

此外，记得勾选域名隐私保护，不然接下来可能会有无数的骚扰电话、骚扰邮件发送给你。

如果域名信息填写的内容是虚假的，ICANN 是有权收回你的域名的，所以尽量选择真实的信息+域名隐私保护。

域名隐私保护服务：

免费开启，同意《[域名隐私保护服务条款](#)》

重要提示：以上选项对“预订域名”无效；如果是“预订域名”，注册成功后，请到域名控制台手动开启。

域名注册信息可被公众查询，为避免电话骚扰，我们推荐默认免费开启隐私保护服务，真实注册信息将不会对公

确认完信息，支付订单即可。

管理虚拟主机

进入 <https://netcn.console.aliyun.com/core/host/list2#> 管理控制台，就可以看到虚拟主机了。

全部主机 急需续费主机 导出列表

主机名: 例如: hyw15312 主机域名: 例如: mydomain.com 主机备注: 例如: hyw15312 到期日: [] 至 [] 搜索

<input type="checkbox"/> 主机名/主机备注	主机类型/配置	主机域名	IP地址	到期日	来源	操作
qxu1098390026 gitchat演示 编辑	共享虚拟主机普惠版 空间: 200M 流量: 10GB	[REDACTED]	[REDACTED]	2018-11-23	BC	续费 升级 管理 更多操作

[设置自动续费](#) [主机续费](#) [转至其他账号](#)

共有1条, 每页显示: 10条 « « 1 » » [GO](#)

单击“管理”链接，即可进入到虚拟主机的管理界面。

首先，需要设置一系列的账户密码：

密码初始化设置 > 账号安全设置 > 设置完成

为了提高系统安全性，请对当前主机的管理密码进行初始化设置。

管理控制台登录密码: [] 密码确认: []
FTP登录密码: [] 密码确认: []
MySQL数据库登录密码(qdm165376226): [] 密码确认: []

[保存, 下一步](#)

这里的密码要记清楚，稍后我们要用到。

然后验证手机或邮箱：

请选择您常用的手机或邮箱进行绑定，以便于日后重置密码等操作验证、接收主机通知等使用。

验证方式: 手机验证 邮箱验证

手机号码:

59秒后重新获取验证码

验证码已发送到您的手机，15分钟内有效，请勿泄露。

验证码:

请输入验证码

[保存设置](#)

设置完成后，会看到一系列的账户信息，可以将其保存在笔记本中。

不妨试试印象笔记？“职场高效率：用印象笔记来提升你的工作效率”

• 安全账号信息

如已绑定手机号或邮箱丢失, 请致电万网客服。

用户名: qxu1098390026

绑定手机号: [REDACTED] [更换绑定](#)

绑定邮箱: 无 [绑定邮箱](#)

• 主机账号信息

主机管理控制台地址:

<https://cp.aliyun.com> [\[加入收藏\]](#)

用户名: qxu1098390026

FTP登录主机地址:

qxu1098390026.my3w.com

用户名: qxu1098390026

[\[FTP客户端下载和使用手册\]](#)

MySQL数据库连接地址:

qdm165376226.my3w.com

用户名: qdm165376226

单击上方的进入管理控制台按钮, 可以进入到我们的虚拟主机页面。

在这里, 我们可以看到一系列可能会用到的设置项。

环境基础设置

在上传文件前需要修改一些配置。

绑定域名

万网的虚拟主机和域名对接的很好, 我们可以很方便的绑定域名。

单击左侧菜单中的**域名管理 | 域名绑定**命令, 在新的页面中单击“绑定域名”按钮, 并在弹出的对话框中设置域名, 或者在下方的选择框中选择要绑定的域名。



绑定成功后会看到, 它会提示域名未备案, 所以暂时我们还不能使用自己的域名去访问, 需要将自己的域名备案后才能访问。

备案状态

✓ 已备案

✗ 未备案，立即备案

✗ 未备案，立即备案

上传文件

首先，下载 WordPress 的源代码，前往 <https://cn.wordpress.org/>，单击页面中的**下载 WordPress 4.9**按钮，下载源代码。

下载WordPress 4.9

.zip — 10.4 MB

[下载.tar.gz — 9.7 MB](#)

资源

安装或使用WordPress的问题 **查看**

下载完成后，将源码解压出来。

上传文件我们使用的是 FTP 协议，这里使用的软件是 FileZilla，读者也可以到软件的官网下载：<https://filezilla-project.org/>

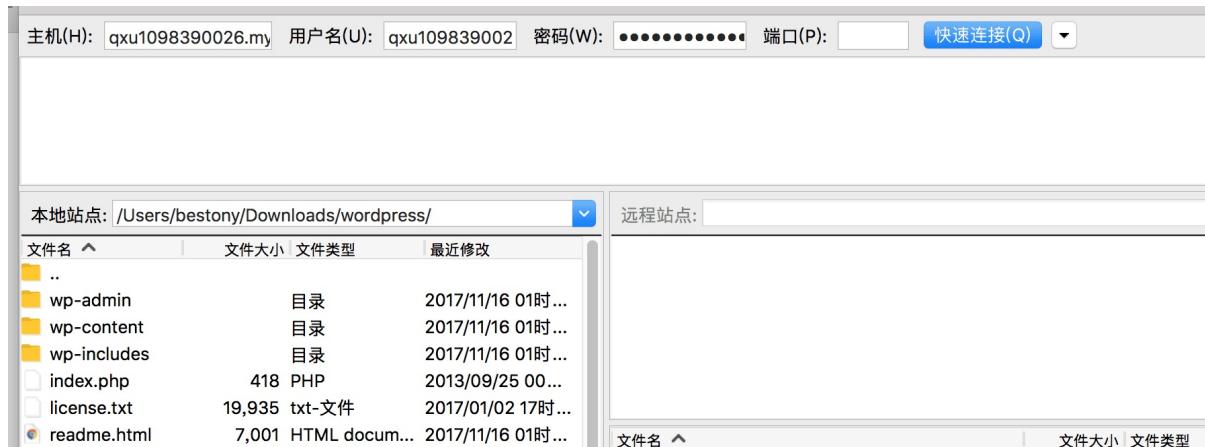
回到虚拟主机管理控制台，单击上方菜单栏中的**站点信息**按钮，进入到控制台主页面。我们可以看到 FTP 链接信息。

FTP 登录用户名：qxu1098390026

FTP 登录密码：***** [\[重置密码\]](#)

FTP 登录主机地址：qxu1098390026.my3w.com

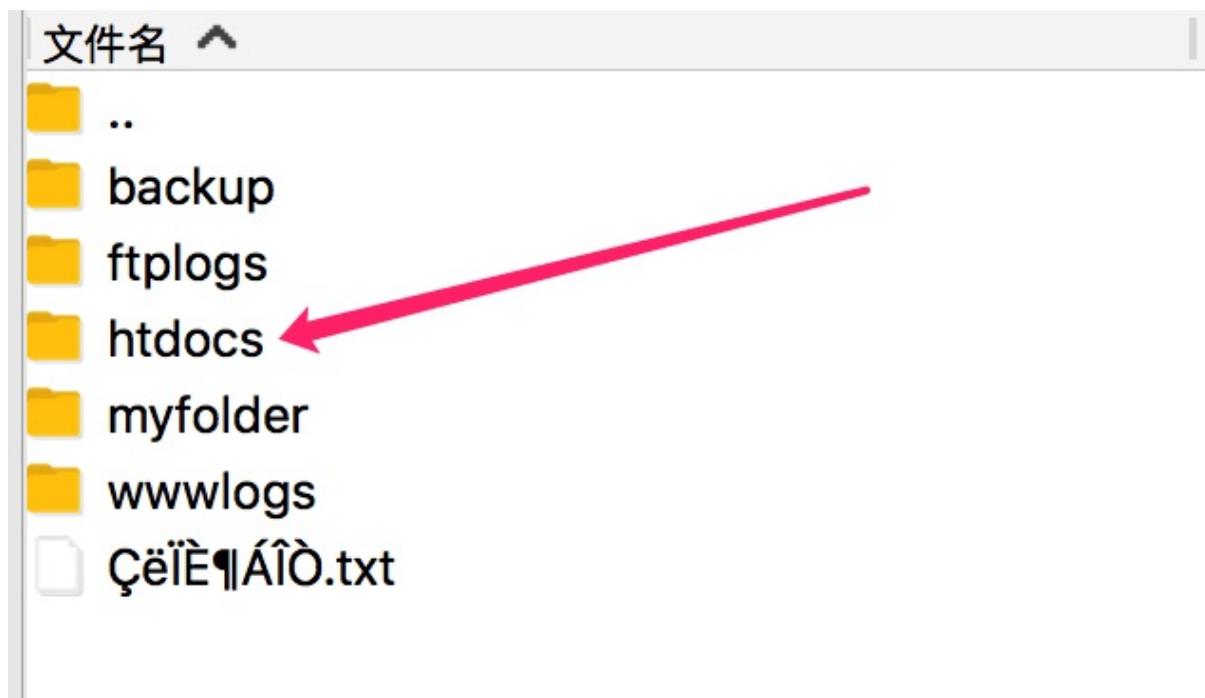
打开 FileZilla，将这几项分别填入输入框内。



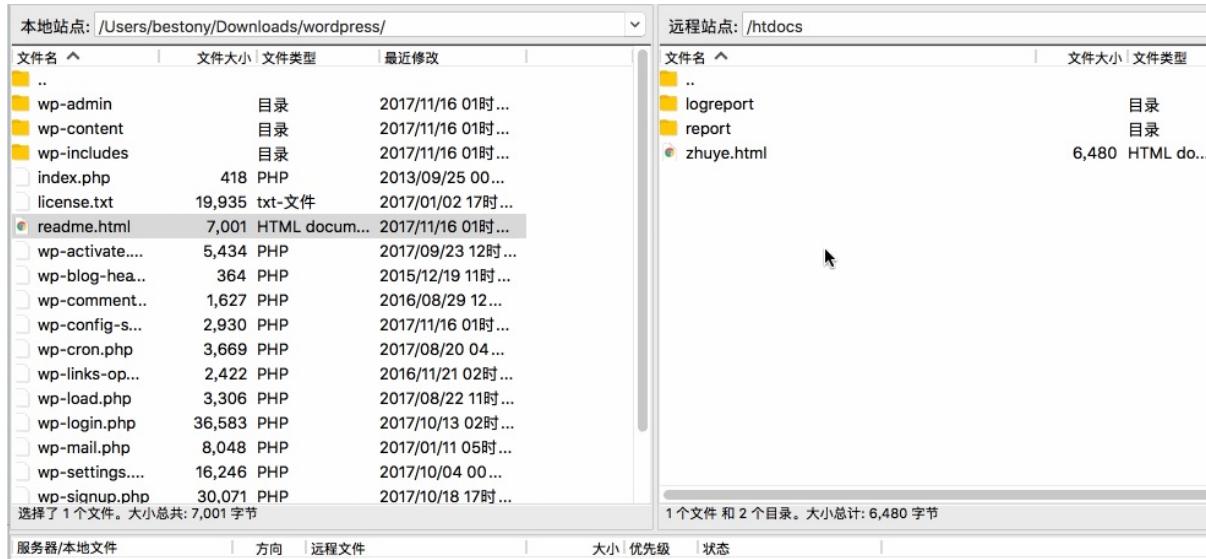
单击快速链接。

连接成功后，在左侧的本地站点链接中，找到 WordPress 源码文件目录，就像上图中那样。

右侧的远程连接则进入到 htdocs 目录下：



删除其中的 zhuye.html，并回到左侧，选中所有文件，将其上传到当前目录。



等待其上传完成。

访问测试

回到管理控制台，复制我们的临时域名，到浏览器中打开，可以看到这样的界面。



输入 FTP 密码，就会看到正常的页面了。



安装 WordPress

当备案完成后，进入备案好的域名，会重新看到 WordPress 的安装界面。



单击**现在就开始!**按钮，输入数据库信息，并单击提交按钮：

请在下方填写您的数据库连接信息。如果您不确定，请联系您的服务提供商。

数据库名	<input type="text" value="qdm165376226_db"/>	将WordPress安装到哪个数据库?
用户名	<input type="text" value="qdm165376226"/>	您的数据库用户名。
密码	<input type="text" value="qwe123asd123"/>	您的数据库密码。
数据库主机	<input type="text" value="qdm165376226.my3w.com"/>	如果localhost不能用，您通常可以从网站服务提供商处得到正确的信息。
表前缀	<input type="text" value="wp_"/>	如果您希望在同一个数据库安装多个WordPress，请修改前缀。

数据库信息可以在虚拟主机管理控制台获取：

账号信息

主机管理控制台用户名: qxu1098390026	主机管理控制台密码: ***** [重置密码] 此账号 密码也用作 备案产品验证	
FTP登录用户名: qxu1098390026	FTP登录密码: ***** [重置密码]	FTP登录主机地址: qxu1098390026.my3w.com
数据库名称: qdm165376226_db	数据库类型: MySQL	数据库连接地址: qdm165376226.my3w.com
数据库用户名: qdm165376226	数据库管理密码: ***** [重置密码]	

在新的页面单击“立即安装”按钮，会进入到站点信息配置，设置具体的站点信息，然后单击安装 WordPress按钮，就会开始安装。

需要信息

您需要填写一些基本信息。无需担心填错，这些信息以后可以再次修改。

站点标题

GitChat Test Page

用户名

mySuperAdmin

用户名只能含有字母、数字、空格、下划线、连字符、句号和 "@" 符号。

密码

J9dFSosaP8GCvXnzId



隐藏

强

重要：您将需要此密码来登录，请将其保存在安全的位置。

您的电子邮件

gitchat@wordpress.dev

请仔细检查电子邮件地址后再继续。

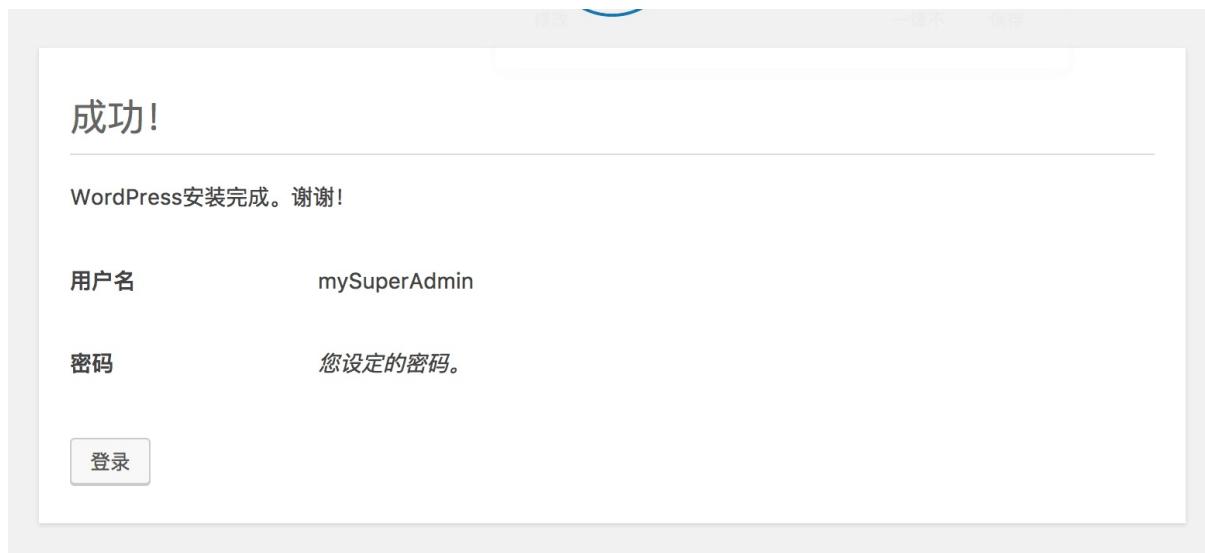
对搜索引擎的可见性

建议搜索引擎不索引本站点

搜索引擎将本着自觉自愿的原则对待WordPress提出的请求。并不是所有搜索引擎都会遵守这类请求。

安装WordPress

当看到这样的界面时就说明安装好了，这样就完成了最基本的 WordPress 部署了。



WordPress 的环境配置

课程实践目标：在自己的电脑上完成基础环境的配置。

在本地的 Mac 电脑上配置开发环境

MacOS 相比于 Windows 有很好的命令行支持，在开发环境上也更友好，系统自带运行 WordPress 需要的 PHP 和 Apache。

不过，为了能够更简单的进行 WordPress 开发，建议读者使用 Laravel 的 Valet 组件来进行 WordPress 开发。

安装 homeBrew

Valet 需要 PHP 7 环境和 composer 来运行，同时还需要 homebrew 来安装依赖环境（Nginx、Ngrok、Dnsmasq）。而 PHP 7 和 composer 环境也可以通过 homeBrew 来安装，所以在开始先来安装 homeBrew。

打开 LaunchPad，找到终端应用，在终端应用中粘贴如下命令，并回车。

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

homeBrew 的安装需要 XCode Command Line 的支持，可以通过执行 xcode-select --install 来安装。

待命令执行完毕后，HomeBrew 就安装完成了。可以执行 brew 命令，查看系统返回，如果显示如下所示，则说明安装完成了。

[cloud ~ brew

Example usage:

```
brew search [TEXT|/REGEX/]
brew (info|home|options) [FORMULA...]
brew install FORMULA...
brew update
brew upgrade [FORMULA...]
brew uninstall FORMULA...
brew list [FORMULA...]
```

Troubleshooting:

```
brew config
brew doctor
brew install -vd FORMULA
```

Developers:

```
brew create [URL [--no-fetch]]
brew edit [FORMULA...]
https://docs.brew.sh/Formula-Cookbook.html
```

Further help:

```
man brew
brew help [COMMAND]
brew home
```

安装 PHP 7 和 Composer

在终端内执行如下命令，来安装 PHP 7 执行环境：

```
brew tap homebrew/php # 添加 php 库
brew update # 更新
brew install php71 # 安装 php71
```

当 PHP 7 安装完成后，我们开始安装 Composer，在命令行中执行如下命令：

```
php -r "copy('https://install.phpcomposer.com/installer', 'composer-setup.php');"
php composer-setup.php
php -r "unlink('composer-setup.php');"
mv composer.phar /usr/local/bin/composer
```

通过上述的命令，把 Composer 安装到本地目录了。

将其移动到 `/usr/local/bin/composer` 主要是为了方便全局运行，这样我们就不用总是使用相对路径来调用 Composer 了。

```
❸ Downloads php -r "copy('https://install.phpcomposer.com/installer', 'composer-setup.php');"
❸ Downloads php composer-setup.php

All settings correct for using Composer
Downloading...

Composer (version 1.5.2) successfully installed to: /Users/bestony/Downloads/composer.phar
Use it: php composer.phar

❸ Downloads php -r "unlink('composer-setup.php');"
❸ Downloads mv composer.phar /usr/local/bin/composer
```

安装完成后，我们要为 Composer 来配置国内镜像（受不可抗力因素影响，官方镜像下载速度缓慢）。

Composer 支持项目加速和全局加速，我们这里没有 Composer 项目，所以选择全局加速。在命令行中执行如下命令：

```
composer config -g repo.packagist composer https://packagist.phpcomposer.com
```

安装 Valet

在命令行中执行如下命令来安装 Valet：

```
composer global require laravel/valet
valet install
```

命令执行完成后，系统就成功帮助我们安装了 Valet 开发环境。

```
❸ Downloads composer global require laravel/valet
Changed current directory to /Users/bestony/.composer
Using version ^2.0 for laravel/valet
./composer.json has been updated
Loading composer repositories with package information
Updating dependencies (including require-dev)
Nothing to install or update
Generating autoload files
❸ Downloads valet install
```

这时执行 `ping gitchat.dev`，可以看到返回的 IP 地址是 127.0.0.1，则说明环境配置成功。

◆ **Downloads** ping gitchat.dev

```
PING gitchat.dev (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.044 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.119 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.098 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.060 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.070 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.072 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.074 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.077 ms
```

配置 Valet

安装完 Valet，我们还要进行简单的配置，来更好的使用。

首先，来创建网站文件的目录：

```
mkdir -p ~/Developer/php # 创建目录
```

然后，将这个目录设置为 Valet 的主目录：

```
cd ~/Developer/php # 进入目录
valet park # 配置 Valet
```

这样，在 PHP 目录下创建的任何一个文件夹，都可以通过 [文件夹名].dev 的域名进行访问。

安装 MySQL

WordPress 需要 MySQL 作为数据库，接下来我们来安装数据库。

```
brew install mysql
```

安装完成后，可以执行 brew services start mysql 设置数据库的开机自启动。

```

1. bestony@MBP: ~/Downloads (zsh)
● Downloads brew install mysql

Updating Homebrew...
^C==> Downloading https://mirrors.ustc.edu.cn/homebrew-bottles/bottles/mysql-5.7
.2
#####
==> Pouring mysql-5.7.20.high_sierra.bottle.tar.gz
==> Caveats
We've installed your MySQL database without a root password. To secure it run:
  mysql_secure_installation

MySQL is configured to only allow connections from localhost by default

To connect run:
  mysql -uroot

To have launchd start mysql now and restart at login:
  brew services start mysql
Or, if you don't want/need a background service you can just run:
  mysql.server start
==> Summary
🍺 /usr/local/Cellar/mysql/5.7.20: 324 files, 233.7MB
● Downloads ■

```

可以执行 `cat ~/.mysql_secret` 来获取 mysql 默认的 root 用户的密码。这个密码稍后要用到，要记下来。

如果提示为 `No such file or directory`，则说明 root 密码为空。

安装 WordPress

接下来，我们来安装 WordPress。

首先，创建目录，并下载 WordPress 的文件。

```

cd ~/Developer/php # 进入目标目录
mkdir wordpress # 创建新的文件夹
cd wordpress # 进入新的文件夹
wget https://wordpress.org/latest.zip
unzip latest.zip
mv wordpress/* ./

```

然后创建数据库：

```
mysql -uroot -p -e "create database wordpress"
```

执行命令后，可能会让你输入密码，输入密码后并回车，就成功创建了数据库了。

```

◆ wordpress mysql -uroot -p -e "create database wp"
Enter password:
◆ wordpress mysql -uroot -p -e "show databases"
Enter password:
+-----+
| Database           |
+-----+
| information_schema |
| 1001cmd            |
| mysql               |
| performance_schema |
| sys                 |
| wordpress          |
| wp                  |
+-----+

```

在浏览器中打开 <http://wordpress.dev/>，就可以一步一步按部就班的安装了。

The screenshot shows the initial setup screen for WordPress. At the top is the classic blue 'W' logo. Below it, a message reads: "请在下方填写您的数据库连接信息。如果您不确定，请联系您的服务提供商。" (Please fill in your database connection information. If you're unsure, contact your service provider.)

数据库名	<input type="text" value="wordpress"/>	将WordPress安装到哪个数据库？
用户名	<input type="text" value="用户名"/>	您的数据库用户名。
密码	<input type="text" value="密码"/>	您的数据库密码。
数据库主机	<input type="text" value="localhost"/>	如果localhost不能用，您通常可以从网站服务提供商处得到正确的信息。
表前缀	<input type="text" value="wp_"/>	如果您希望在同一个数据库安装多个WordPress，请修改前缀。

At the bottom left is a "提交" (Submit) button.

在输入数据库信息时，设置数据库名为 `wordpress`，用户名为 `root`，密码为上方我们获取到的密码。其他两项不同，单击“提交”按钮。



单击“进行安装”按钮，在新的页面中设置基本信息，并单击“安装 WordPress”按钮。

需要信息

您需要填写一些基本信息。无需担心填错，这些信息以后可以再次修改。

站点标题	我的测试站点
用户名	admin
用户名只能含有字母、数字、空格、下划线、连字符和 "@" 符号。	
密码	<input style="background-color: #e0f2e0; border: 1px solid #80c080; width: 150px; height: 20px;" type="password" value="*hq&Bj0sxAzH1wf%y("/> 隐藏
重要：您将需要此密码来登录，请将其保存在安全的位置。	
您的电子邮件	<input style="width: 150px; height: 20px;" type="text" value="gitchat@gitchat.cn"/>
请仔细检查电子邮件地址后再继续。	
对搜索引擎的可见性	<input type="checkbox"/> 建议搜索引擎不索引本站点 搜索引擎将本着自觉自愿的原则对待WordPress提出的请求。并不是所有搜索引擎都会遵守这类请求。
安装WordPress	

这样，我们的本地开发环境就部署完成了。

在 Windows 电脑上配置开发环境

相比于 Mac、Windows 的配置要简单很多，只需要安装一个软件即可 PHPStudy。

安装 PHPstudy

访问 <http://phpstudy.net/download.html> 下载页面，下载 PHPstudy。

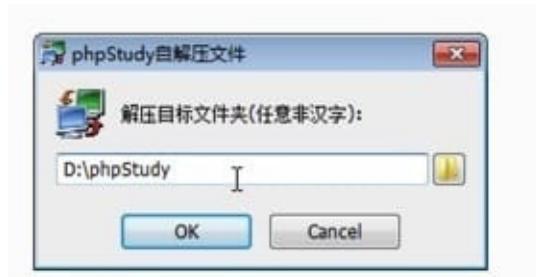
我们使用 33MB 的解压版：



下载后, 将得到的压缩包解压, 执行其中的 exe 安装文件。

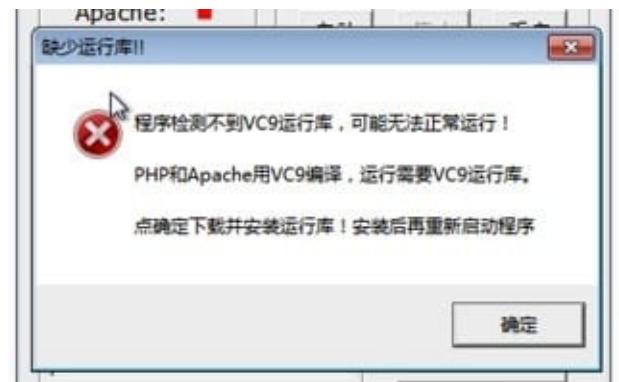


弹出的确认窗口我们不做更改, 直接确认。



并在弹出初始化的窗口中单击 是按钮。

可能会提示读者安装 VC9 运行库:



单击“确定”按钮, 会引导我们到下载页面, 下载对应的支持库, 安装即可。

如果防火墙提示拦截 Apache 和 MySQL 时, 允许即可。

安装 WordPress

配置完 WordPress 后, 我们开始安装 WordPress。



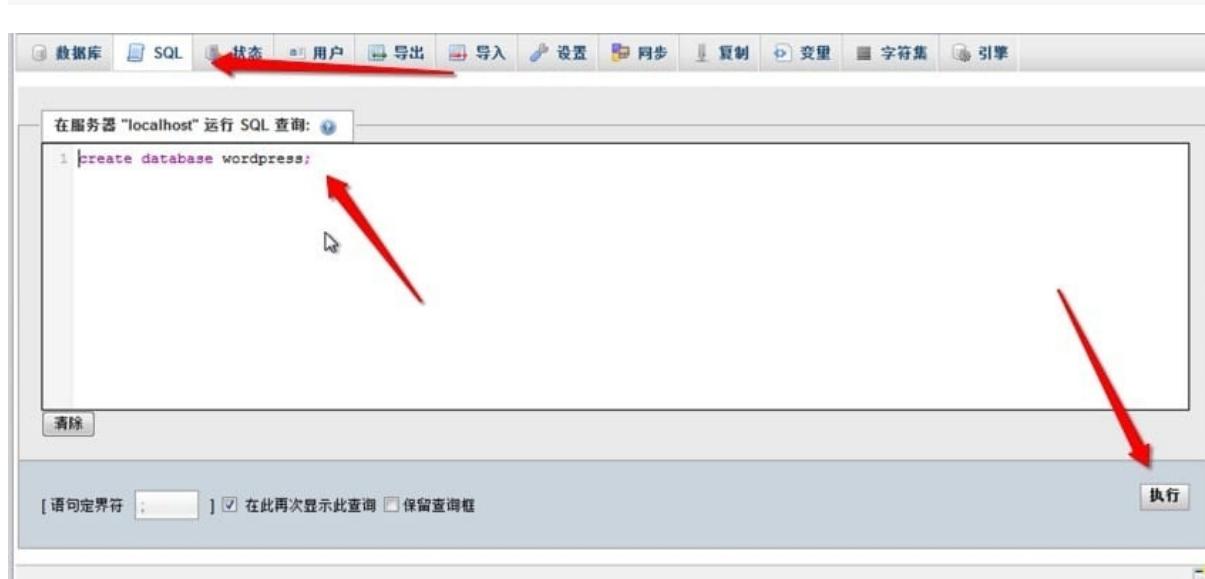
右击托盘中的 phpstudy 图标，选择其中的网站根目录，就会进入到根目录，删除其中的 `l.php` 和 `phpinfo.php`。

打开浏览器，访问 <https://wordpress.org/latest.zip>，可以下载到最新的安装包。下载完成后，将其解压，将解压出来的 WordPress 目录中的所有文件都复制，粘贴到我们刚刚打开的网站根目录。

打开 <http://localhost/phpMyAdmin> 使用用户名 root，密码 root 登录。

登录后，单击上方的 tab 中的 **SQL**，输入如下代码，并单击“执行”按钮：

```
create database wordpress;
```



这样，我们就成功的创建了一个新的 WordPress 数据库。

接下来，在浏览器中打开 <http://localhost> 就可以安装了，安装数据库时用到的用户名和密码都是 root，数据库名为 wordpress。其他安装步骤可以参考 Mac 开发环境配置的相关部分。

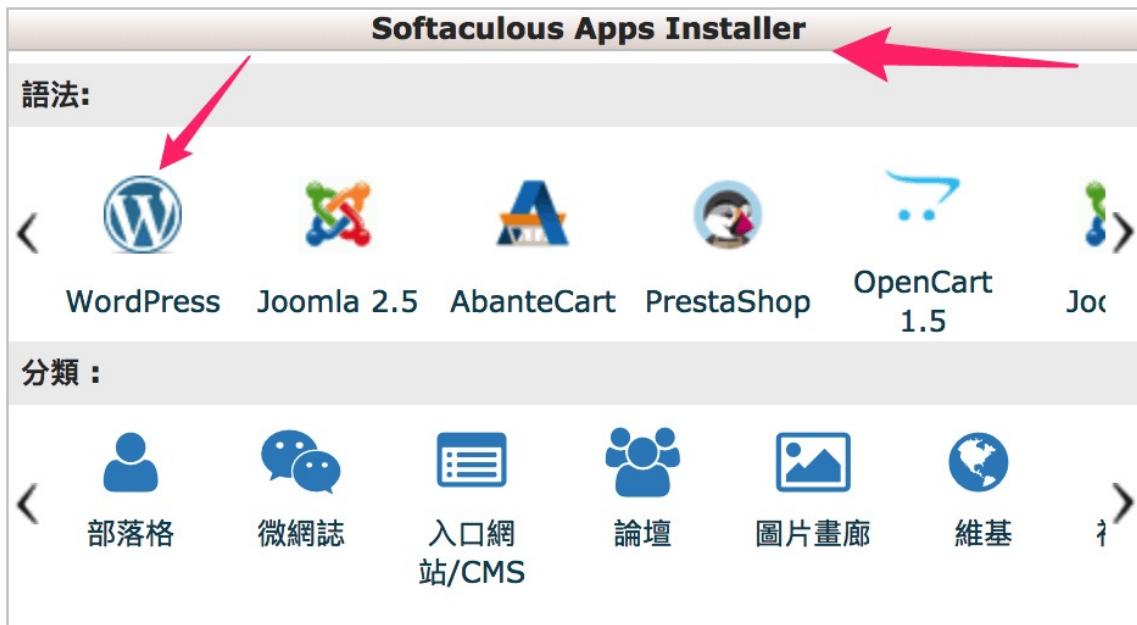
在 cPanel 上安装 WordPress

cPanel 是我们经常使用的一款虚拟主机面板，功能强大。如果购买了一些国外的主机商的虚拟主机，一般提供的都是 cPanel。

在 cPanel 上快速安装 WordPress

有的主机的 cPanel 集成了 Softaculous 的安装工具，这样我们可以通过简单的配置来安装 WordPress。

可以登录你的 cPanel，在其中寻找 **Softaculous App Installer**。



单击其中的 WordPress 会进入到安装界面。

語法安裝

選擇協議
如果您的網頁具有 SSL, 那們請選擇 HTTPS 安全協議.

http://

選擇域名
請選擇域名來安裝語法.

www.yourdomain.com

安裝資料夾
選擇一個要安裝資料夾於您的域名內且 需要是不存在的. 例如
要安裝於 http://mydomain/dir/ 只需要輸入 dir. 如果只要安
裝在家目錄例如 http://mydomain/ 則保留為空.

My Blog

My WordPress Blog

開啟多網站 (WPMU)

網站設定

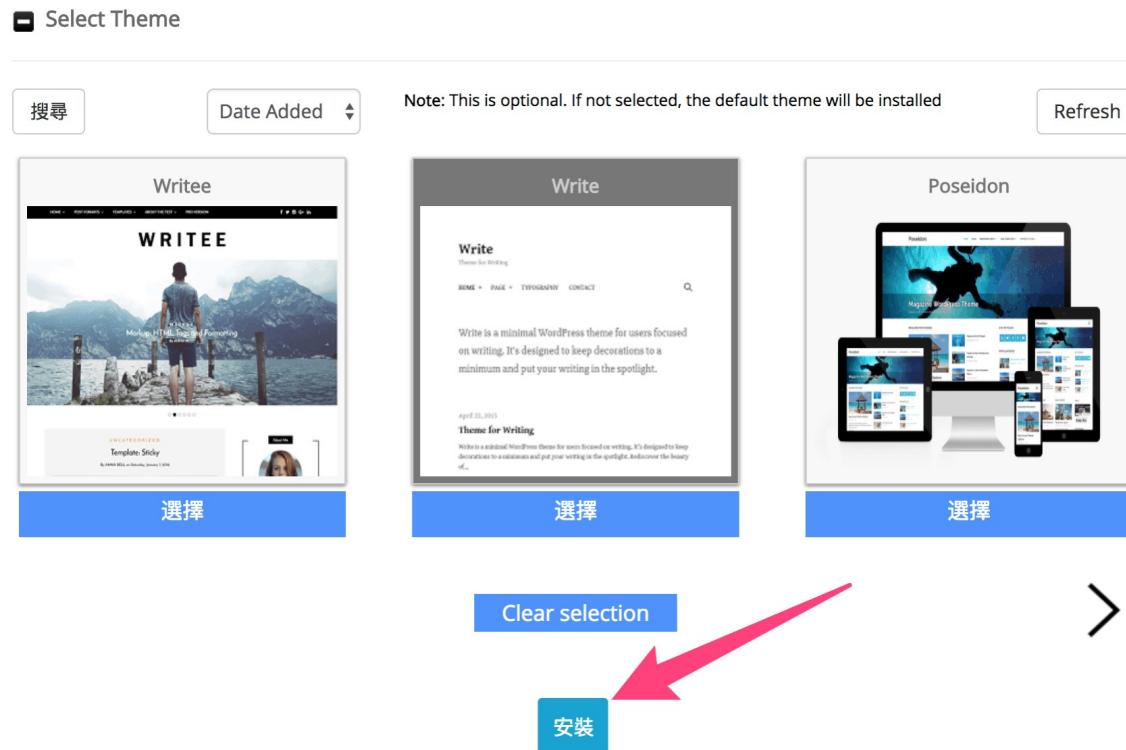
網站名稱
My Blog

網站介紹
My WordPress Blog

管理員帳號

管理員用戶名稱
admin

填写其中的安装信息，点击最下方的安装：



稍等片刻，就安装成功了。

恭喜，語法已經成功安裝

WordPress 已經成功安裝於：

<http://ziyu.com/wp-admin/>

我們希望安裝程序是簡單的。

注意: Softaculous 只是一個自動安裝器，不需負責任何其他個別的安裝程序。請瀏覽該語法供應商的網站獲取支援!

誠摯，
Softaculous 自動安裝器

[回到總覽](#)

Tell your friends about your new site :

I just installed #WordPress via #[[Softaculous]]
#PHP

[Tweet!](#)

可以单击上面的管理员后台的链接，登录 WordPress 后台。

在 cPanel 中使用常规方法安装 WordPress

虽然快捷安装方便，但并不是每个主机商都会集成，所以我们再补充一个常规的安装方法。

下载 WordPress 安装包

访问 <http://wordpress.org/latest.zip> 即可下载到最新的 WordPress 安装包。

上传 WordPress 的安装包

打开 cPanel , 找到文件管理器:



在文件管理器中找到 `public_html` 目录，单击进入该目录。

The screenshot shows the 'File Manager' interface within cPanel. On the left, there's a tree-view navigation pane showing the directory structure under `/home/ziyuanbaobe`. A red arrow points to the `public_ftp` folder. The main pane displays a list of files and folders with columns for Name, Size, Last Modified, and Type. A red arrow points to the `public_html` folder in this list.

Name	Size	Last Modified	Type
mail	4 KB	2016-9-8上午8:05	mail
public_ftp	4 KB	2016-7-16下午11:14	publicftp
public_html	4 KB	今天下午4:43	publichtml
ssl	4 KB	2017-11-6下午4:24	httpd/unix
tmp	4 KB	2016-7-18上午5:10	httpd/unix
var	4 KB	2016-7-17上午7:50	httpd/unix
.bash_logout	18 字节	2016-3-31上午7:29	text/x-gene
.bash_profile	176 字节	2016-3-31上午7:29	text/x-gene
.bashrc	124 字节	2016-3-31上午7:29	text/x-gene
.contactemail	21 字节	2016-7-16下午11:14	text/x-gene
.ftpquota	14 字节	2017-10-17下午3:26	text/x-gene
.gemrc	143 字节	2016-7-16下午11:14	text/x-gene
.htaccess	1,000 字节	2016-7-17下午4:59	text/x-gene
.lastlogin	128 字节	今天下午4:23	text/x-gene
.zshrc	658 字节	2016-3-25下午7:11	text/x-gene

进入该目录后，单击“上传”按钮：

在新的页面中选择我们刚刚下载的 latest.zip 文件，进行上传。

选择要上传至“/home/ziyuanbaohe/public_html”的文件。

允许上传的最大文件大小: 9.76 GB

覆盖现有文件

将文件放在此处以开始上传

或

选择文件

latest.zip

6%

576 KB / 8.50 MB (6%) complete

等待其上传成功后，就关掉这个页面。

删除成功后，会变成绿色的进度条。

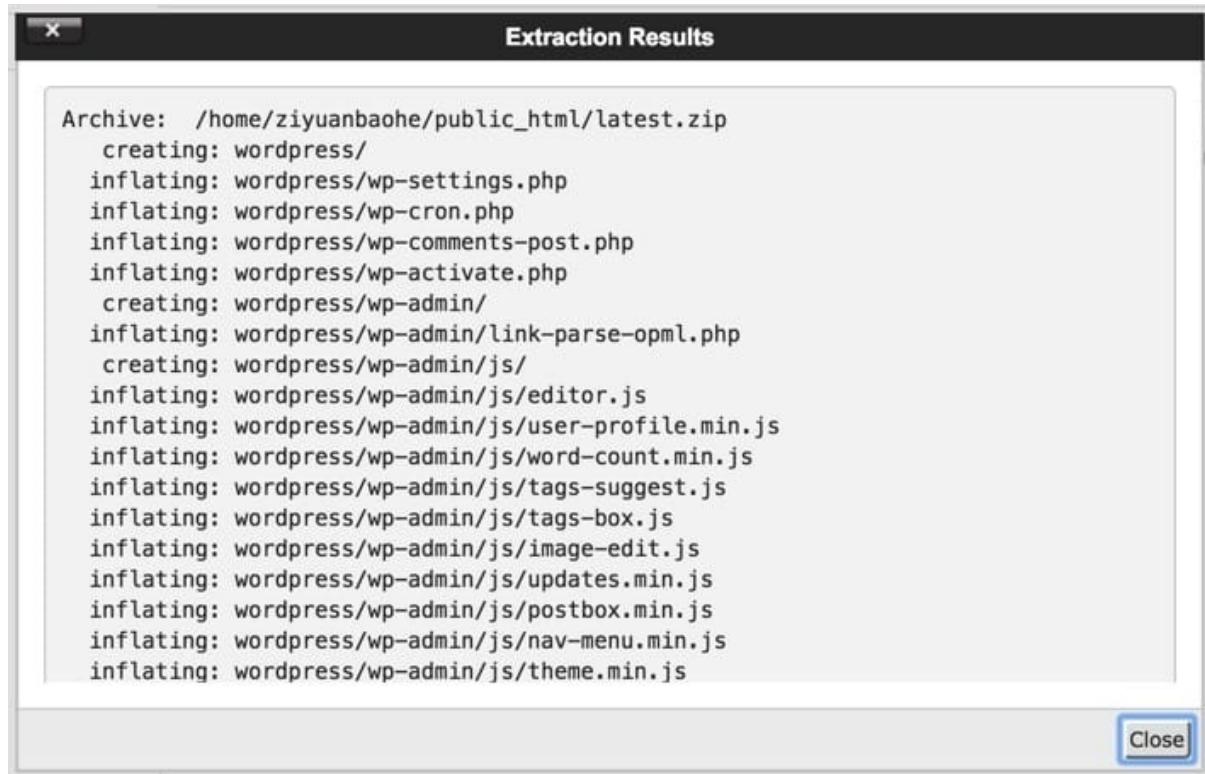
回到文件管理器页面，单击“重新加载”按钮，就可以看到刚刚上传的文件了。

Name	Size	Last Modified	Type
latest.zip	8.5 MB	今天下午4:48	pa

选中它，单击上方的“提取”按钮：



并在弹出的窗口中选择**Extract File**, 稍等片刻, 就解压成功了。



单击解压出来的 wordpress 文件夹, 进入该文件夹。

勾选菜单栏中的“全选”复选框, 再单击上方的“移动”按钮。



在弹出窗口中, 将移动路径改为 `/public_html` , 并单击**Move File**按钮。

这样我们就完成了 WordPress 文件上传了。

创建数据库

接下来回到 cPanel 管理界面，找到其中的 MySQL 数据库向导。点击进入向导。



分别输入数据库名、用户名和密码：

第 1 步：创建数据库

新建数据库：

ziyuanbaohe_ db

注意：最多 51 个字符。 db

下一步

第 2 步：创建数据库用户：

用户名：

ziyuanbaohe_ db

注意：最多 35 个字符。

密码：

密码(再次输入)：

强度

强 (68/100)

密码生成器

创建用户

在选择权限时，勾选“所有权限”复选框：

第 3 步：向数据库添加用户。

用户: ziyuanbaohe_db
数据库: ziyuanbaohe_db

所有权限

<input checked="" type="checkbox"/> ALTER	<input checked="" type="checkbox"/> ALTER ROUTINE
<input checked="" type="checkbox"/> CREATE	<input checked="" type="checkbox"/> CREATE ROUTINE
<input checked="" type="checkbox"/> CREATE TEMPORARY TABLES	<input checked="" type="checkbox"/> CREATE VIEW

然后单击“下一步”按钮，就可以看到添加成功的界面。

cPanel 11

MySQL® 数据库向导

第 4 步：完成任务

 用户“ziyuanbaohe_db”已添加到数据库“ziyuanbaohe_db”。

添加另一数据库。
为 MySQL 数据库添加另一用户。
返回到 MySQL 数据库。
[返回主页](#)

稍后安装 WordPress 时，就可以使用刚刚创建的数据库去安装 WordPress 了。

安装 WordPress

接下来，就可以访问你的虚拟主机绑定的域名去安装 WordPress 了。具体的安装部分，可以参考上方的 Mac 开发环境配置部分的相关内容。

总结

在上面的内容中，我们完成了本地开发环境的配置和以 cPanel 为例的虚拟主机的配置。如果有什么问题不清楚，可以在下方评论。如果后续觉得需要补充其他环境的配置说明，也可以在下方评论提问。

WordPress 基本使用

课程实践目标：掌握 WordPress 的使用。

通用内容

控制内容的展示

模块

概览 活动 快速草稿 WordPress活动及新闻 Welcome

每个页面都可以单击右上角的显示选项来控制页面显示什么内容，方便调整自己的页面布局，符合自己的写作习惯。

帮助模块

概述	您可以依您的喜好和工作方式来安排仪表盘页面的布局。大部分其它管理页面也支持重新排列功能。	更多信息： 仪表盘文档
导航	显示选项 — 使用“显示选项”选项卡来选择要显示的仪表模>块。	支持论坛
布局	拖放自如 — 要重新排列模块，按住模块的标题栏，将其拖>到您希望的位置，在灰色虚线框出现后松开鼠标即可调整模块的位置。	
内容	管理模块 — 点击模块的标题栏即可展开或收起它。另外，>有些模块提供额外的配置选项，在您将鼠标移动到这些模块的标题栏上方时，会出现“配置”链接。	帮助 ▲

WordPress 每个页面都有一个帮助模块，在帮助模块中，可以看到当前页面的具体帮助内容，假如不知道当前页面怎么用，除了在达人课的读者圈中提问，也可以单击上方的“帮助”按钮来寻找答案。

仪表盘

仪表盘是我们登录 WordPress 后台后首先会看到的内容。

Welcome

欢迎使用WordPress！
我们准备了几个链接供您开始：

开始使用 自定义您的站点 或更换主题	接下来 撰写您的第一篇博文 添加“关于”页面 查看站点	更多操作 管理边栏小工具和菜单 打开/关闭评论功能 了解更多新手上路知识
--	--	---

Welcome 是新用户引导，单击“不再显示”链接后，下次就不会再出现。

如果后面还想看，可以单击右上角的“显示”选项：



勾选其中的 Welcome 复选框，这个界面就会出现了。



概述

概览

1篇文章 1个页面
1条评论

WordPress 4.8.3, 使用Twenty Sixteen主题。

概述里展示了我们博客的最基础的几个数据：博客文章数、博客评论数、单页数目、当前版本和当前使用的主题。

单击其中的超链接，就会跳转到对应的列表。例如，单击“1篇文章”链接，就会跳转到文章列表，可以对文章进行管理。

快速草稿

快速草稿

在想些什么?

保存草稿

草稿 **查看所有**

- 测试4 2017年11月15日
- 测试4
- 测试3 2017年11月15日
- 测试3
- 测试2 2017年11月15日
- 测试2

快速草稿可以帮助我们在仪表盘记录心中的所念所想，而无需加载完整的 WordPress 编辑器来编辑格式。当有一些小的灵感时，可以在这里快速的输入、并保存，后续再完善，以免忘记了这个想法。

活动

活动

最近发布

下午6:08 11月12日 世界, 您好!

近期评论

 由一位WordPress评论者发表在《世界, 您好!》
嗨, 这是一条评论。要开始审核、编辑及删除评论, 请访问仪表盘的“评论”页面。评论者头像来自Gravatar。

[全部 \(1\)](#) | [待审 \(0\)](#) | [已批准 \(1\)](#) | [垃圾 \(0\)](#) | [回收站 \(0\)](#)

在这里会显示博客的最新动向, 比如最近发布的文章、最近游客留下的评论等等。

WordPress 活动与新闻

WordPress活动及新闻

参加一场您附近的活动。 

 **WordCamp US** 2017年12月1日
Nashville, TN, USA

未来的 WordPress 4.8.1 和 cn.wordpress.org 计划

Dev Blog: WordPress 4.9 Release Candidate 3

WPTavern: iA Writer 5 for iOS Released, Web Collaboration Version Coming Soon

WPTavern: Watch the State of the Woo! After You Give WooCommerce Your Name and Email Address

聚会  | WordCamp  | 新闻 

这个栏目展示了 WordPress 官方的一些新闻和活动, 不过对于我们大多数人来说都没有任何用。所以这个选项一般都是隐藏的。

WordPress活动及新闻

参加一场您附近的活动。 

 **WordCamp US** 2017年12月1日
Nashville, TN, USA

单击右侧的“小三角”按钮, 就可以收缩 tab 了:



不过，用户也可以单击页面右上角的显示选项，取消该选项的勾选即可。

WordPress活动及新闻

调整仪表盘的布局

仪表盘的每个元素都是可以调整位置的，当将鼠标的光标移动到标题时，等待其变为移动符号时，拖动区块即可移动。

仪表盘

活动

最近发布
下午6:08 11月12日 世界, 您好!

近期评论

由一位WordPress评论者发表在《世界, 您好!》
嗨, 这是一条评论。要开始审核、编辑及删除评论, 请访问仪表盘的“评论”页面。评论者头像来自Gravatar。

全部 (1) | 待审 (0) | 已批准 (1) | 垃圾 (0) | 回收站 (0)

概览

1篇文章 1个页面
1条评论

WordPress 4.8.3, 使用Twenty Sixteen主题。

快速草稿

标题
在想些什么?

保存草稿

草稿

查看所有
测试4 2017年11月15日
测试4
测试3 2017年11月15日
测试3
测试2 2017年11月15日
测试2

文章管理

新建文章

文章是 WordPress 的最基本的单位，我们可以通过两种方式进入新建文章的页面，一个是通过菜单栏中的“文章——新建文章”进入：



或者通过顶部的快捷菜单进入。



页面布局调整

进入到新建文章页面后，会看到很多项目：

The screenshot shows the '撰写新文章' (Compose New Post) screen. On the left is a large text area with a placeholder '在此输入标题' (Enter title here). Above it is a toolbar with a '添加媒体' (Add Media) button and various styling options like bold, italic, and lists. To the right is a sidebar with sections for '发布' (Publish) and '形式' (Format). The '发布' section includes buttons for '保存草稿' (Save Draft), '预览' (Preview), and three publish options: '状态: 草稿 编辑' (Status: Draft Edit), '公开度: 公开 编辑' (Visibility: Public Edit), and '立即发布 编辑' (Publish Now Edit). The '形式' section contains a list of post types with radio buttons: 标准 (Standard) is selected, followed by 日志 (Log), 图像 (Image), 视频 (Video), 引语 (Quotation), 链接 (Link), 相册 (Album), 状态 (Status), 音频 (Audio), and 聊天 (Chat). At the bottom of the sidebar is a green '发布' (Publish) button.

在使用前，建议读者先调整这个页面的布局，调整为符合你自己习惯的布局。

单击上方的显示选项按钮，可以看到具体显示哪些模块，是否切换为单栏布局。

模块

形式 分类目录 标签 特色图片 摘要 发送Trackback 自定义栏目 讨论 别名 作者

布局

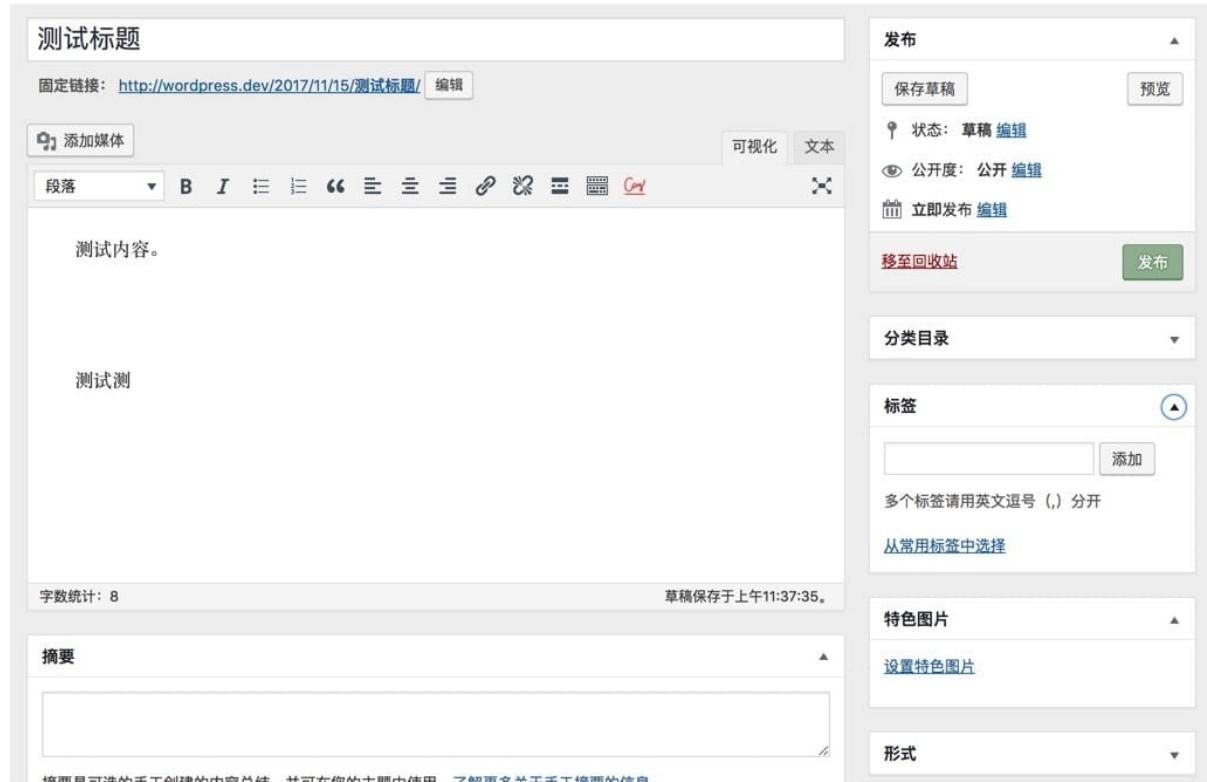
1列 2列

其他设置

启用全高度编辑器和免打扰功能。

一般来说，推荐显示 形式（需要主题支持）、分类目录、标签、特色图片（特色图片）、摘要。

我的布局如下，仅供参考：



其他内容可以根据自己的具体情况来设置。

下方的全高度编辑器其实是“全屏”功能，影响不到，主要是手动开启的。而后面的免打扰功能开启后，会在读者进行写作时进入免打扰模式，隐藏左侧的菜单和右侧的发布等连接。

The screenshot shows the WordPress dashboard with the "撰写新文章" (Compose New Article) screen selected. The left sidebar highlights the "文章" (Articles) section. The right sidebar, which is highlighted with a red box, contains the same publishing and form settings as the previous screenshot.

新建文章

标题栏

页面中的元素非常简单，上方的标题栏填写我们设置的文章标题。

固定链接

下方的固定链接则可以编辑，设置我们文章的链接名。如果你有自己的文章风格，不希望其为你生成默认的中文链接名，可以在这里手动设置。



添加媒体

WordPress 为我们提供了非常完善的多媒体资源管理能力。单击“添加媒体”按钮，就可以上传要添加的图片了。



你可以上传新的图片，或者使用媒体库中的图片，



上传后，选中你要插入的图片（支持选中多个）。单击“插入”按钮即可插入图片到文章。

在插入前可以设置图片的说明和替代文本，说明将会设置为 `caption` 标签。而替代文本则会设置为 `alt` 属性。

下方编辑器也非常好用，可以直接单击下方的按钮，选择对应的样式设置。



下方的摘要则会影响到文章列表和 RSS 列表的内容输出，你可以认真写一写摘要，并借此提高读者的打开率。

有的主题会支持不同形式的主题，可以设置不同形式的主题，以使用主题针对特定形式的设定。

有些主题还支持自定义栏目，还可以根据主题说明设置对应的自定义栏目。

名称	值
测试栏目	测试

添加自定义栏目：

名称	值
<input type="text"/>	<input type="text"/>

添加自定义栏目

自定义字段可用来添加**在您的主题中可用**的额外元数据到您的文章。

分类目录

单击左侧菜单栏中的“文章——分类目录”按钮，可以进入到分类目录的设置页面。

添加新分类目录

名称

 这将是它在站点上显示的名字。

别名

 “别名”是在URL中使用的别称，它可以令URL更美观。通常使用小写，只能包含字母，数字和连字符（-）。

父级分类目录
 无
 分类目录和标签不同，它可以有层级关系。您可以有一个“音乐”分类目录，在这个目录下可以有叫做“流行”和“古典”的子目录。

图像描述

 描述只会在一部分主题中显示。

添加新分类目录

可以新建一个分类目录。分类目录支持多级，这样可以很方便的分类文章。

				<input type="text"/>	<input type="button" value="搜索分类目录"/>	
<input type="button" value="批量操作"/>	<input type="button" value="应用"/>	1个项目				
<input type="checkbox"/> 名称	图像描述	别名	总数			
<input type="checkbox"/> 未分类		uncategorized	1			
编辑 快速编辑 查看						
<input type="checkbox"/> 名称	图像描述	别名	总数			
				<input type="button" value="批量操作"/>		
				<input type="button" value="应用"/>		

右侧则是分类目录的列表，可以单击编辑进入详细编辑页面编辑，也可以单击快速编辑修改分类名和分类别名。单击“查看”按钮，则会进入到前台的目录页面。

The screenshot shows the 'Categories' screen in WordPress. At the top right, it says '1个项目'. Below that is a table header with columns: '名称' (Name), '图像描述' (Image Description), '别名' (Alias), and '总数' (Total). A '快速编辑' (Quick Edit) section is shown below the table, containing fields for '名称' (Name) set to '未分类' (Uncategorized) and '别名' (Alias) set to 'uncategorized'. There are '取消' (Cancel) and '更新分类目录' (Update Category Directory) buttons. Below the table is another table header identical to the first. At the bottom left are '批量操作' (Bulk Operations) and '应用' (Apply) buttons, and at the bottom right is '1个项目'.

标签

除了分类目录，我们还可以通过标签对文章进行管理，标签的操作逻辑和分类目录基本一致，唯一不同的是标签不支持多级分类。

The screenshot shows the 'Add New Tag' form. It has three main input fields: '名称' (Name) with a placeholder '这将是它在站点上显示的名字。' (This will be its name on the site.), '别名' (Alias) with a placeholder '“别名”是在URL中使用的别称，它可以令URL更美观。通常使用小写，只能包含字母，数字和连字符 (-)。' (The alias is used in URLs to make them more aesthetically pleasing. It is usually lowercase and can only contain letters, numbers, and hyphens (-).), and '图像描述' (Image Description) with a note '描述只会在一部分主题中显示。' (Descriptions are only displayed in certain themes.). At the bottom is a green '添加新标签' (Add New Tag) button.

页面管理

页面部分比较简单，很多基础操作和文章管理是一致的，就不在赘述相同的部分了，这里只讲一下主要的不同点。

新建页面

新建页面中大体上和新建文章没有区别这里只说不同点，便是页面属性。



页面没有目录，但是可以设置页面的上级页面，通过一个主页面和若干子页面来管理内容，可以帮助我们更好的整理内容。

模板则是 WordPress 一个非常有用的功能。很多时候，我们需要的功能不止是新闻的展示，可能是一些其他的功能，如展示友情链接、展示日历、展示消息通知等等。

通过选择不同的模板，就可以实现不同功能页面的加载。

此功能需要主题的支持，后续主题开发课也会加入这部分内容。

多媒体资源管理

上传文件

多媒体资源的上传有两种方式，一种是通过新增文章、页面时上传，另一种是通过媒体库进行上传。

单击 **媒体 | 添加** 命令，可以看到上传的界面，选择要上传的内容就会自动上传了。你也可以把要上传的内容拖动到上传区域来进行上传。



管理文件

单击媒体 | 媒体库命令，会进入媒体库，可以在媒体库中管理附件：

文件	作者	上传至	日期
编辑用户 编辑用户.gif	admin	(尚未附加) 现在附加到文章或页面	2分钟前
测试 华东.gif	admin	测试标题 分离	2017-11-15

单击“附件”按钮，可以进入到附件的编辑页面进行管理：

固定链接: <http://wordpress.dev/编辑用户/>

Chip Bennett	—	订阅者
Emil Uzelac	—	订阅者

说明
这是图片的说明<caption></caption>

替代文本
这是图片的替代文本 alt="xxx"

图像描述
这是图片的描述，会展示在附件的页面。|

上传于: 2017年11月23日 @ 10:51
文件URL:
<http://wordpress.dev/wp-content/uploads/2017/11/%E7%BC%96%E7%BB%93%E5%8A%A0.gif>
文件名: 编辑用户.gif
文件类型: GIF
文件大小: 20 KB
分辨率: 895 x 119
永久删除 更新

可以为附件设置具体的信息，如果你的主题中有附件页，就可以看到具体的信息：



此外，我们还可以编辑图片，对图片进行简单的处理，在附件的编辑页面单击“编辑图像”按钮。



WordPress 支持对图片进行裁剪、旋转、镜像等功能。（我这里因为是gif，所以裁剪是灰的。）



编辑完成后，单击右侧的“更新”按钮即可将图片进行更新。

友情链接管理

在最新版本的 WordPress 中默认没有开启这个功能，不过可以通过安装 [Link Manager](#) 插件来开启这个功能。

单击左侧边栏的链接管理，可以进入到链接管理的界面。

首先，我们可以根据需要，创建链接所属目录，创建方式和创建文章目录相同，单击链接分类目录，进入到管理页面。

链接分类目录

添加链接分类目录

名称
这将是它在站点上显示的名字。

别名
“别名”是在URL中使用的别称，它可以令URL更美观。通常使用小写，只能包含字母，数字和连字符（-）。

图像描述
描述只会在一部分主题中显示。

创建完成目录后，我们可以创建具体的链接，单击左侧菜单栏中的添加按钮，进入到添加链接的页面。

名称
例如：好用的博客软件

Web地址
例子： `http://cn.wordpress.org/` ——不要忘了 `http://`

图像描述
通常，当访客将鼠标光标悬停在链接表链接的上方时，它会显示出来。根据主题的不同，也可能显示在链接下方。

保存
 将这个链接设为私密链接

必要设置包括名称和 Web 地址，将要添加的网站的名称和 Web 地址填入其中，即可添加链接了。

不过下方有很多其他的设置也值得知道。

图像描述**链接的描述**

通常，当访客将鼠标光标悬停在链接表链接的上方时，它会显示出来。根据主题的不同，也可能显示在链接下方。

分类目录

[所有分类目录](#) [最多使用](#)

友情链接

链接目录，方便归类

[+ 添加分类目录](#)

打开方式

`_blank` — 新窗口或新标签。

`_top` — 不包含框架的当前窗口或标签。

`_none` — 同一窗口或标签。

页面打开方式

为您的链接选择目标框架。

在下方应该会注意到有个 XFN (XML Friends NetWork)，通过 XFN 可以表明这个链接和你的远近亲疏关系，还是很有趣的。不过目前国内并没有针对此进行优化。

链接关系 (XFN)

关系 (rel) :

同一个人 我的另一个web地址

友情 偶有联系 熟人 朋友 无

网下接触 已见过面

职场关系 同事 同行

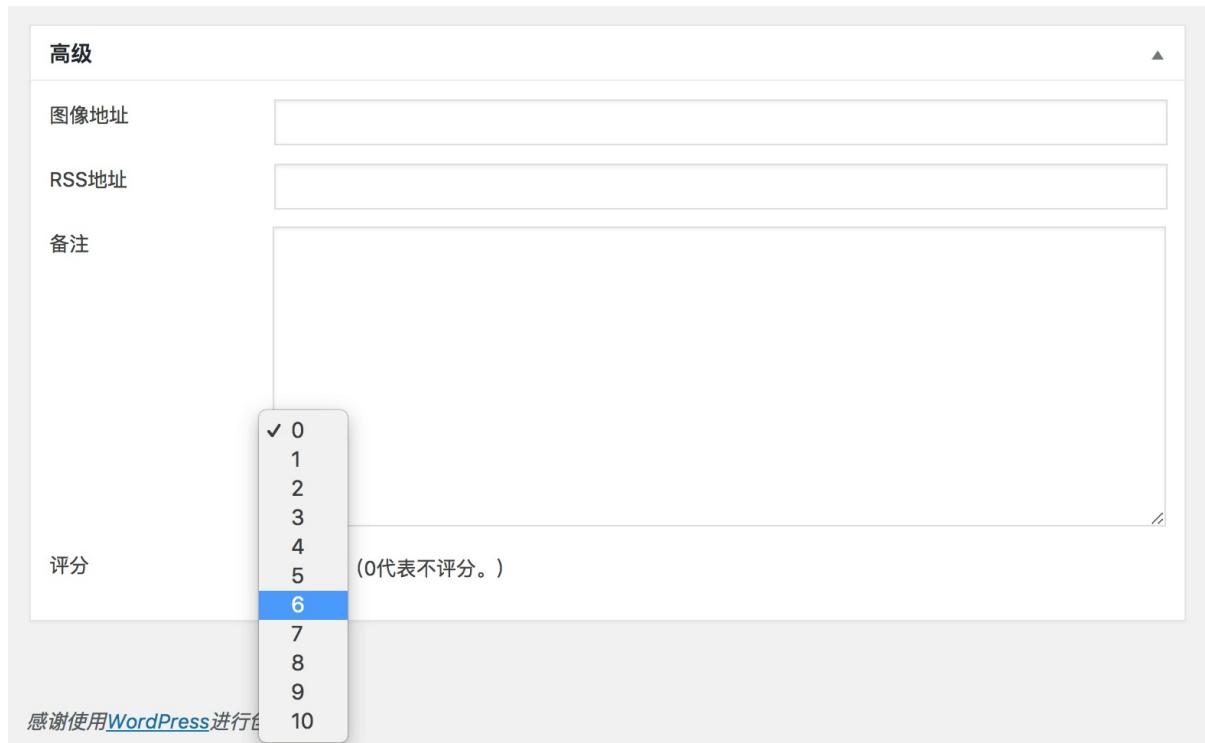
地理关系 同住 邻居 无

家庭关系 子女 亲戚 父母 兄弟姐妹 配偶 无

情感关系 灵感女神 迷恋 交往中 恋人

如果链接指向某个人，您可以用上面的表单指定您与此人的关系。如果您想了解更多，请查看：[XFN](#)。

此外，如果你觉得文字链接不好看，还可以选择使用图片链接，上传图片、设置备注、设置 RSS 地址等。



如果希望链接展示在前台，可以前往外观 | 小工具中添加。

挂件区域

将挂件加入此处来在侧边栏中显示。

链接

选择链接分类目录：

所有链接

排序依据：

链接评级

显示链接图像
 显示链接名
 显示链接描述
 显示链接评级

显示链接数：

[删除](#) [保存](#)

你可以在可视区域中添加一个链接小工具，并设置要显示的链接和具体的类型。

评论管理

单击左侧的评论管理 按钮，会进入到评论管理的页面。

全部 (33) 待审 (3) 已批准 (30) 垃圾 (0) 回收站 (0)			
批量操作		应用	全部评论类型
<input type="checkbox"/>	作者	评论	回复至 提交于
<input type="checkbox"/>	一位WordPress评论者 wordpress.org wapuu@wordpress.example	嗨，这是一条评论。 要开始审核、编辑及删除评论，请访问仪表盘的“评论”页面。 评论者头像来自Gravatar。	世界，您好！ 2017-11-12 下午6:08 查看文章 1
<input type="checkbox"/>	admin admin@admin.com 42.116.222.22	回复给tellyworthtest2。 fdafdsa	Page with comments 2014-12-10 上午1:56 查看页面 3 1
<input type="checkbox"/>	ken fitzgoo@sbcglobal.net 99.52.164.177	I want to learn how to make chinese eggrolls	Blog 2014-11-29 下午9:03 查看页面 0 1
<input type="checkbox"/>	auser auser@kopasoft.com 14.162.185.126	this is test comment from kopatheme	Template: Comments 2014-09-29 上午2:52 查看文章 19 1
<input type="checkbox"/>	John Doe example.org example@example.org 24.126.245.62	Having no content in the post should have no adverse effects on the layout or functionality.	Edge Case: No Content 2013-03-14 下午12:35 查看文章 1

评论默认有四种状态，待审核、已批准、垃圾评论和回收站评论。

所有未被确认为垃圾评论，但又从未留言过的评论都会放在待审核列表中（取决于在评论设置中的设置）。

可以在列表中修改一个评论的状态、回复、甚至是编辑它。

全部 (31) 待审 (3) 已批准 (28) 垃圾 (1) 回收站 (2)			
批量操作		应用	全部评论类型
<input type="checkbox"/>	作者	评论	回复至 提交于
<input type="checkbox"/>	一位WordPress评论者 wordpress.org wapuu@wordpress.example	嗨，这是一条评论。 要开始审核、编辑及删除评论，请访问仪表盘的“评论”页面。 评论者头像来自Gravatar。	世界，您好！ 2017-11-12 下午6:08 查看文章 1
<input type="checkbox"/>	admin admin@admin.com 42.116.222.22	回复给tellyworthtest2。 fdafdsa	Page with comments 2014-12-10 上午1:56 查看页面 3 1
<input type="checkbox"/>	ken fitzgoo@sbcglobal.net 99.52.164.177	I want to learn how to make chinese eggrolls	Blog 2014-11-29 下午9:03 查看页面 0 1
<input type="checkbox"/>	auser auser@kopasoft.com 14.162.185.126	this is test comment from kopatheme	Template: Comments 2014-09-29 上午2:52 查看文章 19 1

也可以单击后面的按钮，查看同一篇文章的其他评论。

评论			
全部 (30) 待审 (3) 已批准 (27) 垃圾 (2) 回收站 (3)		搜索评论	
批量操作	应用	全部评论类型	筛选
<input type="checkbox"/>	作者	评论	回复至 提交于
<input type="checkbox"/>	admin admin@admin.com 42.116.222.22	回复给tellyworthtest2。 fdafdsa	Page with comments 2014-12-10 上午1:56 查看页面 3 1
<input type="checkbox"/>	ken fitzgoo@sbcglobal.net 99.52.164.177	I want to learn how to make chinese eggrolls	Blog 2014-11-29 下午9:03 查看页面 0 1
<input type="checkbox"/>	auser auser@kopasoft.com 14.162.185.126	this is test comment from kopatheme	Template: Comments 2014-09-29 上午2:52 查看文章 19 1
<input type="checkbox"/>	Jane Doe example.org example@example.org 24.126.245.62	Thanks for all the comments, everyone!	Template: Comments 2013-03-14 上午11:30 查看文章 19 1

可以批量选择评论，并对他们进行处理。

评论			
全部 (30) 待审 (3) 已批准 (27) 垃圾 (2) 回收站 (3)		搜索评论	
批量操作	应用	全部评论类型	筛选
<input type="checkbox"/>	作者	评论	回复至 提交于
<input type="checkbox"/>	admin admin@admin.com 42.116.222.22	回复给tellyworthtest2。 fdafdsa 批准 回复 快速编辑 编辑 垃圾评论 移至回收站	Page with comments 2014-12-10 上午1:56 查看页面 3 1
<input type="checkbox"/>	ken fitzgoo@sbcglobal.net 99.52.164.177	I want to learn how to make chinese eggrolls	Blog 2014-11-29 下午9:03 查看页面 0 1
<input type="checkbox"/>	auser auser@kopasoft.com 14.162.185.126	this is test comment from kopatheme	Template: Comments 2014-09-29 上午2:52 查看文章 19 1
<input type="checkbox"/>	Jane Doe example.org example@example.org 24.126.245.62	Thanks for all the comments, everyone!	Template: Comments 2013-03-14 上午11:30 查看文章 19 1

上述就是我们常用的评论的操作。

此外，记得时常去清理下回收站，让数据库更干净。

The screenshot shows the WordPress dashboard with the 'Comments' section selected. At the top, there are navigation links: 全部 (30) | 待审 (0) | 已批准 (30) | 垃圾 (2) | 回收站 (3). Below these are buttons for 批量操作 (Bulk Actions), 应用 (Apply), 全部评论类型 (All Comment Types), 筛选 (Filter), and 清空回收站 (Empty Trash). A red arrow points from the '清空回收站' button to the trash bin icon in the interface below. The main area displays two comments:

作者	评论
admin wordpress.dev test@wp.dev 127.0.0.1	回复给一位WordPress。 你好
John Doe example.org example@example.org 24.126.245.62	Having no content in the post should have no adverse effects functionality.
Jane Doe example.org example@example.org 24.126.245.62	This comment should not be visible until the password is entered.

At the bottom, there are buttons for 作者 (Author) and 评论 (Comment), and a large red arrow points from the '清空回收站' button to the trash bin icon in the interface below.

主题管理

此部分会在后续的主题开发中进行讲解。

插件管理

此部分会在后续的插件开发中进行讲解。

用户管理

用户管理这里大体上和我们后面要讲的个人信息设置相同，具体的细节操作我不再重复，可以到第4课看 WordPress 的基本设置。

用户列表

点击左侧边栏、**用户——所有用户**，我们可以看到目前博客的所有注册用户、以及其所有文章。

The screenshot shows the WordPress User Management screen. At the top, there are buttons for '显示选项' (Display Options) and '帮助' (Help). Below that, a search bar labeled '搜索用户' (Search User) is present. The main area displays a table of users with columns: '用户名' (Username), '姓名' (Name), '电子邮件' (Email), '角色' (Role), and '文章' (Posts). The users listed are:

用户名	姓名	电子邮件	角色	文章
admin	HuanCheng Bai	test@wp.dev	管理员	1
Chip Bennett	—	—	订阅者	0
Emil Uzelac	—	—	订阅者	0
Lance Willett	—	—	订阅者	0
Theme Buster	—	—	订阅者	38

At the bottom of the table, there are buttons for '批量操作' (Bulk Actions), '应用' (Apply), '将角色变更为...' (Change Role), and '更改' (Edit). A note indicates '5个项目' (5 items).

除了自己这个用户无法删除外，你可以删除其他用户。

新建用户

单击左侧边栏 | 用户 | 添加用户命令，我们可以新建一个 WordPress 用户。

基本属性直接填写即可，这里关注一下密码和角色。

添加用户

新建用户，并将用户加入此站点。

用户名 (必填)	<input type="text"/>
电子邮件 (必填)	<input type="text"/>
名字	<input type="text"/>
姓氏	<input type="text"/>
站点	<input type="text"/>
密码	<input type="button" value="显示密码"/>
发送用户通知	<input checked="" type="checkbox"/> 向新用户发送有关账户详情的电子邮件。
角色	订阅者 <input type="button" value=""/>
<input type="button" value="添加用户"/>	




默认情况下，WordPress 会为用户生成一个极为复杂的密码，如果希望自己帮用户指定密码的话，可以单击“显示密码”按钮，在弹出的密码框中输入密码。

密码	<input type="text" value="gitchatPasswor"/>	<input type="button" value="隐藏"/>	<input type="button" value="取消"/>
发送用户通知	<input checked="" type="checkbox"/> 向新用户发送有关账户详情的电子邮件。		

而角色则会影响这个用户在站内的操作，具体的角色权限说明如下：

- 管理员：能够管理网站内的所有内容。
- 编辑：发表文章、编辑文章及编辑其他人的文章等。
- 作者：发布和编辑自己的文章。
- 投稿者：撰写和编辑自己的文章，但不能发布。
- 订阅者：查看评论/添加评论/查看文章等等。

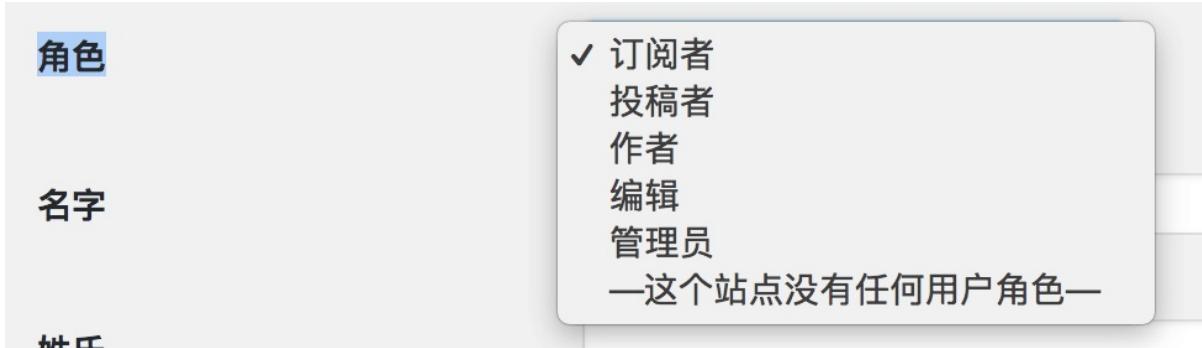
可以根据需要配置不同的角色给特定的用户。

修改用户身份

在用户列表中单击用户名下方的“编辑”按钮，可以进入到用户的个人编辑页面。

<input type="checkbox"/>	 Chip Bennett	—	订阅者
<input type="checkbox"/>	 Emil Uzelac	—	订阅者

在用户的个人编辑页面，我们可以找到角色选项，将其改为其他角色，再保存即可修改一个用户的角色。



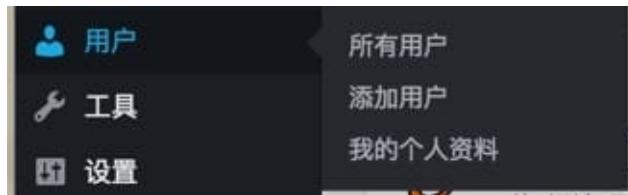
The screenshot shows the 'Edit User' page for 'Emil Uzelac'. On the left, there's a sidebar with '角色' (Roles) and '名字' (Name). A dropdown menu is open over the '角色' field, listing several user roles: '订阅者' (Subscriber), '投稿者' (Contributor), '作者' (Author), '编辑' (Editor), and '管理员' (Administrator). Below the list, a message reads: '—这个站点没有任何用户角色—' (This site has no user roles).

WordPress 基本设置

WordPress 个人设置

在使用 WordPress 前，可以先修改一下个人信息，让我们的个人信息更加丰满。

个人信息可以通过后台菜单中的“用户 | 我的个人资料”来查看：



或将鼠标放在右上角头像处，单击其中的编辑我的个人资料选项进入。



在这个页面，我们可以看到非常多的自定义设置。

可视化编辑器

可视化编辑器

撰写文章时不使用可视化编辑器

在上节课中，我们学习了如何使用 WordPress 发布文章、页面。默认情况下使用的是可视化编辑器，插入的图片都会显示出来。

当勾选这个选项时，编辑器将只显示代码编辑器。

大部分情况下，这个选项并没有什么用，不用勾选。

管理界面配色方案



这个选项可以让我们设置后台的不同色系，如默认是黑色系，可以通过选择不同的色系让后台颜色发生改变，让自己看着更舒服。

海洋色系



键盘快捷键

键盘快捷键

管理评论时启用键盘快捷键。 [更多信息](#)

键盘快捷键的选项可以帮助你开启编辑评论的 VIM 模式，无需移动鼠标就可以选择不同的评论进行管理。

具体的快捷键可以参看：[键盘快捷键](#)

工具栏

工具栏

在浏览站点时显示工具栏

当处在已登录的状态下访问站点时，会在页面顶部看到一个工具栏，通过工具栏，我们可以快速抵达后台的一些页面进行管理。如果你不喜欢在前台显示这个工具栏，可以在这里关闭它。



语言

语言

站点默认



一般情况下，用户的语言都是我们在安装时选择的语言。当然，可以根据自己的需要改成其他语言。不过默认只提供了你选择的语言和英文，如果需要更多语言，需要单独安装。



姓名

姓名

用户名	admin	用户名不可更改。
名字	<input type="text"/>	
姓氏	<input type="text"/>	
昵称 (必填)	admin	
公开显示为	admin	

姓名部分就涉及到我们的具体用户设置了。

我们可以在这里设置名字、姓氏和昵称，默认情况下，公开显示的是用户名。

在填写了昵称、名字和姓氏之后就可以选择在作者一栏显示什么内容了。除了昵称、用户名以外，还会对名字和姓氏进行组合，提供选择。

**联系信息**

联系信息

电子邮件 (必填)	<input type="text"/>
站点	http://wordpress.dev

这里比较关键的是电子邮件，电子邮件的设置将会影响到默认情况下头像的显示。此外，博客的评论会向你设置的这个邮箱发送电子邮件。

站点则是在你回复了其他用户评论时，头像默认连接的地址，其他人单击头像，将会跳转到这里设置的这个站点中去。

关于自己

关于您自己

个人说明

this is my bio.

分享关于您的一些信息。可能会被公开。

资料图片



您可以在[Gravatar](#)修改您的资料图片。

在这一部分，可以设置个人信息，这些个人信息将会展示在个人页面、或者是文章尾部的个人介绍中（此处需要主题的支持）。

头像则需要前往 <https://cn.gravatar.com/> 去更改，由于此站点账户使用的是 WordPress.com 的账户，而后者在国内无法访问，所以需要通过一些特殊渠道进行访问。

账户管理

账户管理

新密码

[生成密码](#)

会话

[登出其他设备](#)

您丢失了您的手机或是在公共电脑上登录着您的账户？您可以在其他设备上登出，只在此处保留登录状态。

在这一部分，可以修改当前用户的密码，还可以将在其他设备上的登录进行注销。

如果密码为他人所知，那么最好的办法就是先修改密码，再退出其他设备。

WordPress 常规设置



常规设置可以通过菜单进入。

站点标题和副标题

站点标题	WordPress Dev
副标题	又一个WordPress站点
用简洁的文字描述本站点。	

这两个选项一般来说，会展现在博客的顶部和网页中的标题部分，也就是 `title` 和 `description` 部分。

WordPress 地址和站点地址

WordPress地址 (URL)	<code>http://wordpress.dev</code>
站点地址 (URL)	<code>http://wordpress.dev</code>
如果您想让您的站点主页与WordPress安装目录不同，请在此输入地址。	

WordPress 允许将主站点和网站的路径设置不一样。这样，可以将后台放在一个独立的域名中，让网站更加的安全，无法被直接通过公开域名访问后台。

电子邮件地址

电子邮件地址	<code>best.tony@foxmail.com</code>
此地址被用作管理用途，如新用户通知。	

当站点有新用户注册、网站程序更新时，就会向这个邮件地址发送通知。一般来说，设置为和管理员相同的用户。

新注册用户设置

成员资格	<input type="checkbox"/> 任何人都可以注册
新用户默认角色	订阅者

在这里，我们可以设置是否允许新用户注册站点以及新用注册的默认模式。如果不打算做一个多人合写的博客，这里保持默认即可。

站点语言和时区

站点语言	简体中文
时区	*上海
选择与您在同一时区的城市或UTC时区偏移。	

可以在这里设置站点的默认语言，如果选择了一个新的语言，会自动下载相关语言的安装包并安装，用户的个人资料设置中也可以选择新的语言。

时区需要选择所在的时区，这个设置会影响到后续文章定时发布的设置。

时间和日期格式

日期格式

- 2017年11月14日 Y年n月j日
- 2017-11-14 Y-m-d
- 11/14/2017 m/d/Y
- 14/11/2017 d/m/Y
- 自定义: 2017年11月14日

时间格式

- 下午7:44 ag:i
- 7:44 下午 g:i A
- 19:44 H:i
- 自定义: 下午7:44

[日期和时间格式帮助.](#)

一星期开始于

在这里，可以设置合适的时间和日期的格式。如果选择自定义日期和时间格式，记得看下方的格式帮助。这里的格式需要满足 PHP 的日期格式。

WordPress 撰写设置

常规设置可以通过菜单进入。

撰写基础设置

默认文章分类目录	<input type="button" value="未分类"/>
默认文章形式	<input type="button" value="标准"/>

这两个选项设置了默认情况下的文章发布目录，当创建了一个新的文章却没有选择目录和文章形式时，会使用此处的设置。

默认目录可以根据你自己的选择设置。默认文章形式大部分主题都不支持，所以也无需理会。

通过电子邮件发布

通过电子邮件发布

通过电子邮件发布设定能让您通过发送您文章的内容到WordPress来发布。此信箱接收到的任何信件都将被发布，所以您使用：`bgWvbKQY`、`2zzF6CyU`、`douu3177`。

邮件服务器	<input type="text" value="mail.example.com"/>	端口	<input type="text" value="110"/>
登录名	<input type="text" value="login@example.com"/>		
密码	<input type="text" value="password"/>		
默认邮件发表分类目录	<input type="button" value="未分类"/>		

可以为 WordPress 设置一个电子邮件发布的功能。在此设置对应的邮件信息，然后将这个邮箱放在你自己的通讯录中，就可以在网络比较差的情况下，通过电子邮件来发布新的文章。

除此之外，我能够想到的用处便是 Kindle 的通过邮件分享功能了，可以将 Kindle 的标注分享到此处的邮件地址中去，自动将标注的内容发表在博客上，很适合做书摘内容。

更新服务

更新服务

在您发表新文章时，WordPress会自动通知站点更新服务。要获取更多资讯，请参见Codex上的[更新服务](#)。用换行分隔多个服务URL。

WordPress 更新服务主要是向下方的 RPC 服务商发送通知，告知已经发布了新的文章。目前在国内用处并不大，如果有兴趣，可以看看[这篇Wiki](#)。

WordPress 阅读设置

常规设置可以通过菜单进入。

首页显示

首页显示

您的最新文章

一个静态页面（在下方选择）

主页：

文章页：

一般作为博客而言，我们显示的都是文章列表，但是在建设企业网站、商城时，可能希望其显示的某个特殊页面，这时，我们就可以将其切换为某个静态页面（这里的静态页面指的是 WordPress 的 Page）。

输出设置

博客页面至多显示	<input type="text" value="10"/> 篇文章
Feed中显示最近	<input type="text" value="10"/> 个项目
对于feed中的每篇文章，显示	<input checked="" type="radio"/> 全文 <input type="radio"/> 摘要

在这里可以显示文章列表的数目。一般来说，当侧边栏特别长或者特别短时，我们会修改此处的设置，以使整个页面的更加美观。

显示全文还是摘要则取决于你是否希望读者回访网站，如果你希望他们能够回访网站，那么设置摘要会让他们“不得不”回到站点。

搜索引擎可见

对搜索引擎的可见性	<input type="checkbox"/> 建议搜索引擎不索引本站点 <small>搜索引擎将本着自觉自愿的原则对待WordPress提出的请求。并不是所有搜索引擎都会遵守这类请求。</small>
-----------	---

可以根据需要来设置这个站点是否对搜索引擎可见。例如，当你的站点是一个测试站点时，就可以设置为不可见，以免被搜索引擎爬到，出现在搜索引擎当中。

WordPress 讨论设置

常规设置可以通过菜单进入。

默认文章设置

默认文章设置	<input type="checkbox"/> 尝试通知文章中链接的博客 <input checked="" type="checkbox"/> 允许其他博客发送链接通知（pingback和trackback）到新文章 <input checked="" type="checkbox"/> 允许他人在新文章上发表评论 <small>(这些设置可被具体的文章设置所覆盖。)</small>
--------	--

前两项主要是 WordPress 会对引用的文章发起请求，告知对方我方引用了其文章，你可以根据你自己的需要开启，一般来说，保持默认即可。

第三项则是缺省的评论设置，如果没有为文章设置是否开启评论，则会根据此处的设置来决定。

注意，此处的设置不会影响已有文章是否开启评论。

其他评论设置

其他评论设置

评论作者必须填入姓名和电子邮件地址
 用户必须注册并登录才可以发表评论
 自动关闭发布 天后的文章上的评论功能
 启用评论嵌套, 最多嵌套 层
 分页显示评论, 每页显示 条评论, 默认显示 一页
在每个页面顶部显示

此处的评论设置主要影响文章尾部的具体配置。一般来说保持默认。

这里比较主要的是“启用嵌套评论”和“分页显示评论”。前者允许评论出现“楼中楼”形式的评论，而后者则允许将评论分页显示。具体的设置可以根据你的主题的使用说明选择。

这两项功能都需要主题支持。

电子邮件通知

发送电子邮件通知我

有人发表评论时
 有评论等待审核时

如果这两个选项钩上时，当你的博客发生对应事件时，就会发送电子邮件通知。

评论显示前设置

在评论显示之前

评论必须经人工批准。
 评论者先前须有评论通过了审核

这里是设置评论审核，一般来说保持默认。但是你也可以仿照微信公众号，设置“评论必须经人工审核”，从而使你的文章的评论更加的优质。

评论审核设置

评论审核

当某条评论包含超过 个超链接时，将其放入等待审核队列。（垃圾评论通常含有许多超链接。）

当评论的内容、姓名、URL、电邮或IP中包含以下文字，它将被设定为待审核。每行输入一个词或IP地址。它也会在单词内部进行比对，所以“press”将会匹配“WordPress”。

这个是添加评论的审核机制，当包含特定关键词时，放入待审核队列。

评论黑名单设置

评论黑名单

当评论的内容、姓名、URL、电邮或IP中包含以下文字，它将被移入回收站。每行输入一个词或IP地址。它也会在单词内部进行比对，所以“press”将会匹配“WordPress”。

这个是拉黑评论的审核机制，当包含了特定的关键词时，会直接放入黑名单。适合将一些有风险的关键词放入其中，如一些敏感词汇。

WordPress 媒体设置

媒体设置可以通过设置菜单进入。

图形大小

图像大小

下面列出来的尺寸决定插入媒体库内的图像之最大尺寸。以像素为单位。

缩略图大小	宽度 150	高度 150
<input checked="" type="checkbox"/> 总是裁剪缩略图到这个尺寸 (一般情况下, 缩略图应保持原始比例)		
中等大小	最大宽度 300	最大高度 300
大尺寸	最大宽度 1024	最大高度 1024

正常情况下，我们上传一张图片到 WordPress 后，WordPress 会自动帮我们生成一个缩略图、一张中等大小和一张大尺寸的图片。一般来说，当我们为主题设置了“特色图像”时，需要修改此次的设置。

如果你不希望生成对应的图片，可以将宽度和高度设置为0，就不生成对应的图片了。

文件上传

文件上传

默认上传路径 developer/php/wordpress/wp-content/uploads
缺省为 wp-content/uploads

文件的完整URL地址 可选配置， 默认留空。

以年—月目录形式组织上传内容

在主题的 functions.php 中加入如下代码，则会开启文件上传路径的设置。

```
if(get_option('upload_path')=='wp-content/uploads' || get_option('upload_path')==null) {
    update_option('upload_path',WP_CONTENT_DIR.'/uploads');
}
```

上面的两个选项一般来说是不用配置的，但是如果使用的是 VPS 主机，则可以考虑通过上述的设置项，将上传路径改到其他目录去，并修改 URL 地址为新的路径绑定的地址，从而实现对附件使用单独的域名，也方便后续的优化。

WordPress 固定连接设置

固定连接设置可以通过菜单进入。

连接风格设置

常用设置

- 朴素 <http://wordpress.dev/?p=123>
- 日期和名称型 <http://wordpress.dev/2017/11/14/sample-post/>
- 月份和名称型 <http://wordpress.dev/2017/11/sample-post/>
- 数字型 <http://wordpress.dev/archives/123>
- 文章名 <http://wordpress.dev/sample-post/>
- 自定义结构 <http://wordpress.dev/%year%/%monthnum%/%day%/%postname%/>

我们可以通过修改链接风格的设置，来使我们的文章生成不同样式的超链接。

根据需要，可以根据日期、名称、时间、数字等参数来进行链接的规划。

需要注意的是，网站上线后，就不要随便修改链接风格了，会影响到收录。

自定义前缀

可选

如果您喜欢，您可以在此给您的分类和标签自定义URL。比如，使用 `topics` 作为您的分类基础将会使您的分类链接变成 <http://wordpress.dev/topics/uncategorized/>。如果您留空此处，默认值将被使用。

分类目录前缀

标签前缀

默认情况下，WordPress 为我们生成的目录链接为 `域名/category/目录名/`，生成的标签链接为 `域名/tag/标签名/`。

我们可以在这里设置不同的前缀，来修改生成的目录。比如将目录前缀设置为 `dir`，生成的路径为 `域名/dir/目录名/`。

WordPress 常用插件使用说明（一）

本节课和下节课我们介绍一些常用的插件的使用，具体包括如下插件：Akismet、WP-Postviews、BackWPup、Google Authenticator、Crayon Syntax Highlighter、XML Sitemap & Google News feeds、WP-Optimize、mailgun、TablePress。

Akismet

插件地址：<https://wordpress.org/plugins/Akismet/>

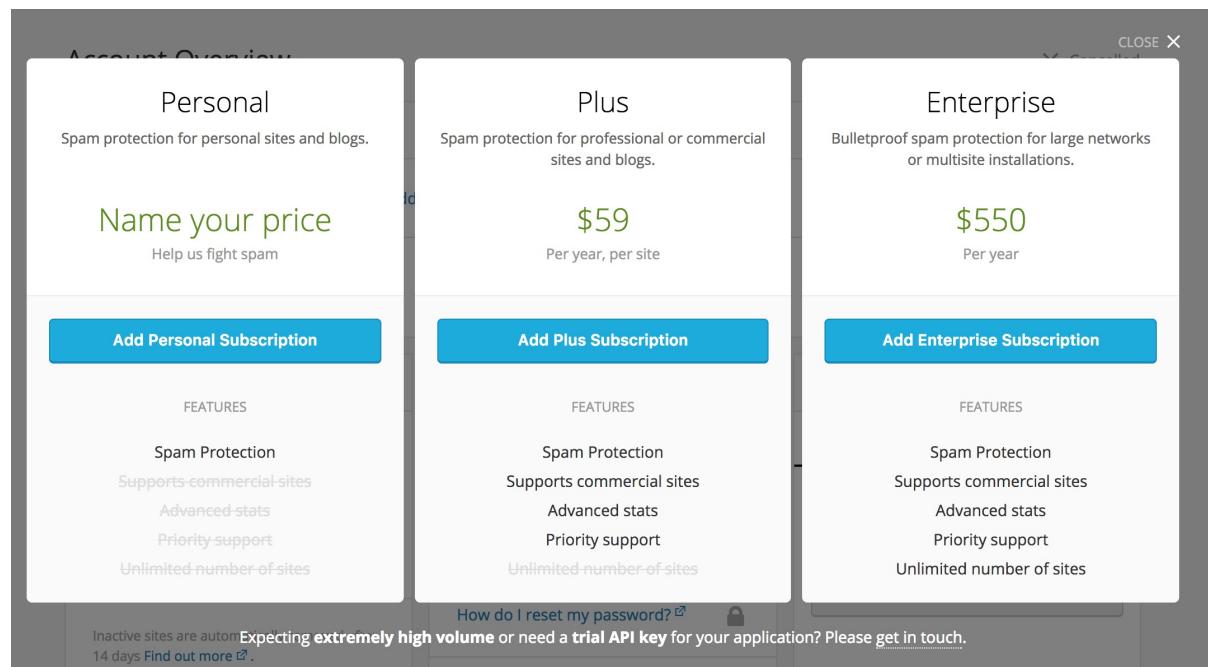
Akismet 是 WordPress 自带的反垃圾评论插件。如果博客的评论系统使用的是 WordPress 自己的评论系统，而没有对接诸如 Disqus 之类的第三方评论系统，则本插件一定要装。

Akismet 对接了其反垃圾服务器，任何评论将发送到其服务器，判断是否是垃圾评论，则将其放在垃圾邮件分组中，会节省大量的时间。

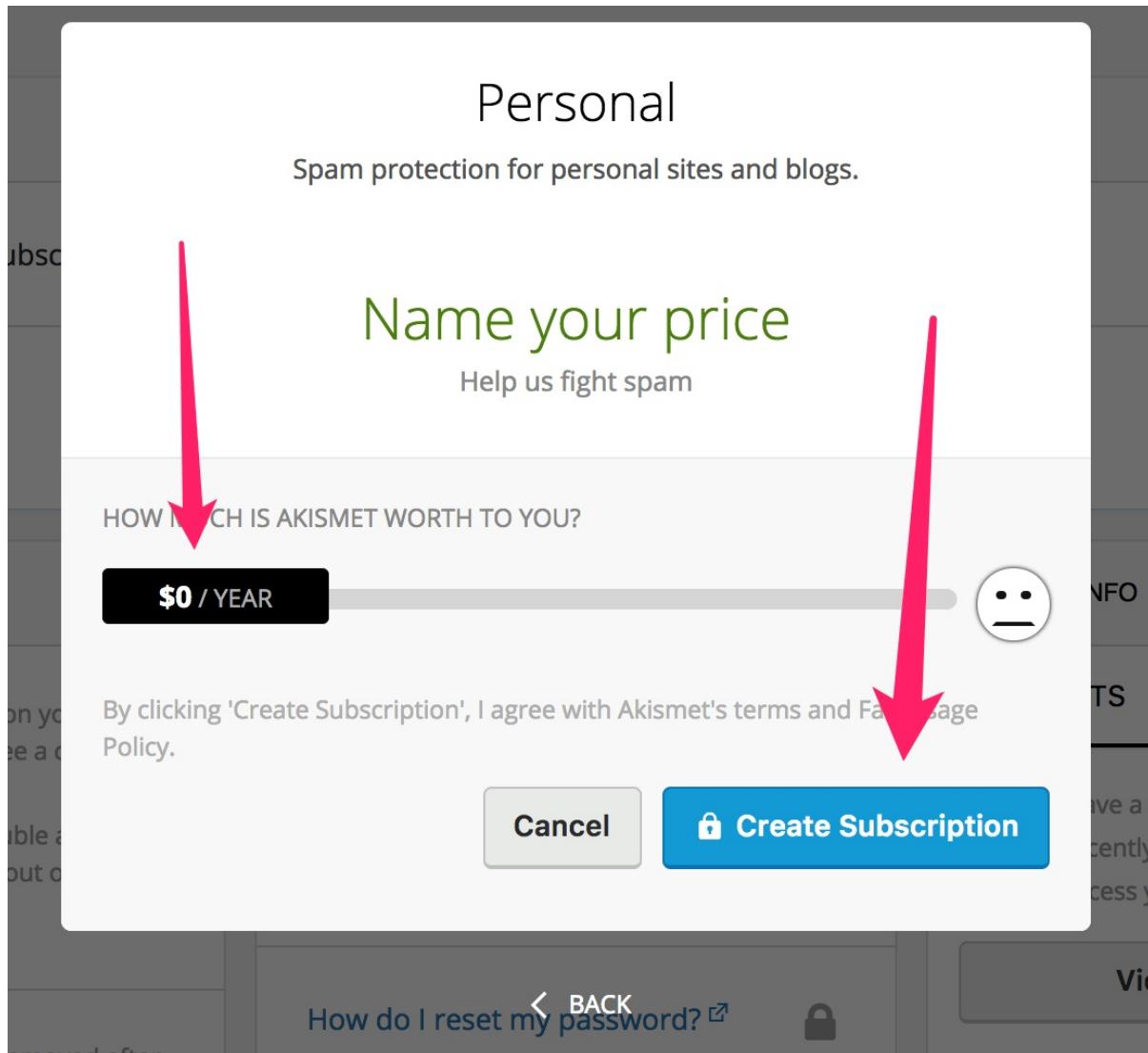
在启用该插件后，需要设置 Akismet key，而这个 Key 需要使用 WordPress.com 的账号登录 [Akismet.com](https://akismet.com) 获取。

创建 API Key

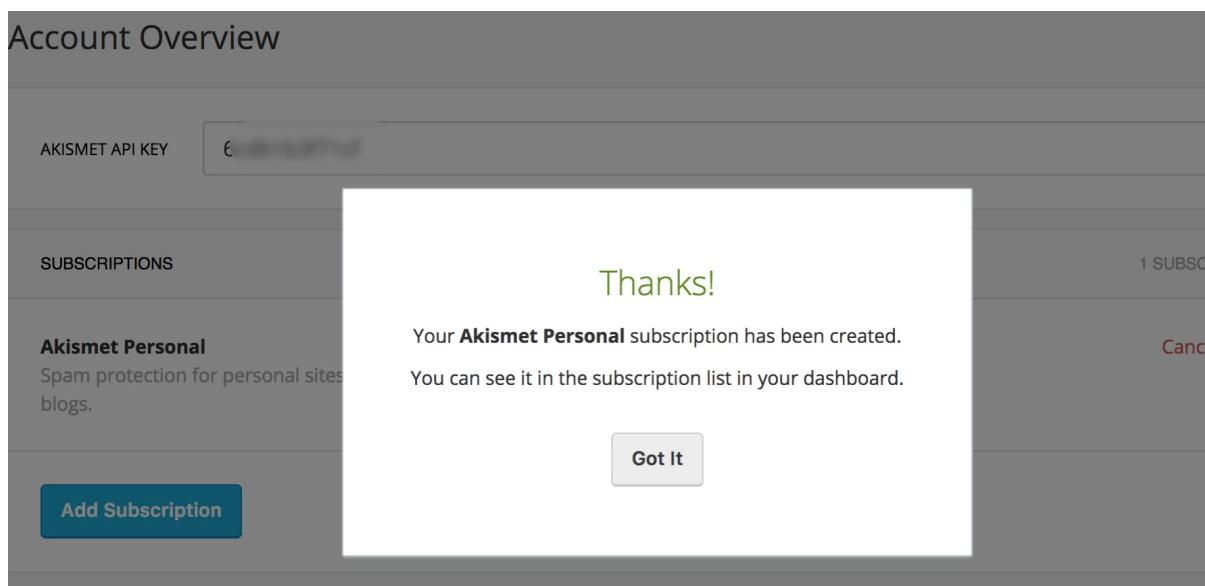
登录成功后，如果是第一次登录，会看到如下界面，单击其中的 **Add Personal Subscription** 按钮。



滑动下方的下拉条，拖动到 0 元每年，单击 **Create Subscription** 按钮。



稍等片刻，就会看到订阅创建成功。



将页面中的 AKISMET API KEY 复制出来：

The screenshot shows the 'Account Overview' page for the Akismet plugin. At the top, there's a green 'Active' status indicator. Below it, an 'AKISMET API KEY' field contains the value '6cd...'. Under the 'SUBSCRIPTIONS' section, there's one entry for 'Akismet Personal' which is 'Free'. It shows '0 active sites' and has links for 'Cancel', 'Edit', and 'Add Site'. A blue 'Add Subscription' button is located at the bottom left of this section.

粘贴到 WordPress 后台的 Akismet 设置页面，并连接即可。

或输入API密钥

已经有密钥了？在此输入。[\(什么是API密钥？\)](#)

This screenshot shows the 'Akismet' settings page in the WordPress admin. It features a large input field containing the API key '6cd...', a 'Connect API Key' button to its right, and a 'Cancel' link below the input field.

设置 Akismet

连接完成后，可以设置插件选项，根据自己的需要设置下面的设置项。如果你和我一样，担心会丢失评论，可以勾选“总是将垃圾放入垃圾目录以便审阅”选项。

This screenshot displays the detailed Akismet settings. It includes an 'API密钥' (API Key) field with a blurred value, a '评论' (Comments) section with a checked checkbox for '在评论作者名旁显示已获准的评论数量' (Show approved comment counts next to comment authors), a '严格度' (Severity) section with a selected radio button for '总是将垃圾放入垃圾目录以便审阅' (Always move spam to the trash bin for review), and a note stating that spam comments will be deleted after 15 days. At the bottom, there are '断开连接这个账户' (Disconnect this account) and '保存更改' (Save changes) buttons.

WP-Postviews

插件地址：<https://wordpress.org/plugins/WP-Postviews/>

WP-Postviews 会统计每篇文章的阅读量，可以通过一个特定的标签输出到前台，很多主题都集成了这个插件。如果你的主题没有集成，也可以自行为主题接入。

在需要展示的地方调用 `<?php the_views();?>` 即可。

插件设置

WP-Postview 安装完成后会新增一个设置页，可以通过修改设置来更好的使用该插件。

Post Views Options

Count Views From: Guests Only **是否统计已登陆用户**

Exclude Bot Views: No **是否排除机器人的浏览**

Views Template:

Allowed Variables: ● 可用标签

- %VIEW_COUNT%
- %VIEW_COUNT_ROUNDED%

%VIEW_COUNT% views **浏览量标签**

Restore Default Template

Most Viewed Template:

Allowed Variables: ● 可用标签

- %VIEW_COUNT%
- %VIEW_COUNT_ROUNDED%
- %POST_TITLE%
- %POST_DATE%
- %POST_TIME%
- %POST_EXCERPT%
- %POST_CONTENT%
- %POST_URL%
- %POST_THUMBNAIL%
- %POST_CATEGORY_ID%

%POST_TITLE% - %VIEW_COUNT% views

侧边栏小工具样式模板

一般来说，我们要修改 Views Template，比如改为 %VIEW_COUNT%人看过。

下方的 Most Viewed Template 则是设置小工具的样式，WP-Postviews 默认注册了一个小工具，我们可以在主题的小工具中设置。

挂件区域

将挂件加入此处来在侧边栏中显示。

Views

Title:

Statistics Type:

Include Views From:

No. Of Records To Show:

Maximum Post Title Length (Characters):
0 to disable.

Category IDs: *

Separate multiple categories with commas.

* If you are not using any category statistics, you can ignore it.

[删除](#) [保存](#)

然后回到主页，就可以在侧边栏中看到了。

WordPress Dev

又一个WordPress站点

首页

示例页面

最多人浏览

- N/A

如果你的主题有特殊的样式，根据在设置页面进行修改即可。此外，还可以设置是否在特定的页面展示阅读量。

Display Options

These options specify where the view counts should be displayed and to whom. By default view counts will be displayed to all visitors. Note that the theme files must contain a call to `the_views()` in order for any view count to be displayed.

Home Page: 展示给所有人

Single Posts: 展示给登陆用户

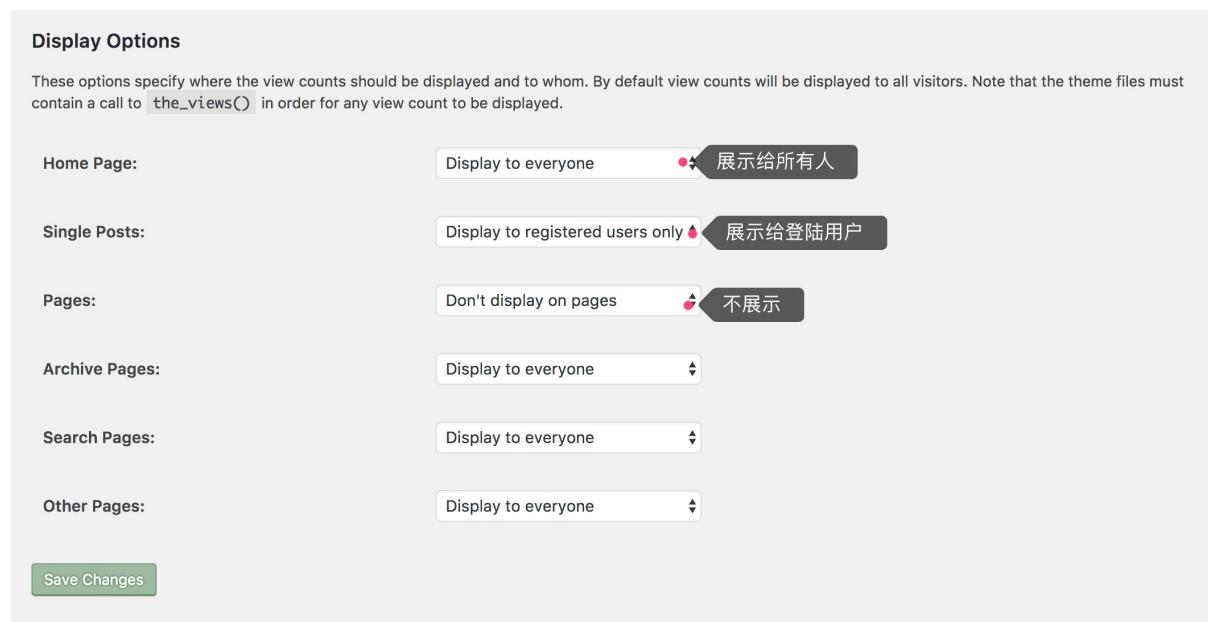
Pages: 不展示

Archive Pages: 展示给所有人

Search Pages: 展示给所有人

Other Pages: 展示给所有人

Save Changes



BackWPup

插件地址：<https://wordpress.org/plugins/BackWPup/>。

BackWPup 是一款非常好用的备份插件，我们可以使用它来备份网站数据，以防数据出现问题后，无法恢复。

创建备份任务

首先，我们要创建一个备份任务，点击左侧菜单**BackWPup | Add New Job** 命令，可以设置各种选项：

BackWPup > Job: New Job

General Schedule DB Backup Files Plugins

Job Name

Please name this job.  任务名, 如月度附件备份

Job Tasks

This job is a ...

● 备份内容

Database backup • **数据库备份**

File backup

WordPress XML export

Installed plugins list

Check database tables

Backup File Creation

Archive name

Note: In order for backup file tracking to work, %hash% must be included anywhere in the

Preview: 2017-11-22_20-29-04_%hash%.zip

这一部分比较核心的是 Job Destination, 就是备份目标位置。

Archive Format

- Zip
- Tar
- Tar GZip
- Tar BZip2

● 备份文件格式

Job Destination

Where should your backup file be stored?

- Backup to Folder
- Backup sent via email
- Backup to FTP
- Backup to Dropbox
- Backup to an S3 Service
- Backup to Microsoft Azure (Blob)
- Backup to Rackspace Cloud Files
- Backup to SugarSync

● 备份到哪里

Log Files ● 备份日志设置

Send log to email address

Leave empty to not have log sent. Or separate with , for more than one.

Email FROM field

以 FTP 来为大家演示。

设置完上面的选项后，单击下方的 save changes 按钮，保存设置。

保存刷新后，会发现顶部的 tab 多了一个 To:FTP，单击进去可以设置 FTP 服务器的具体配置信息。

比如，使用又拍云作为备份的服务器，具体配置如下：

FTP server and login

FTP server	v0.ftp.upyun.com	Port:	21
Username	gitchat/gitchat		
Password	*****		

Backup settings

Folder to store files in	wordpress-dev/	
File Deletion	15	Number of files to keep in folder.

Warning: Files belonging to this job are now tracked. Old backup archives which are untracked will not be automatically deleted.

配置完成后，单击 save changes 按钮，保存设置。

BackWPup > Job: test job

Changes for job *test job* saved. [Jobs overview](#) | [Run now](#)

[General](#) [Schedule](#) [DB Backup](#) [Files](#) [Plugins](#) [To: FTP](#)

FTP server and login

单击 Run now 按钮，就可以开始运行备份任务，当你看到这样的界面，则说明备份成功。

Warnings: 0 Errors: 0

100%
Job completed
100%
Job done in 12 seconds.

[Display working log](#) [Close](#)

可以登录到 FTP 确认一下。



设置自动备份

打开 [EasyCron.com](#), 登录后, 可以看到这样的界面。

Total 60,000 EPDs / 0 Occupied / 60,000 Remaining

单击菜单栏中的 API 选项:

Cron Job API

API Document

If you want to use our cron job service via REST API, please check out more at our [API Document](#) page.

API Token

You may use the below API token in your API requests. Please keep the token private.

a8837814ca58c [REDACTED]

[Regenerate a new API token](#)

复制页面中的 API token，回到 WordPress 后台，单击左侧菜单 **BackWPup | Settings** 名，选择 API。

将我们刚刚复制的 token 粘贴在此，并勾选下方的 *If you check this box, a cron job will be created on EasyCron that all 5 Minutes calls the WordPress cron* 选项：

EasyCron

Here you can setup your [EasyCron.com API key](#) to use this service.

Api key: *****

Trigger WordPress Cron: If you check this box, a cron job will be created on EasyCron that all 5 Minutes calls the WordPress cron.

Save Changes **Reset all settings to default**

然后单击 Save Changes 按钮，这样我们就成功接入了 EasyCron 作为定时任务。

这时如果你去看 EasyCron 后台，会看到这样一条记录：

ID	Name	Expression	URL to Call	EPDs	TS	TF	CF	Email Me	Logs & Predictions	Status	Actions
455245	WordPress on http://wordpress.dev	*/5 * * * *	http://wordpress.dev/wp-cron.php?doing_wp_cron	288	0	0	0	never	don't log	ON	

Showing 1 - 1 of 1

Rows per page: 25

这就是插件自动生成的任务。

单击左侧菜单 **BackWPup | Jobs** 命令，找到刚刚创建的任务，单击 Edit 按钮，进入后，切换到 Schedule Tab，将 Start Job 改为 `with EasyCron.com`。

下方就会出现具体的备份频次配置，根据你的需要配置，并保存即可。

Type	Hour	Minute
monthly	on 1	3
weekly	Sunday	3
daily	3	0
hourly		0

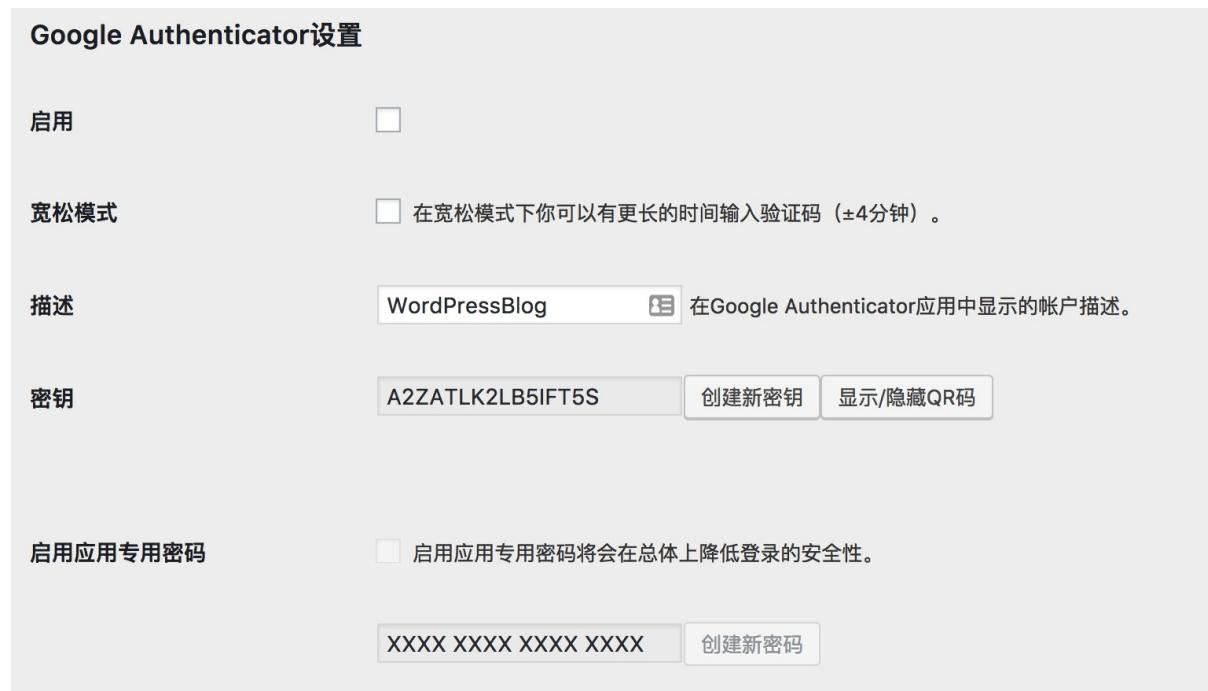
Google Authenticator

插件地址：<https://wordpress.org/plugins/google-authenticator/>

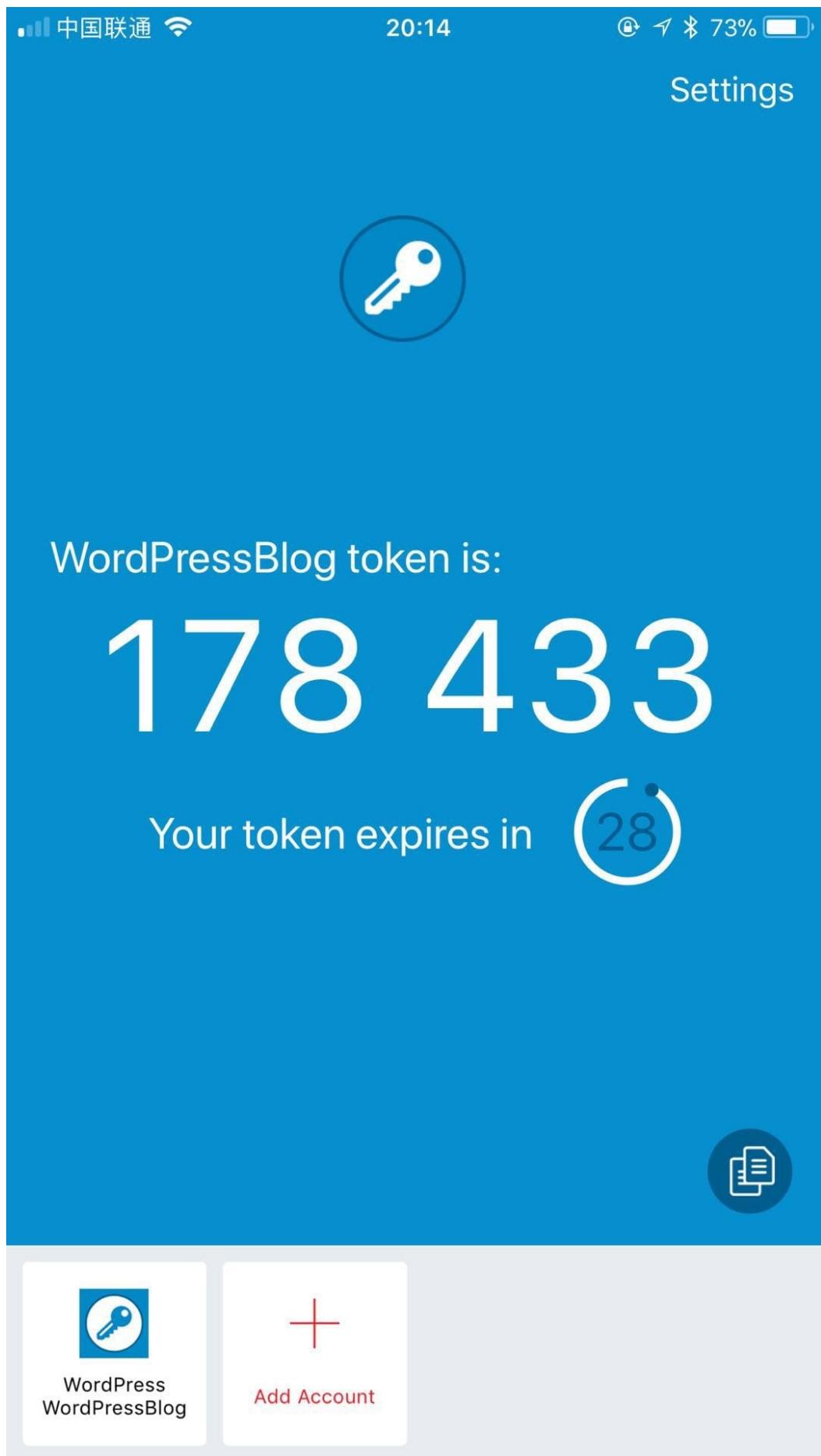
Google Authenticator 可以帮助我们在 WordPress 上加入两步验证的插件，在 WordPress 中启用插件以后，再次登录就需要输入验证码了，为 WordPress 站点增加一重保障。

设置

启用插件后，单击用户中的我的个人资料选项，会看到如下设置项：



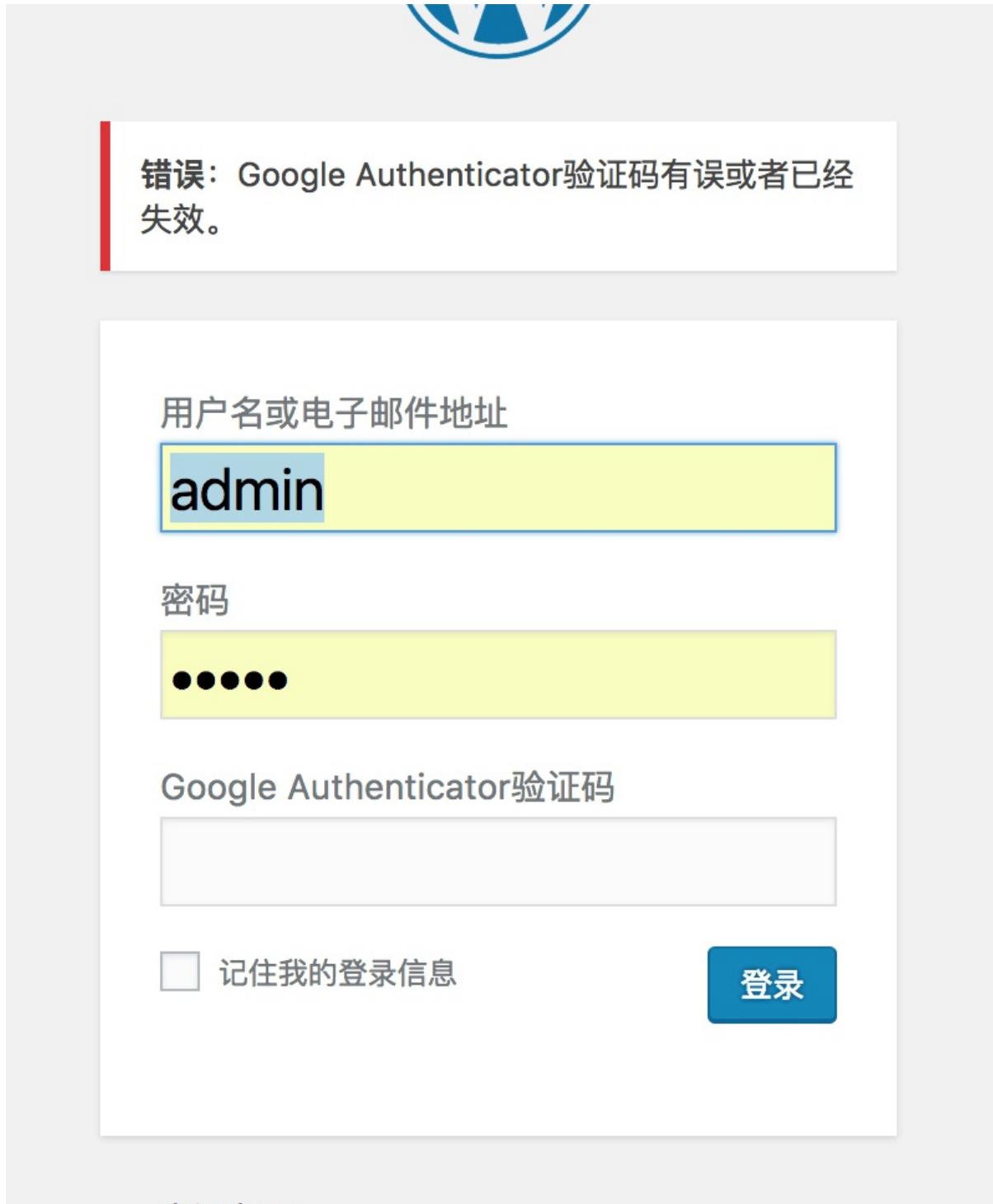
首先，修改下方的描述，设置为你的博客的名字，或者是你自己自定义的名字，然后单击密钥右侧的显示/隐藏 QR 码，会看到下方出现一个二维码。使用安装好的两步验证软件（我使用的是 Authy）扫描这个二维码，然后在应用中确认保存。保存完成后，勾选上方的启用，并更新个人资料。



这样，我们就配置好了两步验证。

测试登录

退出账号，重新登录，会发现多了一个密码框，再登录就需要输入你的两步验证软件中的动态口令了。如果没有填写，则会提醒验证码有误。



输入正确的验证码，就可以正常登录了。

WordPress 常用插件使用说明 (二)

Crayon Syntax Highlighter

插件主页：<https://cn.wordpress.org/plugins/crayon-syntax-highlighter/>。

大家都是写代码的，自然是免不了在博客中粘贴一些代码，WordPress 中最出名、功能最强大的代码高亮工具莫过于 Crayon Syntax Highlighter。

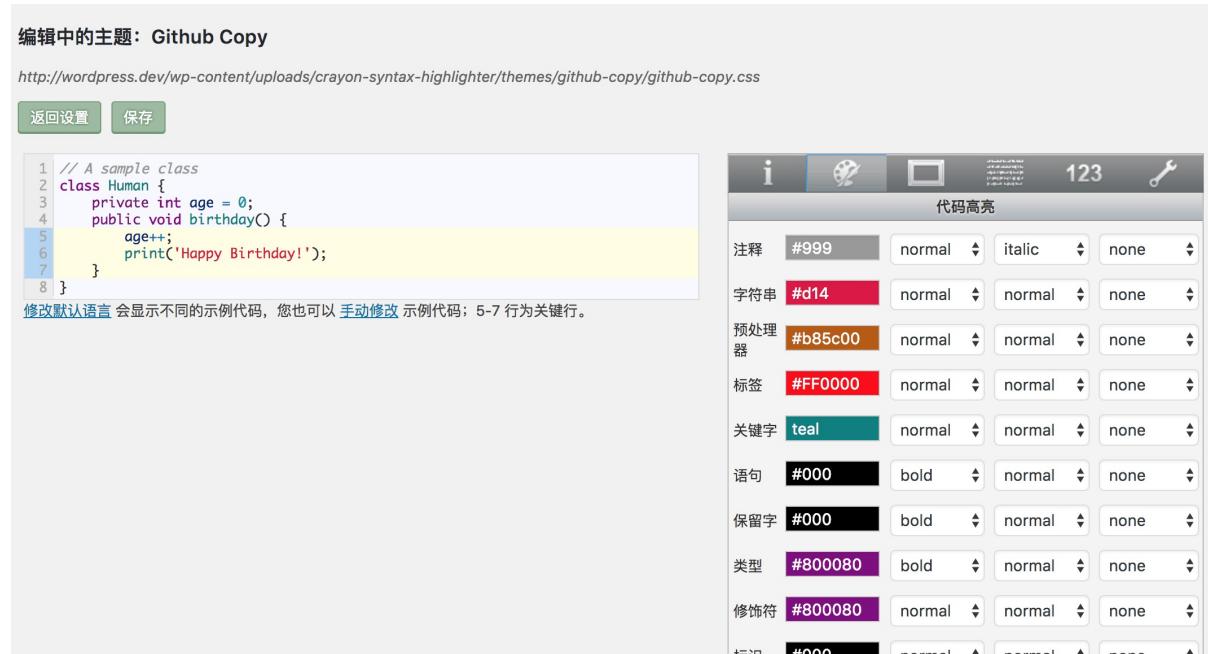
设置插件

启用插件后，单击设置 | Crayon 命令，进入到插件配置页面。

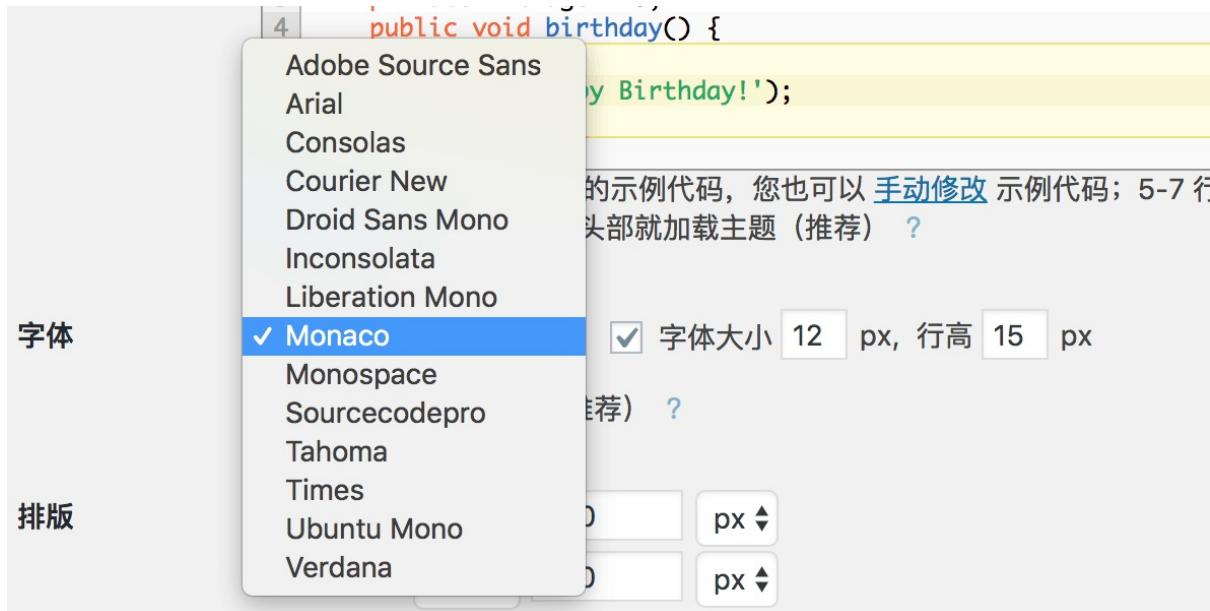
首先是设置主题，Crayon 拥有多种不同的主题，可以选择你喜欢的主题。



如果所有的主题都不满意，也可以自己设计一个。选择一个最满意的主题，勾选复制，生成一个新的主题，然后就可以编辑了。



Crayon 的主题编辑器非常的好用，如果不满意其他的主题，不妨自己动手做一个。主题默认提供了很多不同的字体，可以根据自己的喜好选择。



排版则是设置了主题框在文章中的展示样式，一般来说不做修改，保持默认即可。

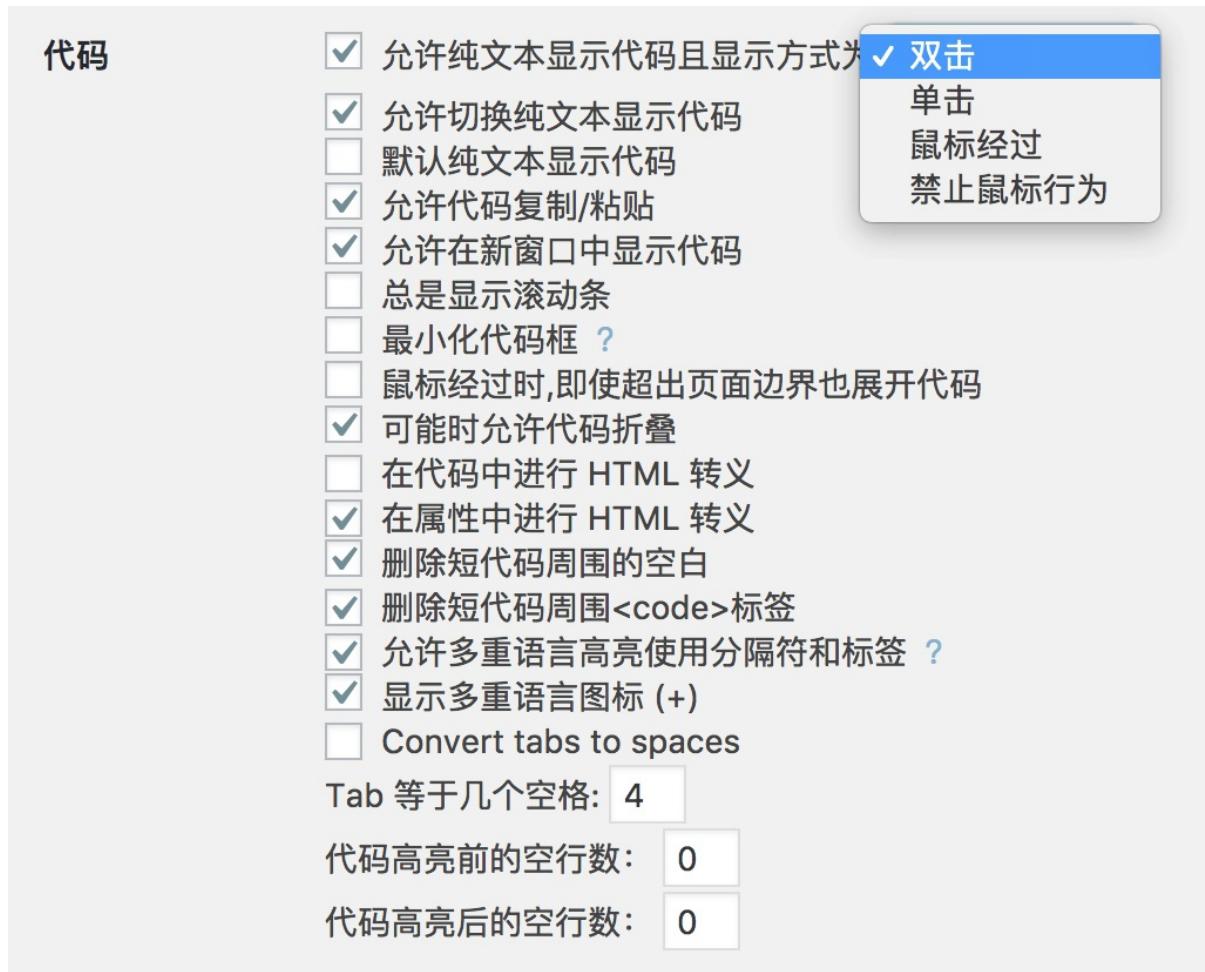


工具栏设置基本不用改，如果希望更方便复制，则建议工具栏选择始终显示，语言也可以选择使用显示。



行设置则根据你的需要选择。

在代码设置中，建议读者将显示方式设置为 禁止鼠标行为，因为默认的设置在真正复制时会出现一些问题。



标签设置中可以勾选上捕获 `<code>` 标签为插件所用行内标签 选项, 这样使用 `<code>` 标签的标记的代码也会加入代码高亮。

标签

- 捕获行内标签 ?
- 行内标签自动换行 ?
- 捕获 <code> 标签为插件所用 行内标签 ?
- 捕获 `反引号` 为 <code> 标签 ?
- 捕获 <pre> 标签为插件所用 ?

Using this markup for Mini Tags and Inline tags is now deprecated! Use the [Tag Editor](#) instead and convert legacy tags.

- 使用迷你标签 (如[php][/php]) ?
- 捕获如{php}{/php}形式的行内标签 ?
- 启用 [plain][/plain] 标签 ?

语言

当没指定语言, 则默认语言为: Python

已支持 65 种语言, 加载完成.

[显示全部语言](#)

文件

当 URL 为相对路径时, 使用的绝对路径为:

http://wordpress.dev/

置于相对路径之前

文章

[显示已使用代码高亮的文章](#) [刷新](#) ?

语言则根据最常用的语言选择即可。

其它

清理用于存储远程代码的缓存频率: 每天 立刻清理

- 按需加载插件的 CSS 与 JavaScript ?
- 不将可能包含Wordpress主循环的页面模板加入队列 ?
- 允许在评论中使用
- 文章摘要不启用代码高亮
- 只从Wordpress的主查询中加载Crayon
- 使用触屏设备时禁止鼠标行为 (如鼠标经过)
- 禁止动画效果
- 禁止运行时间统计

Disable for posts before: 年 /月/日

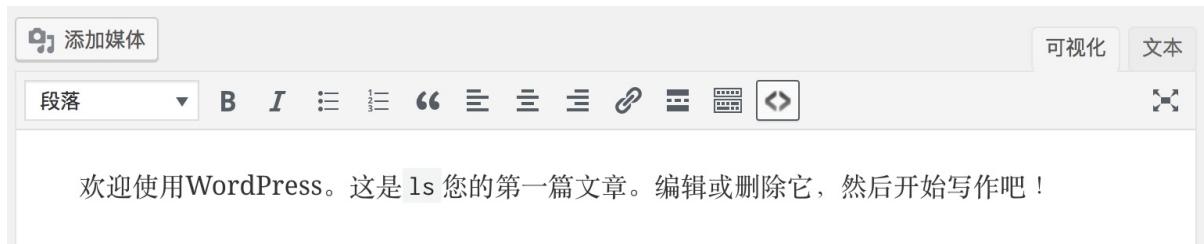
Load scripts in the page footer using wp_footer() to improve loading performance.

最后的其他设置中, 一定要勾选按需加载插件的 CSS 与 JavaScript选项, 不然会在所有页面加载, 把网站的速度拖慢。

设置完成后, 保存即可。

在文章中添加代码

在文章中添加代码时, 可以单击菜单栏中的代码图标:



在弹出的输入框中输入代码，并单击“插入”按钮。

如果希望修改具体的设置，可以修改下方设置的具体内容。



回到前台，我们就可以看到渲染的效果了。

欢迎使用WordPress。这是 1s 您的第一篇文章。编辑或删除它，然后开始写作吧！

```
这是我的演示代码
1 print("Hello World")
```

XML Sitemap & Google News feeds

插件主页：<https://cn.wordpress.org/plugins/xml-sitemap-feed/>

XML Sitemap 能够帮助我们生成合规的 SiteMap，帮助爬虫更好的访问我们的网站。

插件设置

插件启用后，单击设置|阅读命令，可以看到插件的具体配置。

其中 Google News Feeds 对我们大多数人都没用，就不再介绍了，这里只说一下 XML Sitemap。

默认情况下，插件会帮我们打开 Sitemap 的生成。



单击“查看”按钮，就可以看到生成的 XML 文件，

XML Sitemap Feed – Index

This is the XML Sitemap Index to aid search engines like [Google](#), [Bing](#), [Yahoo!](#) and [Ask](#) index **this index file** as your sitemap. Read more about XML sitemaps on [Sitemaps.org](#).

#	XML Sitemap	Last Changed
1	http://wordpress.dev/sitemap-home.xml	2017-11-12 10:08
2	http://wordpress.dev/sitemap-posttype-post.xml	2017-11-22 13:20
3	http://wordpress.dev/sitemap-posttype-page.xml	2017-11-12 10:08

[SITEMAP](#) [XML](#) generated by [XML Sitemap & Google News Feeds](#) running on [WordPress](#).

单击不同的 XML 就可以看到对应的 sitemap 了。

默认情况下，只会生成文章和页面的地址。

如果网站有针对目录和标签进行美化，还可以勾选包含分类中的 目录 和标签。

WP-Optimize

此部分在第 11 课的优化部分进行讲解，具体请看第 11 课的内容。

MailGun

插件地址：<https://wordpress.org/plugins/MailGun/>

MailGun 是一个非常出名的邮件服务提供商，和国内的 SendCloud 做的是同类项产品，不过 SendCloud 的官方插件已经很久没有更新，所以选择使用 MailGun 的官方插件来作为替代品。

插件设置

在安装完成后，需要去设置账户：

Mailgun

A [Mailgun](#) account is required to use this plugin and the Mailgun service.

If you need to register for an account, you can do so at <http://www.mailgun.com/>.

Configuration

Use HTTP API	<input style="border: 1px solid #ccc; padding: 2px 10px; border-radius: 5px; width: 100px; height: 20px; vertical-align: middle;" type="button" value="Yes"/>
Set this to "No" if your server cannot make outbound HTTP connections or if enabling this plugin to use SMTP. Default "Yes".	
Mailgun Domain Name	<input style="width: 200px; height: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px 5px; vertical-align: middle;" type="text" value="samples.mailgun.org"/> <input style="border: 1px solid #ccc; padding: 2px 5px; border-radius: 5px; width: 20px; height: 20px; vertical-align: middle;" type="button" value="Edit"/>
Your Mailgun Domain Name.	
API Key	<input style="width: 200px; height: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px 5px; vertical-align: middle;" type="text" value="key-3ax6xnjp29jd6fds4gc373sgvjxteol0"/>
Your Mailgun API key, that starts with and includes "key-". Only valid for use with this plugin.	
Click Tracking	<input style="border: 1px solid #ccc; padding: 2px 10px; border-radius: 5px; width: 100px; height: 20px; vertical-align: middle;" type="button" value="HTML Only"/>
If enabled, Mailgun will track links. Click Tracking Documentation	
Open Tracking	<input style="border: 1px solid #ccc; padding: 2px 10px; border-radius: 5px; width: 100px; height: 20px; vertical-align: middle;" type="button" value="Yes"/>
If enabled, Mailgun will track links. Open Tracking Documentation	

可以单击[mailgun](#), 前往注册账户。

注册成功登录后, MailGun 会默认创建一个发件域名。

The screenshot shows the Mailgun dashboard interface. At the top, there are three buttons: 'Add New Domain' (dark grey), 'Authorized Recipients' (light grey), and a search bar 'Search 4 domains' with a magnifying glass icon. Below these are several status indicators: 'State' (Active), 'Domain Name' (sandbox1b885f98c39a4c32be8690e45b2 815ed.mailgun.org), 'Outgoing (30d)' (0), 'Bounced' (0%), 'Complaints' (0%), and 'Posts via Routes' (0). A red dashed line separates this from the rest of the page.

建议你使用自己的域名。

我们使用这个沙箱域名来设置我们的邮件, 选择域名, 进入详情页面:

DOMAIN
sandbox1b[REDACTED]e45b2815ed.mailgun.org

Domain Information

State	Active
IP Address	[REDACTED] • Manage IPs
SMTP Hostname	smtp.mailgun.org
Default SMTP Login	postmaster@[REDACTED]be8690e45b2815ed.mailgun.org
API Base URL	https://api.mailgun.net/v3/sandbox[REDACTED]8690e45b2815ed.mailgun.org
Default Password	e2b5eb[REDACTED].5e506b • Manage SMTP credentials
API Key	key-[REDACTED]0cff561ac1e

我们复制这里的 Domain 和 API Key，填写到 WordPress 插件页面的对应位置。

Use HTTP API **Yes** Set this to "No" if your server cannot make outbound HTTP connections or if emails are not being delivered. "No" will cause this plugin to use SMTP. Default "Yes".

Mailgun Domain Name **sandbox1b[REDACTED]e45b2815ed.mailgun.org** Your Mailgun Domain Name.

API Key **key-298[REDACTED]ac1e** Your Mailgun API key, that starts with and includes "key-". Only valid for use with the API.

上方的 USE HTTP API 保持为默认的 YES（如果你的虚拟主机不支持对外发送 HTTP 请求，可以设置为 NO，然后通过 SMTP 发送邮件）。

下方的设置可以参考我这里的图片注释来确定是否开启。

Click Tracking **HTML Only** If enabled, Mailgun will track links. [Click Tracking Documentation](#) 点击跟踪，会经过mailgun对连接进行中转，方便统计多少用户是通过邮件进入的

Open Tracking **Yes** If enabled, HTML messages will include an open tracking beacon. [Open Tracking Documentation](#) 开启追踪，通过开启要求数据，统计用户打开率

From Address **no-reply@wordpress.dev** **发件人地址** The part of the sender information ("Excited User <user@samples.mailgun.org>"). This address will appear as the 'From' address on sent mail. It is recommended that the @mydomain portion matches your Mailgun sending domain.

From Name **GitChat** **发件人姓名** The "User Name" part of the sender information ("Excited User <user@samples.mailgun.org>").

Override "From" Details **No** 是否使用上方的设置来发送邮件 If enabled, all emails will be sent with the above "From Name" and "From Address", regardless of values set by other plugins. Useful for cases where other plugins don't play nice with our "From Name" / "From Address" setting.

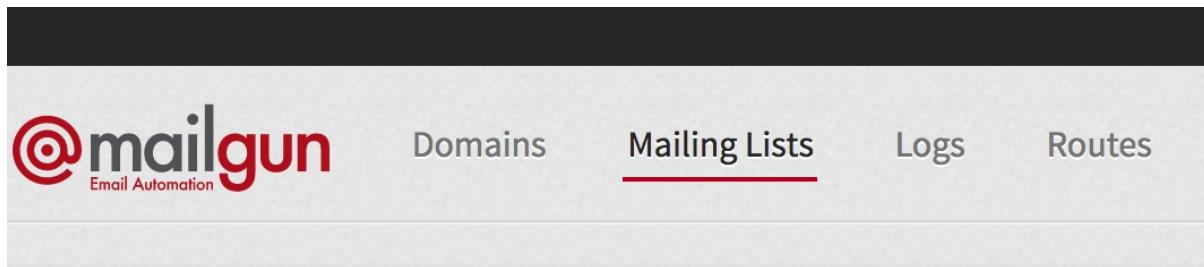
设置完成后，就可以保存设置了。

使用 Mailing List

MailGun 插件还提供了 Mailing List 的快速接入功能，可以帮助我们在网站上快速加入一个 Mailing List 的订阅表单。

创建一个 Mailing List

在 MailGun 后台选择 mailing list 选项，可以进入到 mailist 的主界面。



Message Delivery (Last 30 Days)

选择 **Create Mailing List** 选项，在新的页面设置具体的信息，设置邮件地址、名称、描述、权限。

Add Mailing List

 A screenshot of the 'Add Mailing List' form. It includes fields for Alias Address (hereismylist), Name (List #GitChat), Description (Here is a GitChat Examle List), and Access Level (Read Only). The 'Read Only' option is checked. A tooltip for 'Read Only' says: 'can post to this list. Used for mass announcements, newsletters.' Buttons at the bottom are 'Add Mailing List' (highlighted in black) and 'Cancel'.

Alias Address	hereismylist	@	sandbox1b885f98c
Name	List #GitChat		
Description	Here is a GitChat Examle List		
Access Level	<input checked="" type="checkbox"/> Read Only <input type="checkbox"/> Members <input type="checkbox"/> Everyone		

Add Mailing List **Cancel**

权限的说明：

Read Only：大多数人只读，部分用户可以发送邮件，比较适合新闻、公共类型的。

Members：列表内成员可以自由讨论，比较适合项目交流。

Everyone：任何人都可以发送邮件，比较适合用户反馈类型的，不过使用时需要注意进行垃圾过滤。

设置完成后，单击**Add Mailing List**按钮，即可创建一个新的列表。

回到 WordPress，在设置 | Mailgun Lists 中就可以看到我们刚刚创建的列表了。

复制这里的短代码，粘贴到任何一篇文章即可在文章下方加入对应邮件列表的订阅入口了。

Available Mailing Lists | [Back to settings](#)

You must use a valid Mailgun domain name and API key to access lists

List Address	Description	Shortcode
hereismylist@sandbox1234567890abcdef0e45b2815ed.mailgun.org	Here is a GitChat Examle List	[mailgun id="hereismylist@sandbox1234567890abcdef0e45b2815ed.mailgun.org"]

Multi-list subscription

To allow users to subscribe to multiple lists on a single form, comma-separate the Mailgun list ids.

Example: [mailgun id="list1@mydomain.com,list2@mydomain.com"]

就像这样：

欢迎使用WordPress。这是ls您的第一篇文章。编辑或删除它，然后开始写作吧！

Email:

SUBSCRIBE

TablePress

插件地址：<https://cn.wordpress.org/plugins/tablepress/>

WordPress 默认的编辑器是没有 table 的插入的，所以为了方便插入表格，我们可以使用 TablePress 插件。

创建新表格

单击菜单中的 **TablePress | 新建** 按钮，即可进入到新建表格的页面。

新建表格

表格名称:

表格名称

表格的名称或者标题。

描述 (可选) :

这是表格的描述

关于表格内容的描述。

行数:

5

表格的行的数目。

列数:

5

表格的列的数目。

设置表格的基础信息后，单击创建表格按钮。

在新的页面中展示了表格，可以选中表格，然后输入具体的内容。

The screenshot shows the 'Create Table' dialog in the WordPress editor. At the top, there's a 'Table Content' section with two input fields for 'Rows' (5) and 'Columns' (5). Below this, a preview area shows a 5x5 grid of table cells. The first cell contains the text '表格1' and the second cell contains '表格2'. The third cell contains the HTML code for an anchor tag: ''. The bottom half of the dialog is the 'Table Operations' section, which includes buttons for inserting links and images, merging cells, and performing row and column operations like hide, show, copy, insert, delete, and add.

如果想要在表格中插入连接、图片，可以先单击下方的表格“操作”按钮，再连接上方的表格中的某一项，以输入内容，就像下面这样。

表格内容

A	B	C	D	E
1 <input type="checkbox"/> 表格1	表格2			1 <input type="checkbox"/>
2 <input type="checkbox"/> height: 100 class="alignnone size-"		<a href="http://wordpress		2 <input type="checkbox"/>
3 <input type="checkbox"/>				3 <input type="checkbox"/>
4 <input type="checkbox"/>				4 <input type="checkbox"/>
5 <input type="checkbox"/>				5 <input type="checkbox"/>

表格操作

插入链接 高级编辑器 合并单元格: ?

选择的行: 选择的列:

选择的行: 选择的列:

增加 行 增加 列

下方是表格的一些设置，可以根据自己的需要设置。

表格选项

表格标题行: 表格的第一行是表头（标题行）。
表格脚注行: 表格最后一行是表格脚注。

交替行背景色: 交替变换连续行的背景颜色。
鼠标悬停时高亮行: 当鼠标选定在某行上方时，通过改变背景颜色来高亮该行。

显示表格名称: 将表格名称显示在表格的 。
显示表格描述: 将表格描述显示在表格的 。

附加 CSS 类: 编写用于设定表格样式的附加 CSS 类。注意这里不是编写[自定义 CSS](#)的地方！

DataTables JavaScript 库特性

使用 DataTables: 为此表格启用下面的 DataTables JavaScript 库特性:

排序: 为访客提供表格排序功能。
搜索/过滤: 为访客启用过滤或搜索表格的功能。只显示包含搜索词的行。
分页: 为访客启用表格分页功能（一次只显示表格中的有限几行）。
每页显示 行。

分页长度可变: 当使用分页功能时，允许访客改变每页显示的行数。
信息: 启用表格信息显示功能，显示当前可见数据的信息，如行号等。
水平滚动条: 显示水平滚动条，在查看列数比较多的表格时比较方便。

自定义命令: 来自 [DataTables 文档](#) 的额外参数将会增加到 JS 调用中。

设置完成后，单击保存更改按钮，就可以将表格保存了。

回到顶部，复制表格的短代码 [table id=1] :

表格信息

表格 ID:	1	简码:	<code>[table id=1 /]</code>
表格名称:	表格名称		
描述:	这是表格的描述		
最后更改:	2017年11月23日 上午9:27 由 admin		

粘贴到文章中的合适位置即可。具体的渲染效果如下：

显示 `10` 项结果 搜索:

◆ 表格1 ◆ 表格2 ◆ ◆ ◆ ◆ ◆

编辑

显示第 1 至 4 项结果, 共 4 项

◀ 上页 下页 ▶

导入表格

点击菜单中的**TablePress | 导入表格**命令, 可以上传我们自己的表格, 方便表格输入。

目前支持导入 CSV、XLS、XLSX 等格式, 如果使用 XLS、XLSX 导入不正常, 建议使用 CSV 格式, 因为 CSV 格式相对简单, 出错的可能更小。

导入表格

导入源: 上传文件 URL 地址 服务器上的文件 手工输入

选择文件: 未命名.csv
您可以通过将多个表格压缩成一个 ZIP 文件来同时导入。

导入格式: CSV - 符号分割记录格式
 HTML - 超文本标记语言
 JSON - JavaScript 对象表示法
 XLS - Microsoft Excel 97-2003 (试验)
 XLSX - Microsoft Excel 2007-2013 (试验)

新建、替换还是追加? : 新建 替换 追加 有表格后面追加行

要替换掉的, 或者追加到的表格:

选择完成后, 设置为新建表格, 然后单击“导入”按钮, 就会看到我们刚刚创建表格的页面。具体的设置项和我们刚刚创建表格时是一致的。

表格信息

表格 ID:	2	简码: [table id=2 /]
表格名称:	未命名.csv	
描述:	未命名.csv	
最后更改:	2017年11月23日 上午9:31 由 admin	

表格内容

	A ▲▼	B ▲▼	C ▲▼	D ▲▼	E ▲▼	F ▲▼
1	□	横向1	列表项1	列表项2		
2	□	横向1	1	2	3	
3	□	横向2	1	2	3	
4	□		1	2		

WordPress 相关资源分享

课程实践目标：了解一些 WordPress 的资源

这节课主要是分享一些个人觉得不错的站点，帮助读者更好的使用 WordPress。

WordPress 官方站点

WordPress Make · plugins

地址：<https://make.wordpress.org/plugins/>

这是 WordPress 官方的插件团队博客，在这里可以看到 WordPress 官方在插件方面的动态。

WordPress Make · themes

地址：<https://make.wordpress.org/themes/>

这里是 WordPress 官方的主题审核团队博客，如果希望你的主题更快的被审核，减少打回的次数，那么平时多关注是必要的。

WordPress Codex

地址：https://codex.wordpress.org/zh-cn>Main_Page

这是 WordPress 官方的 Wiki，任何关于 WordPress 本身的问题和使用问题都可以在这里找到对应的答案。

WordPress Developer

地址：<https://developer.wordpress.org/>

这是 WordPress 官方的开发者资源中心。相比于 Codex，他更加专注于 WordPress 开发，当你在开发时遇见问题，可以第一时间在这里搜索，这里的答案会是最标准的。

这个站点中我最常用的就是 <https://developer.wordpress.org/reference/>，你可以在这里查看每个函数的调用方式以及它调用的函数和它被哪些函数所调用。此外，可以在里面看到最新版本的 WordPress 提供的 API 有哪些。

第三方 WordPress 站点

ThemeForest

地址：<https://themeforest.net/>

ThemeForest（主题森林）可以说是目前国内外最大的主题市场，不止是 WordPress，包括 HTML、Drupal、设计模板等都包含。

如果你希望了解国外的 WordPress 流行风格，可以到 ThemeForest 去查看最近的流行主题。此外，如果你是一个主题作者，也可以将自己的主题托管在上面进行售卖。

CodeCanyon

地址：<https://codecanyon.net/>

提到了 ThemeForest，就不能不提 Codecanyon。两家都是 Envato 旗下的在线商城，前者专注于主题，而后者则专注于插件。

在 Codecanyon 当中可以找到大量的付费插件，可以根据自己的需要，购买对应的插件。同样的，你可以提交自己的作品到上面去售卖。

WP 大学

地址: <https://www.wpdaxue.com>

WP 大学是目前国内做 WordPress 相关内容做的最好的站点之一，站长多年关注 WordPress，网站内容涵盖 WordPress 新手教程、WordPress 使用技巧、WordPress 主题开发、WordPress 插件开发、WordPress 主题售卖等功能。堪称国内 WordPress 第一站。

WP 酷

地址: <https://www.mywpku.com/>

WP 酷是国内的一个专注 WordPress 的主题分享站点，如果找不到合适的 WordPress 主题，不妨去看看里面分享的主题，大多设计感不错，而且大部分免费。

此外，博主还会分享一些 WordPress 的使用技巧在博客中，值得一看。

我爱水煮鱼

地址: <http://blog.wpjamblog.com/>

我爱水煮鱼是国内知名的作者，他所写的 WordPress 主题教程、WordPress 插件教程是很多开发者的入门教程，值得一看。

爱找主题

地址: <http://www.2zzt.com/>

这个站点收集了大量的主题，如果你找不到想要的主题也可以到这里去看看，里面收集了大量的企业站主题、个人博客主题。

WordPress魂

地址: <http://www.2zzt.com/>

WordPress 魂的站长黄聪录制了两批课程“[《跟黄聪学Wordpress插件开发》](#)”和“[《跟黄聪学WordPress主题开发》](#)”，都是非常棒的视频课程，而且是免费的。如果你想学习相关开发，值得一看。

WOPUS

地址: <http://www.wopus.org/>

Wopus 是一个多年的 WordPress 站点，上面可以看到很多和 WordPress 相关的文章，不过这个站点最值得关注的还是其 IDC 业务，因为其本身是做 WordPress 相关的，旗下的虚拟主机也针对 WordPress 进行了很多优化。

知更鸟

地址: <http://zmingcx.com/>

知更鸟也是国内一个知名的 WordPress 主题制作人，他的主题广泛被 WordPress 博主使用，而且他会经常在博客上分享制作 WordPress 主题时遇见的坑和解决办法，值得一看。

WPBeginner

地址: <http://www.wpbeginner.com/>

WPBeginner 是国外的一个 WordPress 教程分享网站，里面每天都有新的 WordPress 文章发布，值得订阅，平时经常去看。不过里面的内容是英文的，对于一些英文不好的人来说，可能会稍微有些吃力，但整体难度不大。

Tutsplus

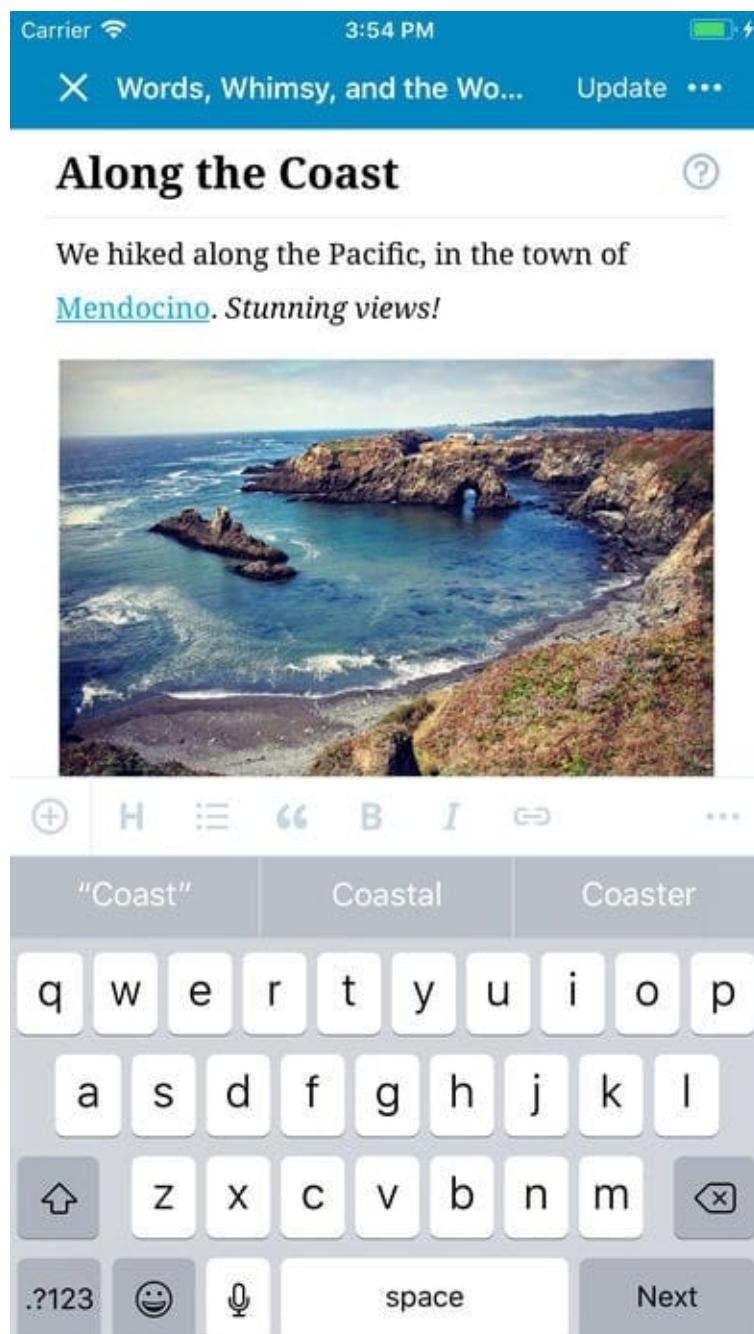
地址: <https://code.tutsplus.com/categories/wordpress>

Tusplus 是 envato 旗下的在线知识分享平台，上面有非常多的 WordPress 相关的教程。而且由于和 ThemeForest 是同一家总部，所以在内容上很多都是 ThemeForest 的作者写的。

WordPress 相关应用

WordPress for iOS [Free]

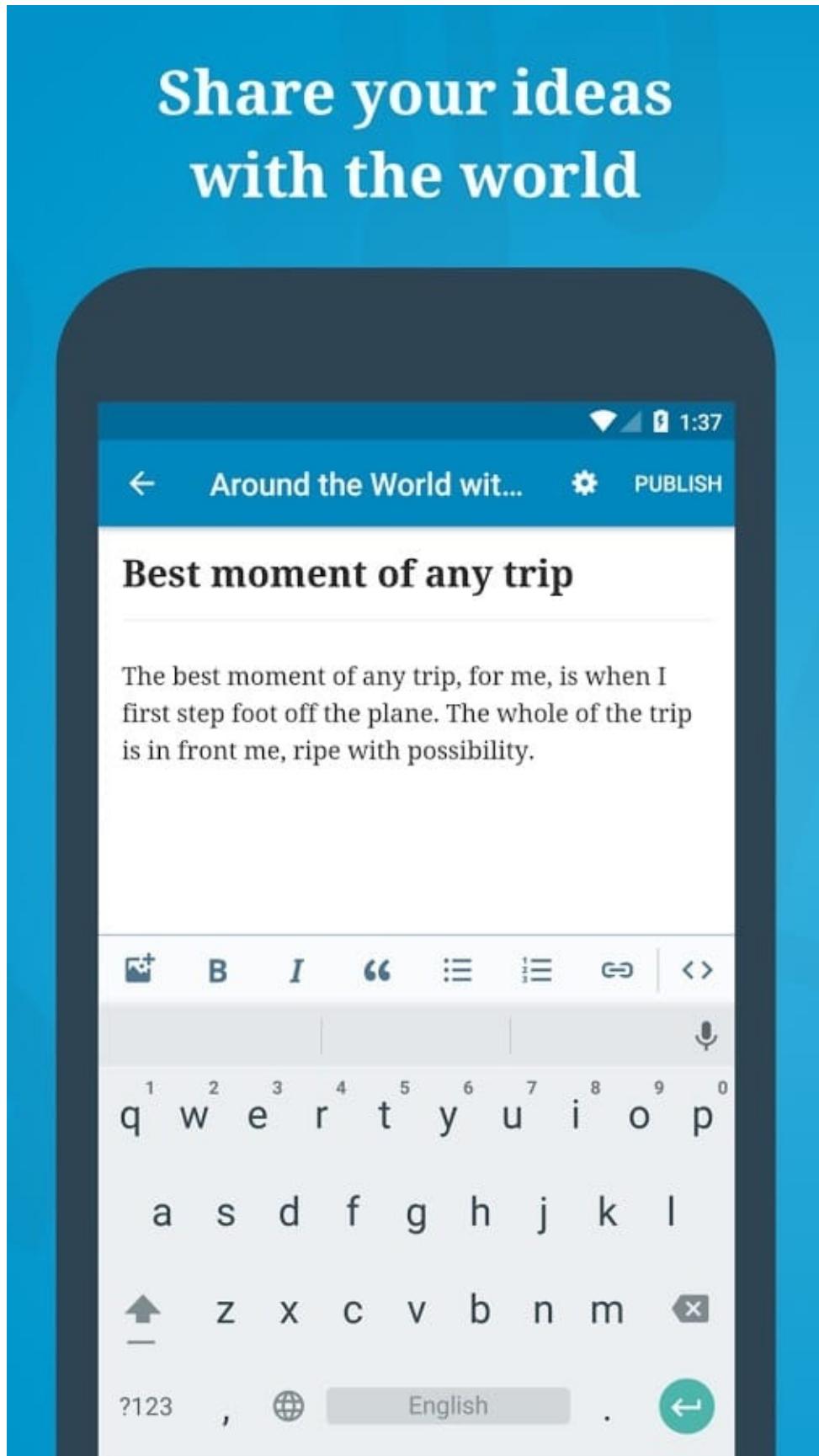
地址: <https://itunes.apple.com/us/app/wordpress/id335703880?mt=8>



WordPress 官方客户端配合 JetPack 插件将会非常好用，不过可能会比较臃肿，个人不喜欢。

WordPress for Android [Free]

地址: <https://play.google.com/store/apps/details?id=org.wordpress.android>



WordPress 官方客户端，也是需要配合 JetPack 插件才能更好用，不过个人不喜欢。

值得一提的是，官方的两个客户端都是开源的，你可以在 <https://github.com/wordpress-mobile> 找到。

MWeb for iOS / Mac [Paid]

地址: <http://zh.mweb.im/>

MWeb 是一款我非常喜欢的 Markdown 客户端, 这节达人课就是在 MWeb 上写的。



MWeb 对 WordPress 有非常好的支持, 只需要简单的设计就可以发布内容。

免费商用图片站点

写博客时总是要配一些图片, 免费商用的图片自然是用起来最放心了。在这里给大家分享一些可以下载到免费商用图片的站点。

Pixabay

地址: <https://pixabay.com/>

Pixabay 是全球最大的免费商用图片下载站点, 而且支持中文关键词搜索, 非常友好。我在给文章配特色图像会优先来这里搜索。

Gratisography

地址: <http://www.gratisography.com/>

Gratisography 的图片每周会更新, 而且是分好类的, 可以很方便的根据分类选择图片。

Magdeleine

地址: <http://magdeleine.co/>

Magdeleine 内有非常多偏向黑白对比的图片, 这个站点着重提供风景照。

Unsplash

地址: <https://unsplash.com/>

UnSplash 是一个完全免费的、无版权的高清图片资源网站, 有大量的图片。此外还提供了 Unsplash Wallpaper, 可以用 Unsplash 的每日图片作为你的壁纸。

Foodiesfeed

地址: <https://foodiesfeed.com/>

这个网站上全是吃的照片，如果你是美食博主，那这个网站必不可少，打开这个网站就会让你充满食欲。

Travel Coffee Book

地址: <http://travelcoffeebook.com/>

这个网站里的内容都是旅行者拍的全世界的风景照，都非常漂亮。

泼辣有图

地址: <http://www.polayoutu.com/collections>

泼辣有图是国内的开源无版权图片，每期会精选10张照片来分享。而且每张图片都会配上图片故事，也很有意思。

PixelSQUID

地址: <https://www.pixelsquid.com/stock-images>

这个网站的图片都是一个个不同的对象（Object），每个对象都可以通过鼠标调整显示的角度，很适合拿来做剪贴画，而且都是白底的图片。

Sozai

地址: <http://www.sozai-page.com/>

这个网站分享的都是免费的食材图片，也很适合美食博主。

WordPress 性能优化：动静分离

总论

为什么网站打开速度很重要

当用户第一次登录你的网站时，如果你在最开始的几秒钟没有捕捉到他们的注意力，没有让他们对内容产生兴趣，他们可能马上就关掉网站。调查显示如果网站 3 秒内没有打开，那么将会有 40% 的用户离开。如果网站加载时间太长，用户还没看就走了，那就白白流失了一个访客。

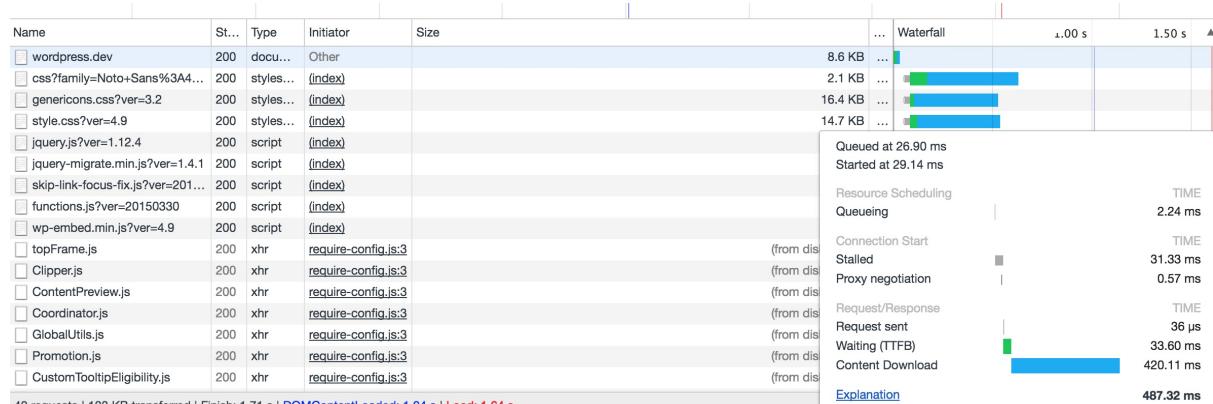
此外，网站速度还将影响你的网站在搜索引擎中的排名，速度缓慢、跳出率大，都会导致排名下降，流量减少。

接下来，我会用大约 4 节课的篇幅讲讲如何优化？

如何分析网站速度慢的原因

想要进行优化，首先要搞清楚问题在哪，找到问题的源头，才能更好的解决问题。

排查网站加载速度的问题一般可以使用 Chrome 自带的 Devtools 来进行检查。在 Devtools 中，我们可以看到每个文件的加载时间、加载顺序、加载大小等等。通过这个时间，可以很方便的找到我们的网站速度到底慢在哪。



此外，可以安装 WordPress 插件 Debug Bar 来调试数据库查询，并借此对耗时较长的数据库查询进行优化。

TOTAL QUERIES:	TOTAL QUERY TIME:
19	5.7 ms

```

SELECT option_name, option_value FROM wp_options WHERE autoload = 'yes'

require('index.php'), require('wp-blog-header.php'), require_once('wp-load.php'), require_once('wp-config.php'), require_once('wp-settings.php'), wp_not_installed, is_blog_installed, wp_load_alloptions #1 (1.0 ms)

SELECT * FROM wp_users WHERE user_login = 'admin'

require('index.php'), require('wp-blog-header.php'), require_once('wp-load.php'), require_once('wp-config.php'), require_once('wp-settings.php'), WP->init, wp_get_current_user, _wp_get_current_user, apply_filters('determine_current_user'), WP_Hook->apply_filters, wp_validate_logged_in_cookie, wp_validate_auth_cookie, get_user_by, WP_User::get_data_by #2 (0.3 ms)

SELECT user_id, meta_key, meta_value FROM wp_usermeta WHERE user_id IN (1) ORDER BY umeta_id ASC

require('index.php'), require('wp-blog-header.php'), require_once('wp-load.php'), require_once('wp-config.php'), require_once('wp-settings.php'), WP->init, wp_get_current_user, _wp_get_current_user, apply_filters('determine_current_user'), WP_Hook->apply_filters, wp_validate_logged_in_cookie, wp_validate_auth_cookie, get_user_by, WP_User->init, WP_User->for_site, WP_User->get_caps_data, get_user_meta, get_metadata, update_meta_cache #3 (0.2 ms)

```

此外，还可以借助一些其他的浏览器插件来进行网站速度的判别，比如 Yslow。

加速：提升带宽

最直接的方法：提升带宽

如果使用的是虚拟主机，那么一般来说不会存在太多的带宽问题，服务商一般提供的都是 100M 的网络，更多应该关注的是你的流量是否够用。

如果使用的是国外的 VPS，如 Linode、DigitalOcean、Vultr 等，那么你的带宽也不太是问题，你得问题更多是主机延时太高，导致请求的基本线比较高。

不过应该有不少人使用的还是国内的云主机，国内的云主机有个很大的特点，就是带宽贵，大部分人购买的可能都是 1M-5M 以内的云主机，很少有人会买更高的带宽，因为 5M 以上带宽的价格会飙升。

我们的网站大多数情况下都需要加载很多文件，各种不同的样式表、各种功能的 javascript 文件、各种图片，如果带宽很小（如 1M），那么加载速度就可能受到影响。因为云主机的带宽是固定的，如果图片的下载将带宽占满，那其他文件的下载就必定会卡住，所以，最直接的方案就是提升带宽。通过提供一个足够大的带宽，保证各个文件的下载。带宽变大，除了可以保证其他文件的下载，还可以保证让你的文件更快的被下载。

动静分离

在有些时候，我们没有办法提升带宽，毕竟当已经到了 5M 时，每 M 的提升价格都远超之前。不过，除了提升带宽，我们也的确有一些其他方法来加速网站，这就是——动静分离。

动静分离就是将网站的主程序（动态文件）和附件（静态文件）进行分割。虽然并没有实质上提升我们的带宽，但是通过将静态文件拆分，变相达到了动静分离的效果。

使用各种云存储来实现动静分离

目前各大云计算服务商都提供了云存储业务，而一些独立的云存储服务商也都干的有声有色。一般来说，我们可以选择 [阿里云的 OSS](#)、[又拍云的云存储](#)、[七牛云的云存储](#)。

其中，如果主机是阿里云的，推荐使用[阿里云的 OSS](#)，这样可以在数据存储时使用内网上传，不占用外网的带宽。同理，如果使用的是腾讯云的云主机，也可以使用[腾讯云的 COS](#)。

各家为了方便用户接入，基本都提供了对应的插件，具体可以看下表，将各家的免费额度也标注了出来。

厂商	免费额度	插件地址	使用说明
阿里云OSS	无	插件地址	使用说明
腾讯云COS	免费 50G 存储, 10G 流量	插件下载&&说明	

又拍云	免费 10G 存储, 15G 流量	插件下载	教程地址
七牛云	免费 10GB 存储, 免费 10GB 流量	插件地址	插件说明

当接入了对象存储后，你的附件都将上传至对象存储，加载的链接也会变成对象存储的链接，从而减少了从主站加载数据的需要。

留个思考题：有没有想过我们在第 4 课，媒体设置中的文件上传配置能如何帮助我们进行动静分离呢？

使用 CDN

在某些情况下，你可能觉得使用动静分离很麻烦，那有没有相对简单一点的方法呢？有。你可以试试 CDN（内容分发网络）。CDN 可以把网站中的内容缓存到一个个 CDN 节点，用户访问时会先访问 CDN 节点，如果 CDN 节点中有相关的内容，就不会再回源，请求你自己的服务器。

在使用 CDN 时，需要注意给你自己的网站开启网站静态化缓存（WP Super Cache），再在 CDN 处开启缓存，不然如果没有缓存，每次访问都会请求源站，反而可能会让你的网站变得速度更慢。

压缩图片

如果上述动静分离的方法对于来说，有一些麻烦，也可以选择将图片压缩从而来减少加载的时间。毕竟大部分情况下，一个页面中最大的还是图片。

图片压缩一般来说是两种方式，在上传之前的手动压缩和上传之后的程序自动压缩。

使用 [tinypng.com](#) 进行手动压缩

Tinypng 是一个在线的图片压缩服务，可以压缩 png 或 jpg 文件，可以在上传图片前先将文件批量上传到 Tinypng 中进行压缩，然后再上传到博客去。

The screenshot shows the homepage of tinypng.com. At the top, there are navigation links: HOME, PHOTOSHOP, DEVELOPER API, ANALYZER, and LOGIN. Below the header is a large dashed box with a download icon and the text "Drop your .png or .jpg files here!". Underneath it, it says "Up to 20 images, max 5 MB each.". To the right, there's a logo with the words "tiny" and "png" next to a bamboo illustration. Below the upload area, a table lists four compressed files:

File Name	Original Size	Status	Compressed Size	Download Link	Compression Ratio
Jietu20171120-222608@2x.png	2.8 MB	Finished	687.0 KB	download	-76%
Jietu20171122-010401@2x.png	94.3 KB	Finished	25.7 KB	download	-73%
Jietu20171122-010414@2x.png	94.2 KB	Finished	25.5 KB	download	-73%
Jietu20171122-010430.png	100.0 KB	Finished	28.2 KB	download	-72%

At the bottom, there are two buttons: "Save to Dropbox" and "Download all". The "Download all" button is highlighted with a red box.

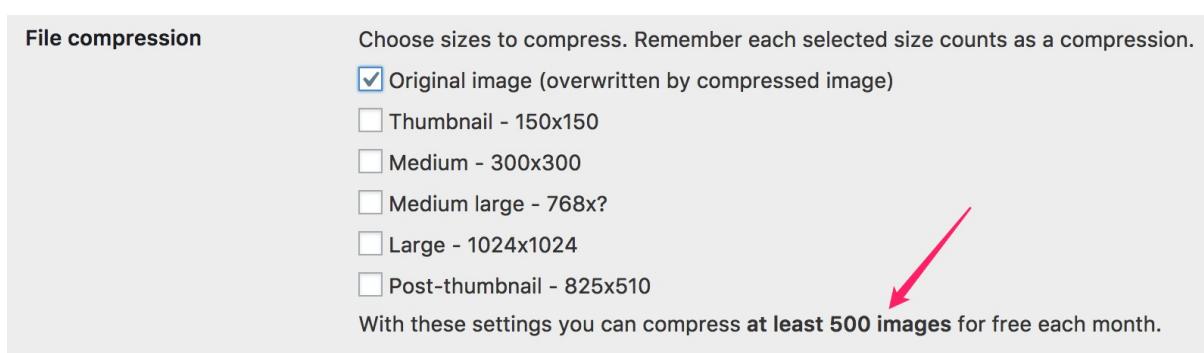
如果一次性上传了多个图片，可以单击下方的 download all 按钮来下载所有图片。

此外，如果感觉麻烦，也可以使用其官方推出的 [WordPress 插件](#)。



The screenshot shows the 'Compress JPEG & PNG images' plugin page on the WordPress.org plugin repository. At the top, there's a large image of a smiling panda. To the right of the image, the plugin title 'Compress JPEG & PNG images' is displayed in blue text, along with a progress bar indicating '正在安装...' (Installing...) and a '更多详情' (More details) button. Below the title, the tagline 'Speed up your w...' is visible, followed by '由 TinyPNG' (by TinyPNG). A rating section shows 5 stars (yellow) with '(107)' reviews, and a count of '100,000+个活跃安装' (100,000+ active installations). To the right, it says '最近更新: 2周前' (Last updated: 2 weeks ago) and includes a checkmark icon with the text '该插件兼容于您当前使用的WordPress版本' (The plugin is compatible with your current version of WordPress). A red arrow points from the text 'With these settings you can compress at least 500 images for free each month.' to the 'Thumbnail - 150x150' checkbox.

不过这个插件每个月只能压缩 500 张图片，所以在使用时注意设置。



The screenshot shows the 'File compression' settings in the Smush plugin. It lists several image sizes for compression: 'Original image (overwritten by compressed image)' (checked), 'Thumbnail - 150x150', 'Medium - 300x300', 'Medium large - 768x?', 'Large - 1024x1024', and 'Post-thumbnail - 825x510'. Below the list, a note states: 'With these settings you can compress at least 500 images for free each month.' A red arrow points from the text to the 'Thumbnail - 150x150' checkbox.

使用 Smush 插件压缩图片

Smush Image Compression and Optimization 是 WPMU 团队推出的图片压缩插件。

下载并安装插件

到 <https://wordpress.org/plugins/wp-smushit/> 下载插件，并在 WordPress 后台上传、安装、启用。

设置

安装完成后，会有一些设置项，可以根据我下方的解释，选择合适的选项。

Automatically smush my images on upload



When you upload images to your site, Smush will automatically optimize them for you.

上传自动压缩

Preserve my image EXIF data



EXIF data stores camera settings, focal length, date, time and location information in image files. EXIF data makes image files larger but if you are a photographer you may want to preserve this information.

保留exif信息

Resize my full size images



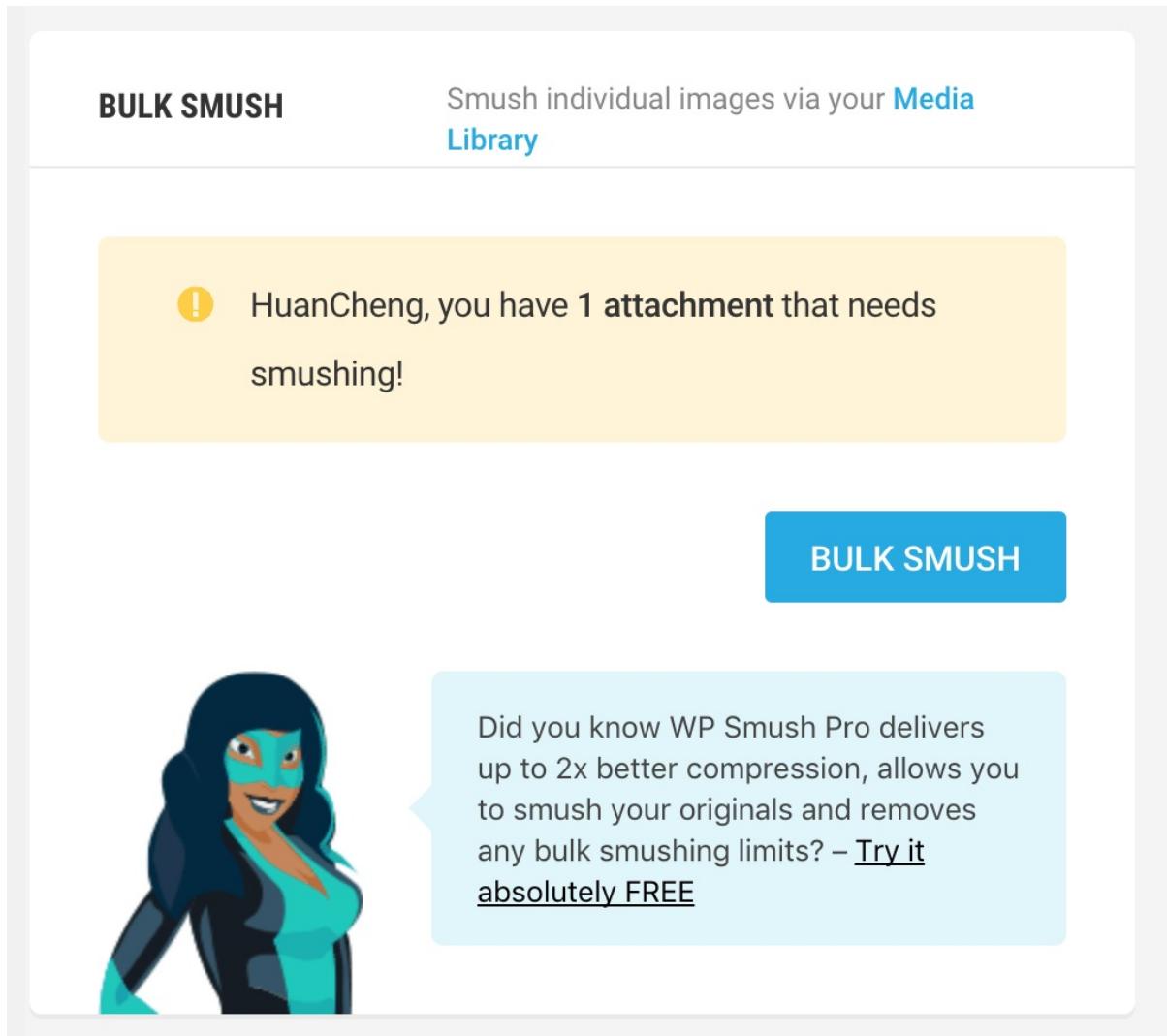
Set a maximum height and width for all images uploaded to your site so that any unnecessarily large images are automatically resized before they are added to the media gallery. This setting does not apply to images smushed using Directory Smush feature.

压缩所有图片

设置完成后，单击 **Get Started** 按钮，即可进入插件页面。

旧图片压缩

进入插件页面会看到提示：你有旧的图片需要压缩，单击 **Bulk SMUSH** 按钮，即可压缩已有附件。



压缩完成，即可看到相关的压缩信息：

The screenshot shows a performance analysis report from Smush.it. At the top right is a button labeled "RE-CHECK IMAGES". On the left, under the heading "STATS", there are two large numbers: "4" and "0". Below "4" is the text "Images smushed". To the right of "4" are two smaller numbers: "1" and "0". Below "1" is the text "Attachments smushed" and below "0" is "Images resized". Below these sections is a summary row with "Total savings" on the left and "181 B / 0.1%" on the right. Further down, there are sections for "Resize savings" and "Directory-smush savings", each with descriptive text and a "Choose directory" link.

Total savings	181 B / 0.1%
Resize savings	Save storage space by resizing your full sized uploads down to a maximum size. Enable image resizing
Directory-smush savings	Smush images that aren't located in your uploads folder. Choose directory

加入预加载功能

我们可以通过一些标签，为博客页面加入预加载功能，分别是 `dns-prefetch`、`preconnect` 和 `prerender`。

通过在网页中加入如下代码，则可以实现 DNS 的预解析：

```
<link rel="dns-prefetch" href="//wordpress.dev">
```

在新的页面加载前，先请求 DNS 的地址，减少解析的时间。

在网页中加入如下代码，则可以预先建立链接，通过预链接，浏览器后续可以直接使用准备好的链接获取数据，提高链接效率。

```
<link rel="preconnect" href="//wordpress.dev">
<link rel="preconnect" href="//cdn.wordpress.dev" crossorigin>
```

在网页中加入如下代码，可以让网页预先加载好下一页的数据，对于多页的数据，可以实现体验更好的加载。

```
<link rel="prerender" href="//wordpress.dev/?p=3">
```

当用户单击“下一页”按钮时会嗖的一下就跳到下一页去，极大地提高了用户体验。

看到这里，相信读者会有点崩溃，这怎么设置？难道一个个手动加？并不用，可以安装 [WordPress Instant Articles 插件](#) 来实现相关的功能。

WordPress 性能优化：服务器优化

加速：服务器优化

使用 Nginx 替代 Apache

虽然 Apache 对于 PHP 的处理更好，但是 Nginx 本身更加轻量，能够花费更少的系统资源。在 Nginx 节省的资源将会用到 PHP 中。

为 PHP 开启 OPCache

什么是 OPCache

当解释器完成对脚本代码的分析后，便将它们生成可以直接运行的中间代码，也称为操作码（Operate Code，OPCode）。OPCode Cache 的目地是避免重复编译，减少 CPU 和内存开销。如果动态内容的性能瓶颈不在于 CPU 和内存，而在于 I/O 操作，比如数据库查询带来的磁盘 I/O 开销，那么 opcode cache 的性能提升是非常有限的。

启用 OPCache

默认情况下 PHP 会安装 OPCache，但是不会启用，我们可以通过在 php.ini 中添加如下代码，开启 OPCache。

```
; 开关打开
opcache.enable=1

; 可用内存酌情而定，单位 megabytes
opcache.memory_consumption=256

; 对多缓存文件限制，命中率不到 100% 的话，可以试着提高这个值
opcache.max_accelerated_files=5000

; Opcache 会在一定时间内去检查文件的修改时间，这里设置检查的时间周期，默认为 2，定位为秒
opcache.revalidate_freq=240

; 设置缓存的过期时间
opcache.revalidate_freq=0

; 控制内存中最多可以缓存多少个PHP文件
opcache.max_accelerated_files=7963

; 是否快速关闭，打开后在 PHP Request Shutdown 的时候回收内存的速度会提高
opcache.fast_shutdown=1

; 不保存文件/函数的注释
opcache.save_comments=0
```

为 MySQL 开启 Query Cache

通过为 MySQL 开启 Cache，可以加快 WordPress 的查询速度。

在 MySQL 的配置文件 my.cnf 中添加如下代码，并重启 MySQL，即可开启 MySQL Query Cache。

```
query_cache_type = 1
query_cache_limit = 1M
query_cache_size = 16M
```

但是，需要注意的是，MySQL Query Cache 并不适合所有场景，如果你的博客浏览量较大、且更新频率不高，可以考虑开启 Query Cache，如果更新频繁且浏览量不大，那么 Query Cache 反而可能带来负效应。

<http://www.orczhou.com/index.php/2009/08/query-cache-1/>

为服务器开启 Gzip 压缩

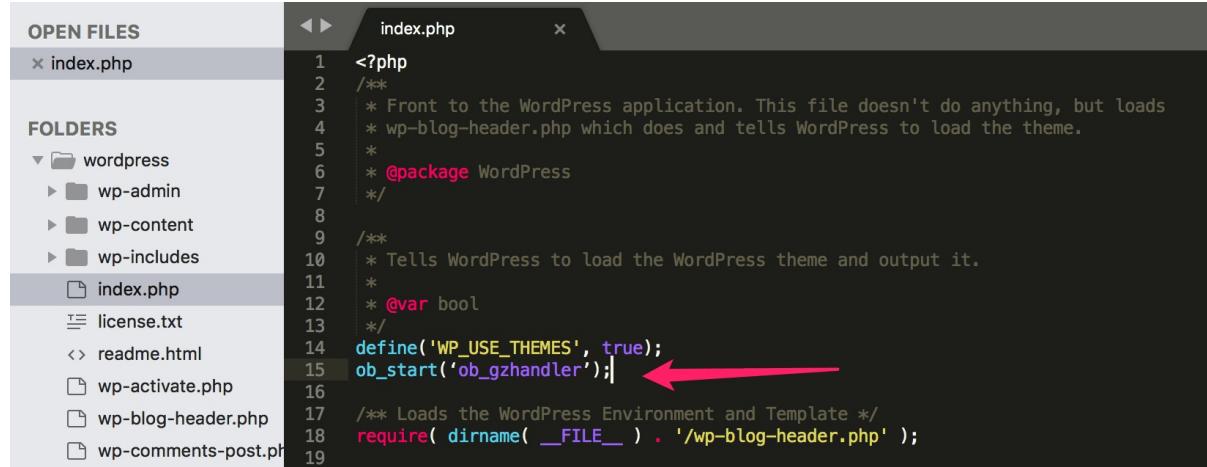
为程序加入 Gzip 压缩，可以有效减少传输数据的大小。一般来说，我们可以使用如下几种方式开启 Gzip。

cPanel 可以直接在面板的“优化网站”中开启 Gzip。

在 WordPress 中开启 Gzip

可以在根目录下的 index.php 中添加如下代码，以开启 Gzip 压缩。

```
// 此代码需要放在 define('WP_USE_THEMES', true); 后,
ob_start('ob_gzhandler');
```



通过 Apache 的 .htaccess 文件开启

可以在网站根目录中的 .htaccess 文件中添加如下代码，以开启 apache 的 gzip 压缩：

```

<IfModule mod_expires.c>
ExpiresActive On
ExpiresByType image/gif A2592000
ExpiresByType image/jpeg A2592000
ExpiresByType image/png A2592000
ExpiresByType image/x-icon A2592000
ExpiresByType application/x-javascript A604800
ExpiresByType text/css A604800
</IfModule>
<IfModule mod_deflate.c>
SetOutputFilter DEFLATE
AddOutputFilterByType DEFLATE text/html text/css image/gif image/jpeg image/png application/x-javascript
</IfModule>
```

通过 Nginx 的 Conf 文件开启

在 nginx.conf 中添加如下代码，则可以开启 Gzip 压缩：

```

gzip on;
gzip_disable "msie6";

gzip_vary on;
gzip_proxied any;
gzip_comp_level 6;
```

```
gzip_buffers 16 8k;
gzip_min_length 256;
gzip_http_version 1.1;

gzip_types text/plain text/css text/x-component text/html application/json application/x-javascript text/xml application/xml application/xml+rss text/javascript application/javascript image/x-icon image/svg+xml image/jpeg image/gif image/png font/opentype;
```

通过 php.ini 开启

可以通过给 php.ini 中添加如下代码，来开启 gzip 压缩：

```
zlib.output_compression=On
zlib.output_compression_level = 5
```

通过 WP Super Cache 开启

通过 WP Super Cache 可以开启压缩，具体可以看下第 10 课相关内容。

为文件加入 expires 头信息

通过加入 expires 头信息可以让静态文件优先使用缓存，提升加载速度。

大部分时候，我们上传的图片都不会发生改变，使用缓存可以帮助加速浏览。

可以通过在 .htaccess 中添加如下代码。

```
# BEGIN Expire headers
<ifModule mod_expires.c>
    ExpiresActive On
    ExpiresDefault "access plus 5 seconds"
    ExpiresByType image/x-icon "access plus 2592000 seconds"
    ExpiresByType image/jpeg "access plus 2592000 seconds"
    ExpiresByType image/png "access plus 2592000 seconds"
    ExpiresByType image/gif "access plus 2592000 seconds"
    ExpiresByType application/x-shockwave-flash "access plus 2592000 seconds"
    ExpiresByType text/css "access plus 604800 seconds"
    ExpiresByType text/javascript "access plus 216000 seconds"
    ExpiresByType application/javascript "access plus 216000 seconds"
    ExpiresByType application/x-javascript "access plus 216000 seconds"
    ExpiresByType text/html "access plus 600 seconds"
    ExpiresByType application/xhtml+xml "access plus 600 seconds"
</ifModule>
# END Expire headers
```

WordPress 性能优化：缓存

加速：加入缓存机制

WordPress 本身提供了一种叫做 WordPress Object Cache 的对象缓存机制，它是把需要缓存的内容按照 Key-Value 这样的模式进行缓存。Key-Value 形式的数据，要远比我们在数据库中查询要快的多。

WordPress 的三种缓存机制

文件缓存：文件缓存是把 WordPress 默认的对象缓存的对象存储为文件，对外提供服务时，WordPress 直接读取文件中的缓存来进行渲染和输出。但是此类缓存对硬盘的 I/O 要求极高，所以自 WordPress 2.5 就被弃用。不过如果虚拟主机无法正常运行后面两种缓存，也可以试试文件缓存，通过安装 [WP File Cache](#) 来启用文件缓存。

内存缓存：内存缓存是把 WordPress 默认的对象缓存的对象保存到内存中去，下一次访问的时候，直接到内存中获取内容，这样就没有了文件缓存的 I/O 操作，并且 SQL 查询也减少了很多，整个系统都能变快很多。

内存缓存和文件缓存一个相同的地方也是网站的动态内容根据功能分成很多个部分，分别对每个部分缓存，而不是把这个页面当作一个整体缓存了，所以访问一个页面还是需要从内存中读取多个内容。

HTML 静态缓存：HTML 静态缓存可以把 WordPress 页面缓存一个静态的 HTML 页面存到服务器上，下次访问该页面的时候，不需要再次运行 PHP 代码，直接从服务器返回这个 HTML 文件即可，这样就大大降低了 CPU 的占用率，但是使用 HTML 静态化缓存插件，也有一个不好的地方，比如日志流量、最新日志，这些动态的内容不会时时更新。

内存缓存：使用 Memcached 和 Batcache 进行缓存

我们可以安装 Memcached 来实现数据查询的缓存，在启用了 Memcached 缓存后，WordPress 会将在数据库中查询到的数据以对象的形式缓存进入内存中。我们每次访问时，WordPress 就会访问内存中的缓存，将对象读取出来进行页面渲染，可以有效的减少数据库的查询。

而 BatCache 则是将页面整个缓存进入到内存中，用户进入后，WordPress 直接读取缓存中的页面，并返回给访客，能够再次进行提速。

安装 Memcached 和 Batcache

安装 Memcached 和 BatCache 是有要求的，一般来说，要求使用的是独立的主机而非虚拟主机（需要有 SSH 权限），不过国外的一些 cPanel 主机也提供了 Memcached 的服务，也可以使用，具体可以咨询一下你的虚拟主机提供商。

首先，在服务器上安装 Memcached

在主机上使用命令安装 Memcached：`yum -y install memcached` 或 `apt-get -y install memcached`。

编译 PHP 的 Memcached 的支持

为 PHP 编译 `Memcached` 拓展。

下载 Memecached 插件

到 <https://wordpress.org/plugins/memcached/> 下载 WordPress 官方提供的 Memcached 插件，解压后将其中的 `object-cache.php` 移动到 `wp-content` 目录下（此处注意，不是 `wp-content/plugins/` 目录）。

下载 BatCache 插件

到 <http://wordpress.org/extend/plugins/batcache/> 下载 WordPress 官方提供的 BatCache 插件，解压后，将其中的 `advanced-cache.php` 移动到 `/wp-content/` 目录。

开启缓存

在 wp-config.php 文件中加入如下代码，开启缓存：

```
define('WP_CACHE', true);
```

HTML 静态缓存：使用 WP Super Cache 进行缓存

除了使用 Memcached 进行缓存，我们还可以使用 WP Super Cache 来进行缓存。WP Super Cache 把整个网页直接生成 HTML 文件，这样 Web 服务器就不用解析 PHP 脚本，通过使用这个插件，能使得你的 WordPress 博客将显著的提速。

安装 WP Super Cache

安装 WP Super Cache

到 <https://wordpress.org/plugins/wp-super-cache/> 下载插件，并上传到后台进行安装、启用。

开启缓存

插件安装完成后，会在设置中多一个 WP Super Cache 的设置页面，单击进入设置页面，进入“通用” TAB，勾选启用缓存功能选项。



再到“高级” TAB 中勾选所有的推荐选项。这样会帮助我们自动针对未登录用户启用缓存、并对所有页面加入 Gzip 压缩。

WP Super Cache 设置

-

缓存功能 Enable Caching

Cache Delivery Method
 Simple (推荐)
 Expert

Expert caching requires changes to important server files and may require manual intervention if enabled. Nginx rules can be found [here](#) but are not officially supported.

杂项

不要为已知用户缓存。 (推荐)
 不要为 GET 请求缓存。 (地址结尾为?x=y)
 压缩页面以便让来访者更快浏览。 (推荐)

压缩默认已禁用，因为有些主机对压缩过的文件处理有问题。勾选或取消勾选该功能会清除缓存。
 Cache HTTP headers with page content.
 缓存重建。当新缓存生成时调用缓存文件给匿名用户。 (推荐)
 304未修改浏览器缓存。表示网页自从它最后一次请求后未被修改。 (推荐)

默认情况下禁用304支持，因为某些主机有与过去使用的标头 (headers) 的问题，
 让已知用户匿名使他们浏览的内容是缓存文件。

 自豪地告诉世界[Stephen Fry proof!](#)! (在您的博客底部显示一行感谢作者信息)

CDN 选项

如果需要开启 CDN，可以在 CDN 页面中进行配置，勾选启用 CDN，并设置 CDN 域名即可：

The screenshot shows the 'WP Super Cache' settings page under the 'Performance' tab. It includes sections for 'Site URL' (http://wordpress.dev), 'Off-site URL' (http://mycdndomain.dev), 'Include Directories' (wp-content, wp-includes), and 'Exclude Patterns' (.php). Red arrows highlight the 'Site URL' field, the 'Off-site URL' field, and the 'Exclude Patterns' field.

开启 CDN 支持

Site URL
http://wordpress.dev
The URL of your site. No trailing / please.

Off-site URL
http://mycdndomain.dev
这个新的URL替换http://wordpress.dev用来重写, 不要用 / 结尾,
例如 http://cdn.wordpress.dev/wp-includes/js/jquery/jquery-migrate.js .

包括目录
wp-content,wp-includes
Directories to include in static file matching. Use a comma as the delimiter.
Default is wp-content, wp-includes , which will be enforced if this field is left empty.

如果是之前有则排除
.php
Excludes something from being rewritten if one of the above strings is found in the URL. Use a comma as the delimiter like this, .php, .flv, .do , and always include .php (default).

清除缓存

当我们的内容发生变化，需要清空缓存时，可以单击设置页面中的“内容” TAB 按钮，单击下方的删除已过期文件、删除缓存按钮来清空系统的缓存。

缓存内容

缓存统计信息不是自动生成的，您需要点击下面的链接来重新生成。

[重新生成缓存统计信息](#)

缓存统计已于 4分钟前生成。

WP-Cache (0KB)

0 已缓存页面

0 已过期页面

WP-Super-Cache (0KB)

0 已缓存页面

0 已过期页面

[列出所有已缓存的文件](#)

到期文件是那些存在时间大于1800秒的文件。它们仍在被调用并会定期被删除。

[删除已过期文件](#)

[删除缓存](#)

WP Super Cache 的功能很强大，就不在这里一一说明了，默认的这些设置项，已经足够我们的日常使用了。

静态化：Super Static Cache 插件

如果觉得 HTML 网页缓存也无法满足，那么只有静态化插件才能满足你了。

江湖上盛传的 cos-html-cache 插件已经太久没更新了，所以这里推荐用 Super Static Cache 来进行静态化。

使用 Super Static Cache 进行缓存

下载插件

前往 <https://wordpress.org/plugins/super-static-cache/> 下载该插件，并上传到后台进行安装、启用。

设置插件

安装完成后，进入设置中的 **Super Static Cache** 选项，可以看到如下的配置项，选择其中的一种模式开启即可。

Super Static Cache选项

基本设置 **高级设置**

缓存模式

Direct模式将会把缓存内容直接存放在服务器上，这是最节省资源的模式，但是这种方式会造成缓存内容管理困难。

PHP模式将会把缓存内容存放在一个目录里，这样将会方便管理，但PHP模式会依赖数据库服务器，如果您的数据库服务器宕机，网站将不可访问。

Rewrite模式会把缓存内容放到一个目录，缓存成功之后，网站不再依赖数据库，但是您需要在服务器上添加一条伪静态规则。

关闭 Direct模式 PHP模式 Rewrite模式 (推荐)

压缩

压缩页面以节省硬盘空间和访问时间。

部分服务器对压缩功能支持有限，如开启此项设置，请确保工作正常。

开启 关闭

开启后，会看到网站的根目录下多了一个 index.html：

```
wp@wordpress: ~
$ ls
index.html      wp-comments-post.php  wp-login.php
index.php        wp-config-sample.php wp-mail.php
license.txt      wp-config.php       wp-settings.php
readme.html      wp-cron.php        wp-signup.php
super-static-cache wp-includes        wp-trackback.php
wp-activate.php  wp-content         xmlrpc.php
wp-admin          wp-links-opml.php
wp-blog-header.php wp-load.php
```



这就是生成的静态文件。

WordPress 性能优化：程序优化

除了之前提到的动静分离、服务器优化、缓存加速以外，还有一点很重要的就是程序层面的优化。

加速：程序优化

关闭 Google 字体

Google 字体绝对是拖慢国内 WordPress 浏览速度的一个重要原因。可以通过在主题的如下插件来关闭 Google 字体。

```
function remove_open_sans() {
    wp_deregister_style( 'open-sans' );
    wp_register_style( 'open-sans', false );
    wp_enqueue_style('open-sans','');
}
add_action( 'init', 'remove_open_sans' );
```

加速 Gravatar 头像

由于 Gravatar 的主域名被屏蔽，可以考虑使用 https 的 Gravatar 域名，或者将图片缓存到本地。

将下面的代码加入到主题的 functions.php 中，并在根目录中创建一个 avatar 目录，给予 PHP 可写的权限，上传一个 default.jpg 作为默认的头像。

```
function my_avatar($avatar) {
    $tmp = strpos($avatar, 'http');
    $g = substr($avatar, $tmp, strpos($avatar, "'", $tmp) - $tmp);
    $tmp = strpos($g, 'avatar/') + 7;
    $f = substr($g, $tmp, strpos($g, "?", $tmp) - $tmp);
    $w = get_bloginfo('wpurl');
    $e = ABSPATH . 'avatar/' . $f . '.jpg';
    $t = 1209600; //设定时间为14天，单位为秒
    if ( !is_file($e) || (time() - filemtime($e)) > $t ) { //头像不存在或时间超过14天，重新获取
        copy(htmlspecialchars_decode($g), $e);
    } else $avatar = strtr($avatar, array($g => $w . '/avatar/' . $f . '.jpg'));
    if (filesize($e) < 500) copy($w . '/avatar/default.jpg', $e);
    return $avatar;
}
add_filter('get_avatar', 'my_avatar');
```

加速 PHP 代码

使用最新版的 PHP

PHP 目前在线上跑的版本有很多，从 5.4、5.5 到最新的 7.1 在线上都有，而自 PHP 7 开始，PHP 的性能有大幅度的提升，所以建议将你的旧版 PHP 升级到最新版的 PHP 7。

需要注意的是，部分插件不兼容 PHP 7，要确认使用的插件是否兼容 PHP 7。

控制插件、主题的数量

一方面要尽可能的减少用到的插件，插件越多，程序就越臃肿，运行的效率就越低。确认你是否需要使用到相关功能？能不能将相关的功能使用其他的方式来引入？对于一些评分较低的插件，可以考虑使用评分更高的同类插件来替换。

另一方面，对于不用的主题和插件，可以将其删除，以减少 WordPress 对目录的扫描。

优化主题

首先，要尽可能的选择框架更小、引用更少的主题，用更少的代码实现更多的功能。对于一些引用文件特别多的主题，可以考虑减少一部分引用文件，去掉一些用处不大的功能。如果实在无法去除，则确保对每个引用文件都进行压缩（minify），去除代码中无用的注释和空格，缩小文件大小。此外，还可以借助诸如 Better WordPress Minify 之类的插件来简化请求，将多个 CSS/JS 文件简化为一个 CSS/JS 文件，也能提升网站的加载速度。

其次，对于主题使用到的函数，可以考虑使用获取缓存数据的函数来替换直接从数据库查询数据的函数。比如 `get_the_terms` 和 `wp_get_object_terms`。

It should be noted that the results from `wp_get_object_terms` are not cached which will result in a db call everytime this function is called. For performance, functions like `get_the_terms()` (which the results of has been cached), should be used.

https://codex.wordpress.org/Function_Reference/wp_get_object_terms.

通过使用缓存数据可以有效减少数据库查询，提升数据的查询结果。

如果一定要从数据库中直接查询，则要注意对查询结果进行缓存，可以使用 [WordPress Object Cache 函数](#)来进行处理。

此外，也要精简主题的结构，去除一些无用的标签，减少 HTML 的代码，也可以有效的提升网站加载速度。

优化图片

这里的图片优化不同于前面的图片优化，前面的优化更多是用户层面的，而这里的图片优化值得是 CSS 雪碧图（Sprites），通过使用 CSS 雪碧图，可以减少 CSS 调用的图片的数量，加速页面的展示。

加入 LazyLoad

为博客加入 LazyLoad 可以让页面不在一开始就加载图片，从而让图片的加载分步进行，能够更好的利用服务器的带宽，将压力分流到整个页面的浏览过程中。除此之外，还可以给你的服务器减少一些请求的压力和流量的使用，如果图片没有被浏览到，则不加载相关的图片。

使用 ROCKET LAZYLOAD

到 <https://wordpress.org/plugins/rocket-lazy-load/> 下载插件，在后台启用该插件后，可以设置图片、iFrame 和 YouTube 视频的 LazyLoad。

Lazyload

LazyLoad displays images, iframes and videos on a page only when they are visible to the user.

This mechanism reduces the number of HTTP requests and improves the loading time.



Images



Iframes & Videos



Replace Youtube videos by thumbnail

 ✓ Save changes

优化数据库

WordPress 有修订版本的功能，可以使用插件来删除旧文章的修订版本，来优化数据库，达到更好的查询效果。

安装 WP-Optimize

可以到 <https://wordpress.org/plugins/wp-optimize/> 下载插件，并在后台安装、启用插件。

清除数据库中的垃圾

打开后台的 WP-Optimize 插件页面，勾选要优化的项目，单击上面的 **Run All Selected Optimizations** 选项，即可进行优化。

Optimizations

警告：It is best practice to always make a backup of your database before any major operation (optimizing, upgrading, etc.).

Run all selected optimizations

Take a backup with UpdraftPlus before optimizing

[Follow this link to install UpdraftPlus, to take a backup before optimization](#)

Optimization	Notes	
<input type="checkbox"/> 优化数据库表	Tables using the InnoDB engine (12) will not be optimized. Other tables will be optimized (0).	Run optimization
<input checked="" type="checkbox"/> 删除所有文章修订记录	4 post revisions in your database	Run optimization
<input checked="" type="checkbox"/> Clean all auto-drafts and trashed posts	2 auto draft posts in your database 1 trashed post in your database	Run optimization
<input checked="" type="checkbox"/> Remove spam and trashed comments	暂无垃圾评论 No trashed comments found	Run optimization
<input checked="" type="checkbox"/> 删除未清除未审核的评论	暂无未审核评论	Run optimization
<input type="checkbox"/> Remove expired transient options	2 expired transient in your database	Run optimization
<input type="checkbox"/> 删除 Pingbacks	暂无 Pingbacks	Run optimization
<input type="checkbox"/> 删除 Trackbacks	暂无 Trackbacks	Run optimization

开启 ETag

在为我们的博客打开 ETag 后，浏览器将自动判断内容是否更新，如果未发生更新，则使用旧的数据，这样可以有效帮助我们减少网页的加载，提升页面的加载速度。

在 WordPress 中，我们可以通过安装 [Smart WordPress](#) 插件来开启 ETag。

插件安装完成后，进入插件的设置页面，可以看到有这样的选项，勾选选项打开 ETag 的插入即可。

ETag settings

The [ETag](#) (or Entity Tag) ist an HTTP Header used to validate cached responses. Its unique value is generated by combining different properties of your content (ID, Content, Publishing Date, ...).

Add ETag Header



i Check this setting if you want an ETag on your site. (Recommended)

Generate Weak ETag



i Check this setting if you want to generate a [weak ETag](#) instead of a [strong ETag](#). (Recommended)

这个插件除了可以帮助我们控制 ETag 以外，还可以帮我们设置 `Last-Modified` Response Header。

Last-Modified settings

The [Last-Modified header](#) is an HTTP Header that displays the date, and the time at which the current resource was last modified. This header is used to check if a cached version of a resource differs from its online version.

Add Last-Modified Header



i Check this setting if you want a Last-Modified Header on your site.

以及 `Cache-Control` Response Header。

Cache-Control settings

The [Cache-Control Header](#) is an HTTP header that specifies two things: The period a resource should be cached and the way this caching should take place.

Add Cache-Control Header



i Check this setting if you want a Cache-Control Header on your site. (Recommended)

Maximum Age for Cache (Standard)

0

i Set the maximum time in seconds that you want the response to be cached and reused.

Maximum Age for Cache (Search Results)

0

i Set the maximum time in seconds that you want the response to be cached and reused on search result pages.

Maximum Age for Cache (Logged In)

0

i Set the maximum time in seconds that you want the response to be cached and reused when the user is logged in.

Old content Threshold

0

i A piece of content older than the "Old content threshold" will be regarded as old content. ([learn more](#))

通过上述的这些设置，可以帮助我们有效的提升浏览器对博客缓存的利用率。

加速浏览

鼓励和提醒你的用户使用最新版的高性能浏览器，如 FireFox、Opera、Chrome，这些浏览器的高性能渲染引擎会比 IE 有更好的表现，间接实现了对 WordPress 的加速。

WordPress 安全固化

什么是安全

安全不是绝对的，安全是一个持续不断的过程，你需要不断的调整、检查问题来降低安全风险。

安全措施

采用不同防御深度的安全解决方案

没有任何一个解决方案可以处理所有安全问题，我们需要使用多种不同的解决方案搭配使用来解决不同深度的安全问题，在多层次的解决方案处理下，即使某一层的解决方案出现了故障，整个系统仍然处于保护之下。

限制访问权限

对于一些有特殊安全需要站点，要尽可能的减少能够访问到站点的人数，尽可能的减少外部入口的展示。

选择靠谱的主机商

选择靠谱的主机商，或者自己建设 VPS。

在选择主机商时，尽可能选择那些历史悠久、风评比较好的。免费、廉价的主机随时可能遇见风险。而比较新的主机商可能在安全问题上无法做到很好的预防，你自己虽然安全做的非常棒，但是在主机商那里出了问题，也是不可以的。

遵守最小权限原则

在配置 WordPress 时，应当授予其满足其需要的最小权限。官方的建议是给目录 755 权限，给文件 644 权限。

设置目录/文件的权限

1.wp-admin 目录

使用 Web Server 的 Basic Auth 来为 wp-admin 的设置一层验证，从而让别人无法直接对你的后台进行爆破。

2.wp-includes 目录

在网站根目录中的 .htaccess 中添加如下代码，可以实现对 wp-includes 目录的保护，让这些文件无法被直接访问。

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^wp-admin/includes/ - [F,L]
RewriteRule !^wp-includes/ - [S=3]
RewriteRule ^wp-includes/[^\/]+\.\php$ - [F,L]
RewriteRule ^wp-includes/js/tinymce/langs/.+\.\php - [F,L]
RewriteRule ^wp-includes/theme-compat/ - [F,L]
</IfModule>
```

3.wp-content/uploads 目录

在 .htaccess 文件的顶部添加如下代码，可以禁止 uploads 目录下的 PHP 执行权限，会减少大部分的麻烦。

```
<Files ~ "\.ph(?:p[345]?\|t|tml)$">
    deny from all
</Files>
```

4.wp-config.php

在 .htaccess 文件的顶部添加如下代码，可以让配置文件更加的安全。

```
<files wp-config.php>
order allow,deny
deny from all
</files>
```

关闭文件的修改功能

在 wp-config.php 文件中添加如下代码，可以关闭在线的文件编辑，这样可以保证不会因为账户密码丢失导致的被在主题和插件中植入恶意代码。

```
define('DISALLOW_FILE_EDIT', true);
```

监控网站的文件变动

可以使用 iThemes Security 或 Wordfence 来监控 WordPress 的文件变动。

文件的变动情况也会帮助你更好的认识自己的网站状态。

不同站点进行功能隔离

避免在用一个虚拟主机账户下安装多个应用程序，以免受到其他应用的波及。对每一个应用程序应该设置一个独立的账户，以保证其安全。

为博客使用不同的数据库

如果一个主机上有多个程序，最好为每个程序创建一个数据库和数据库用户，这样不会导致你的一个应用程序被攻破连累其他程序也被攻破。

日常备份

为你的站点做备份，并且要经常验证你的备份是否有效（不要忘了 gitlab 丢数据结果备份也失效的惨剧）。

给站点做一个恢复清单，并时常演习，一旦出了问题，直接按照计划进行即可。

从可信的源获取代码

不要使用不受信任的源代码，最好是从 WordPress 官方插件仓库、或者是开发者主页下载相关的代码。网上流传的破解主题和插件可能会加入一些恶意的代码，给网站留下隐患。

但是这个安全只是相对的，因为按照经验，WordPress 官方只会审查第一版，后续的版本都是开发者自行更新的。如果开发者加入了恶意代码，第一时间是没有人审查的，所以除了从官方源下载，还要选择一些知名的插件。

在服务端使用最新的软件

对于使用 VPS、云主机的用户，建议使用最新版本的软件（包括操作系统、Web Server、PHP、MySQL 等）。

如果使用的是虚拟主机，一般来说虚拟主机商会帮你搞定这些事情，只需要使用即可。

使用最新版本的 WordPress

一些用户在使用时会选择锁死 WordPress 的版本，这样会导致你无法及时获得 WordPress 的安全更新。出于安全起见，建议将站点实时更新，或者定期进行更新，不要锁死在一个版本上。

在 wp-config.php 中加入如下的代码，可以开启 WordPress 核心的自动更新：

```
define( 'WP_AUTO_UPDATE_CORE', true );
```

使用最新的插件、主题

大部分安全漏洞都来自你安装的主题、插件，所以尽可能的使用最新的插件和主题。这些漏洞可能只是开发者的一时疏忽，所以一旦插件发布了安全更新，还是早升级为妙。

如果一个插件/主题长期不更新的话，最好考虑一下是否还使用整个插件，看看能否换为其他的同类项的插件/主题，尽量避免使用无人维护的插件。

在主题的 `functions.php` 中添加如下代码，可以开启插件和主题的自动更新：

```
add_filter( 'auto_update_plugin', '__return_true' );
add_filter( 'auto_update_theme', '__return_true' );
```

关闭错误输出

在 `wp-config.php` 中加入如下代码即可：

```
define( 'WP_DEBUG_DISPLAY', false );
define( 'WP_DEBUG_LOG', true );
```

使用 SSL

给站点加入 HTTPS，并设置强制SSL，在 `wp-config.php` 中加入如下代码即可：

```
define('FORCE_SSL_ADMIN', true);
```

修改数据库默认前缀 wp_

在安装时，修改数据库的默认前缀，不要使用 `wp_`，如果已经安装好了，可以试试 [Change Table Prefix](#) 插件。

修改默认的用户名 Admin

在安装时，不要使用 Admin 作为用户名（如果已经使用，可以登录到 phpMyAdmin 去修改），后台个人资料中的“公开显示为”也要使用昵称，而不是用户名。

使用邮箱登陆

相比于更短的用户名，用一个更长的邮件地址作为登录名会更加的安全。

可以通过安装一个插件来实现[Email Login](#)

隐藏 WordPress 版本信息

在主题的 `functions.php` 中添加如下代码，可以关闭 `head` 部分中的版本输出，至少可以让黑客不知道你使用的是哪个版本的 WP，会更加安全一些。

```
/* Hide WP version strings from scripts and styles
 * @return {string} $src
 * @filter script_loader_src
 * @filter style_loader_src
 */
function fjarrett_remove_wp_version_strings( $src ) {
```

```

global $wp_version;
parse_str(parse_url($src, PHP_URL_QUERY), $query);
if ( !empty($query['ver']) && $query['ver'] === $wp_version ) {
$src = remove_query_arg('ver', $src);
}
return $src;
}
add_filter( 'script_loader_src', 'fjarrett_remove_wp_version_strings' );
add_filter( 'style_loader_src', 'fjarrett_remove_wp_version_strings' );

/* Hide WP version strings from generator meta tag */
add_filter( 'the_generator', '__return_empty_string' );

```

经常关注安全新闻和更新

经常关注最新的安全新闻和漏洞数据库，比如可以订阅 WordPress 官方的[安全新闻](#)，或者是[WPVulnDB](#)

确保本地环境的安全

确保电脑、浏览器、路由器使用的是最新的、没有恶意代码的软件。如果使用的是公共 WiFi 热点，最好对通信过程进行加密。如果使用手机、平板电脑管理你的站点，同样也要保持更新。

使用 SFTP 替代 FTP

SFTP 使用加密传输认证信息和传输数据，相对来说会更安全。所以尽可能使用安全度更高的 SFTP 来传输数据。

使用复杂密码，经常更换密码

密码的目标就是让其他人难以猜测，并且很难被暴力破解，其中的关键就是让其变得复杂、无序。建议对所有密码都使用密码生成器。

默认情况下，WordPress 会提供一个密码生成器，当更改密码时，会自动生成一个高强度密码。如果记不住这个密码，可以考虑使用诸如 1Password、LastPass 这样的服务来存储密码。同时，还可以安装 `Force Strong Passwords` 插件来强制你必须使用更加复杂的密码。

为程序引入两步验证（Two Step Authentication）

使用两步验证能够让站点更加的安全，我们在前面的常用插件中已经加入了 Google Authenticator 的使用说明，就不再重复安装过程了。如果需要接入，可以回到第 5 课去看具体的内容。

安装安全插件处理安全问题

BulletProof Security

这个插件会提供 防火墙、登录保护、数据库备份等一些功能。如果有问题还可以通过电子邮件通知处理，还可以阻止一些 IP 的访问。

Login LockDown

这个插件可以封禁掉想要暴力破解你网站的 IP 地址。

关闭掉 XML-RPC

在 .htaccess 中添加如下代码，可以封掉 XML-RPC 请求：

```

<Files xmlrpc.php>
order deny,allow

```

```
deny from all
allow from 123.123.123.123
</Files>
```

对于大多数用户来说，用不到 XML-RPC，可以使用上述代码关掉该功能。此外，如果需要使用，只要用你自己的 IP 替换 123.123.123.123 即可。

关掉 JSON Rest API

在主题的 functions.php 中添加如下代码，可以关闭 Rest API 输出。

```
add_filter('json_enabled', '__return_false');
add_filter('json_jsonp_enabled', '__return_false');
```

大部分人是用不到 WordPress 的 Rest API 的，所以可以在主题的 functions.php 中添加上述代码，关闭掉 REST API。

封禁垃圾请求

在 .htaccess 中添加如下代码，可以封掉一些毫无意义的请求：

```
<ifModule mod_rewrite.c>
RewriteCond %{QUERY_STRING} enter|separated|query|strings|here [NC]
RewriteRule .* http://www.%{HTTP_HOST}/$1? [R=301,L]
</ifModule>
```

禁止机器人的请求

在 .htaccess 中添加如下代码，可以封掉机器人的请求：

```
<IfModule mod_rewrite.c>
RewriteCond %{REQUEST_METHOD} POST
RewriteCond %{REQUEST_URI} wp-comments-post\.php\*
RewriteCond %{HTTP_REFERER} !.wordpress.dev.* [OR]
RewriteCond %{HTTP_USER_AGENT} ^$
RewriteRule (.*) ^http://.%{REMOTE_ADDR}/$ [R=301,L]
</IfModule>
```

禁止包含 SQL 查询的请求

在 .htaccess 中添加如下代码，可以封掉包含 SQL 查询的请求：

```
<IfModule mod_rewrite.c>
RewriteBase /
RewriteCond %{REQUEST_METHOD} ^(HEAD|TRACE|DELETE|TRACK) [NC]
RewriteRule ^(.*)$ - [F,L]
RewriteCond %{QUERY_STRING} \.\.\// [NC,OR]
RewriteCond %{QUERY_STRING} boot\.ini [NC,OR]
RewriteCond %{QUERY_STRING} tag\= [NC,OR]
RewriteCond %{QUERY_STRING} ftp\: [NC,OR]
RewriteCond %{QUERY_STRING} http\: [NC,OR]
RewriteCond %{QUERY_STRING} https\: [NC,OR]
RewriteCond %{QUERY_STRING} (\|\%3E) [NC,OR]
RewriteCond %{QUERY_STRING} mosConfig_[a-zA-Z_]{1,21}(=|\%3D) [NC,OR]
RewriteCond %{QUERY_STRING} base64_encode.*\(.*\) [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(\[|\]|\(|\)|é|"|;|\?|\^|=|$).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*("'|<|>|\||{}|).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(%24&x).* [NC,OR]
```

```
RewriteCond %{QUERY_STRING} ^.*(%0|%A|%B|%C|%D|%E|%F|127\.\d).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(globals|encode|localhost|loopback).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(request|select|insert|union|declare).* [NC]
RewriteCond %{HTTP_COOKIE} !^.*wordpress_logged_in_.*$
RewriteRule ^(.*)$ - [F,L]
</IfModule>
```

推荐阅读

如果觉得还不够，建议去读一读道哥（吴翰清）的[《白帽子讲 Web 安全》](#)。

WordPress 主题的使用和管理

对于很多新手来说，选择 WordPress 的一个很重要的原因便是它丰富的主题，海量的主题让我们有充足的选择。对于新手来说，无疑是十分友好的，无需编写一行代码，就可以任选主题。

因此，WordPress 也成就了非常多的外包团队，他们使用 WordPress 为企业、个人建立网站，帮助企业和个人成功。

WordPress 后台主题管理

切换主题

在我们安装完成了 WordPress 后，WordPress 默认提供三款主题，比如目前提供的是 Twenty Seventeen、Twenty Sixteen，可以在后台直接切换主题来使用。

从仪表盘中验证「外观」—「主题」。

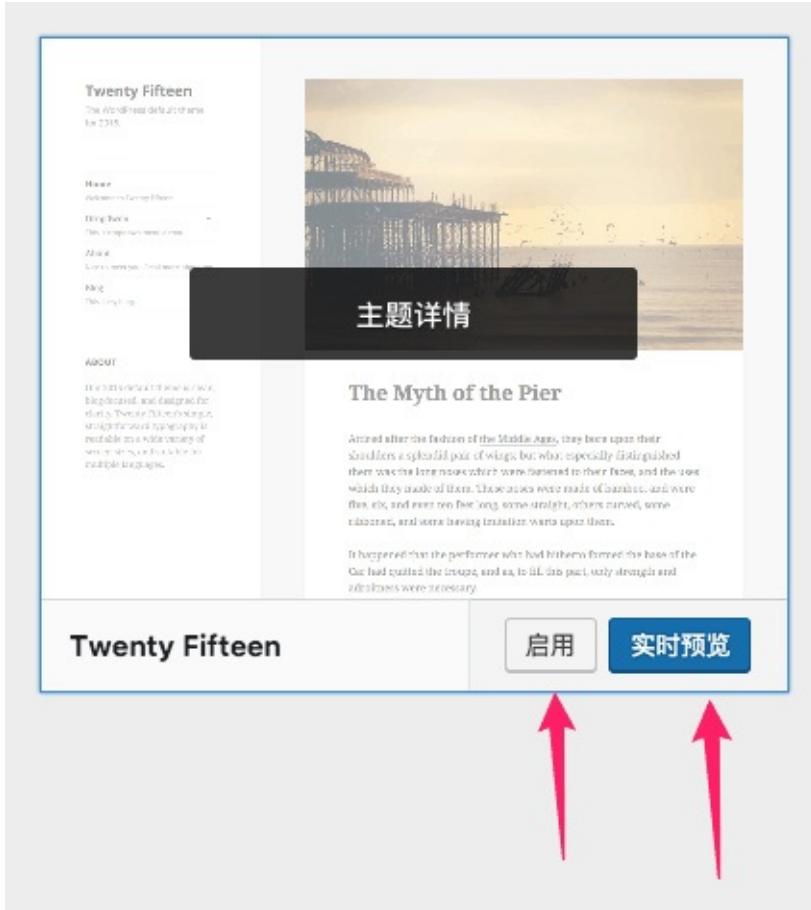


在主题页面，我们可以看到系统默认安装的三个主题，其中第一个是我们正在使用的主题。点击这个主题，就能够看到主题详情。

The screenshot shows the 'Twenty Seventeen' theme details page. At the top right, there's a button labeled '当前主题' (Current Theme). Below it, the theme name 'Twenty Seventeen' and version '1.4' are displayed, along with the text '由WordPress团队' (Developed by the WordPress team). A large thumbnail image of the theme is shown, featuring a potted plant on a wooden surface. To the right of the thumbnail, there's a detailed description of the theme's features, including its ability to support multiple page layouts, flexible headers, and various customization options. At the bottom of the page, there's a footer with links to 'Home', 'About Us', 'Blog', and 'Contact', and a note about the theme being '由WordPress进行创作' (Created by WordPress).

在另外两个主题上，你可以执行启用主题，或者实时预览主题。可以在启用之前先进行预览，看看这个主题应用在你的内容上是否合适。预览完成后，可以根据预览结果选择是否要启用。

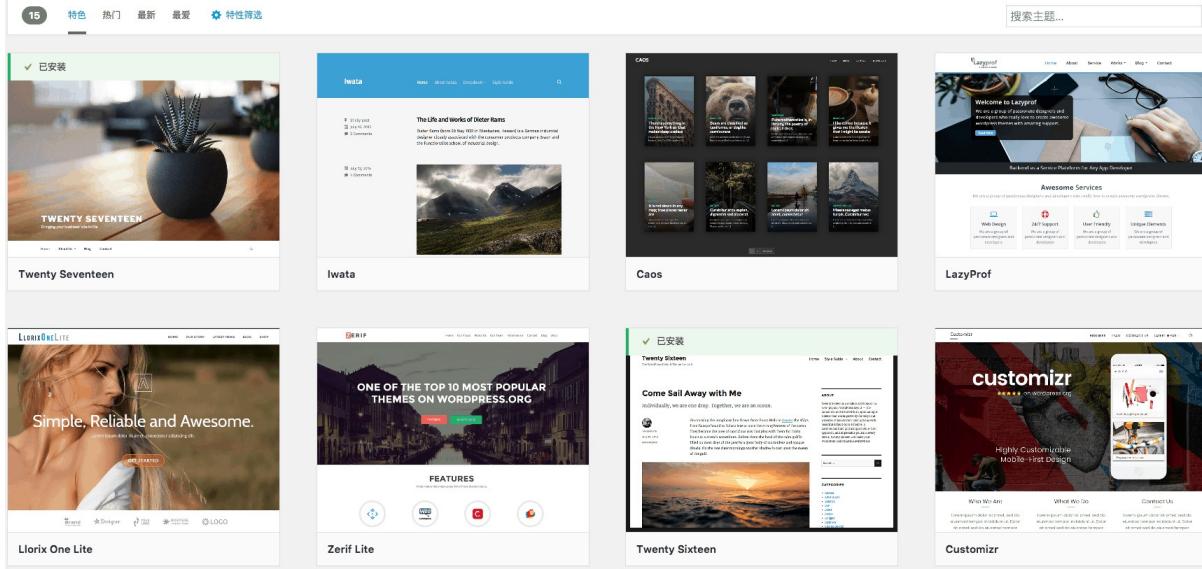
预览的界面中可以对主题进行修改、设置。设置完成后，直接启用，就可以以你修改的结果启用主题了。这里的界面和后续主题使用部分的「自定义界面一致」，我们放在那里来讲。



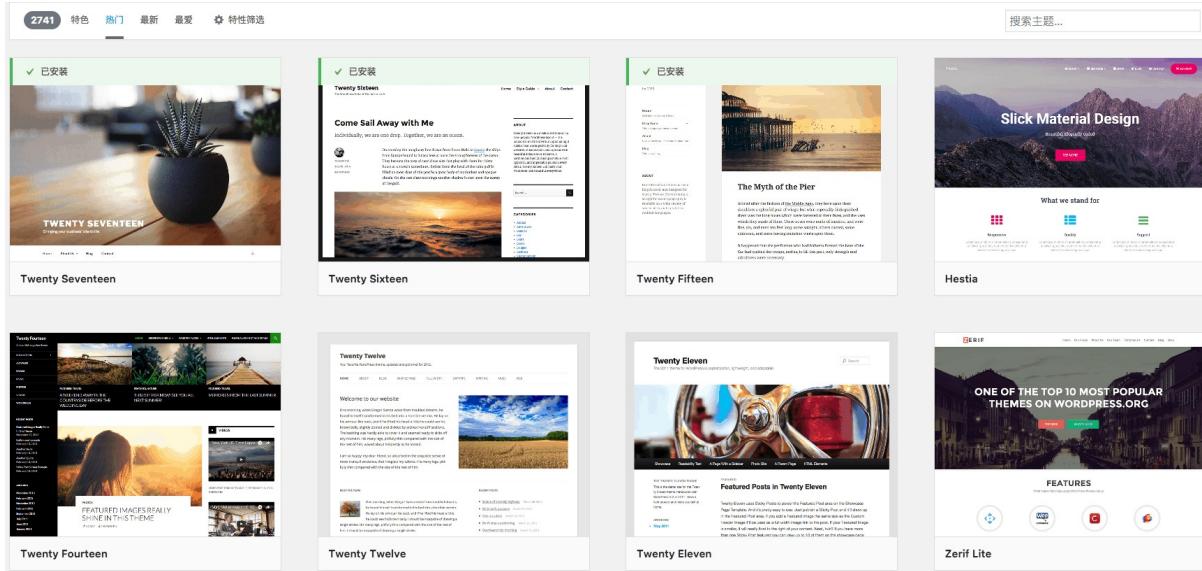
安装主题

安装主题分为三种方法，在主题中心安装、在后台上传和在FTP中上传。这里只讲前两者，第三种方法和插件的上传方法基本一致，只是目录从 `wp-content/plugins` 变为 `wp-content/themes`。

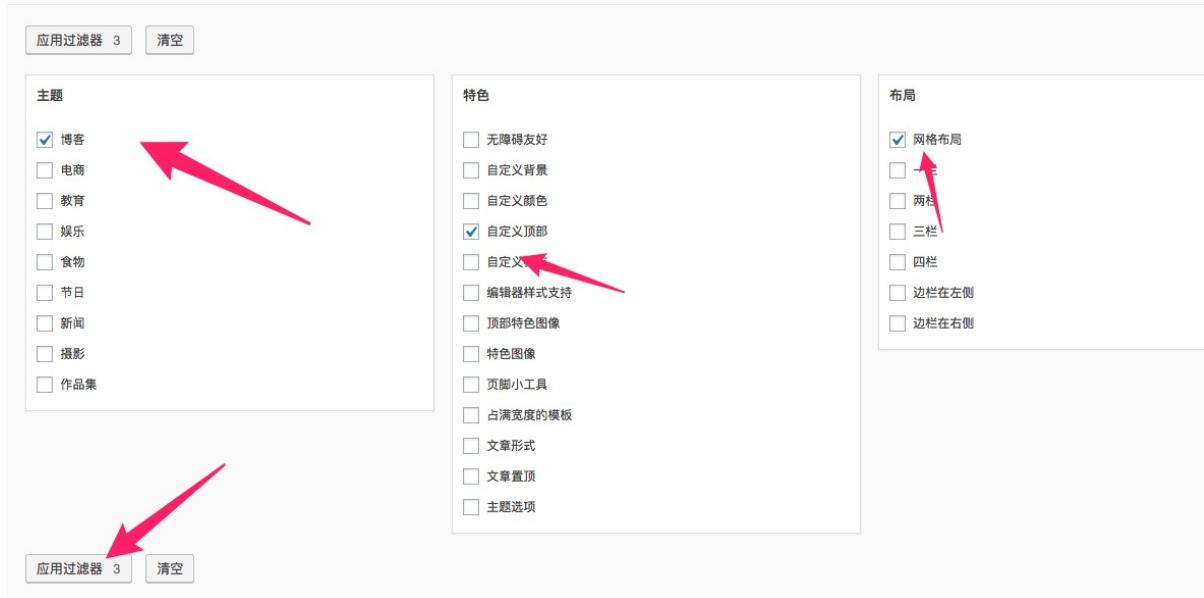
安装主题我们可以在主题界面点击「添加」，进入到主题添加界面。



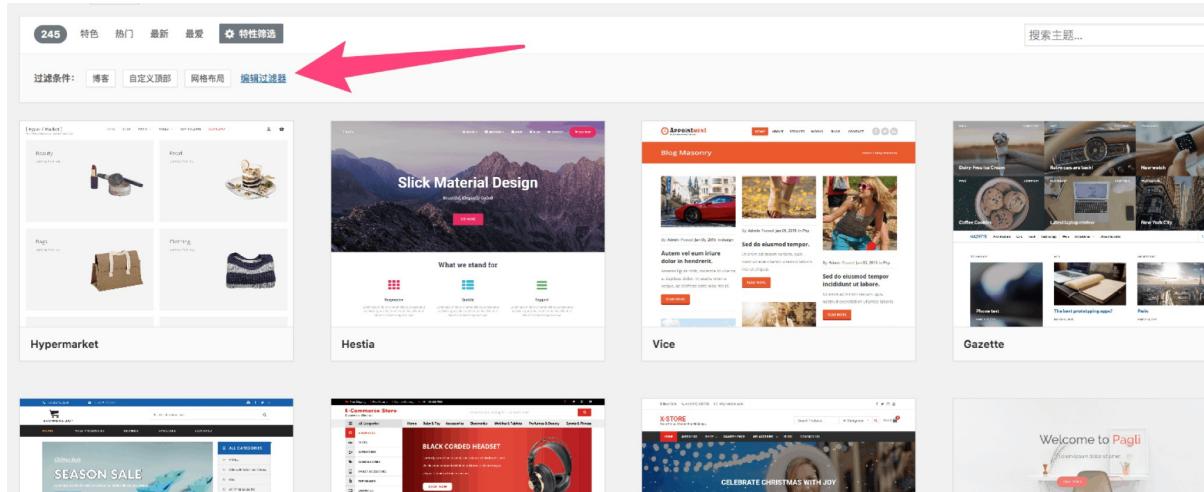
在主题中心可以看到一些特色主题，你可以先查看特色主题中是否有你喜欢的主题。如果没有，可以单击上方的「热门」，查看最近使用的比较多的主题，或切换到最新，查看最新发布的主题。



如果你不喜欢查看别人选出来的主题，也可以自己来选择主题。单击上方的「筛选」，进入到筛选页面，选择你需要的主题的特性，然后点击应用过滤器，就可以自动筛选符合我们的条件的主题了。



如果筛选结果不满意，可以点击上方的编辑过滤器，再次勾选条件，筛选需要的主题。



卸载主题

在前面的课程中曾说过，安装的主题多了，可能会导致 WordPress 的速度变慢，我们需要根据使用情况，删除不用或不太可能再用的主题。

从主题列表中的未启用主题中选择要删除的主题，点击主题，进入主题的详情页，在详情页的右下角，可以看到主题的删除按钮，点击删除按钮，即可删除主题：



WordPress 主题使用

主题自定义

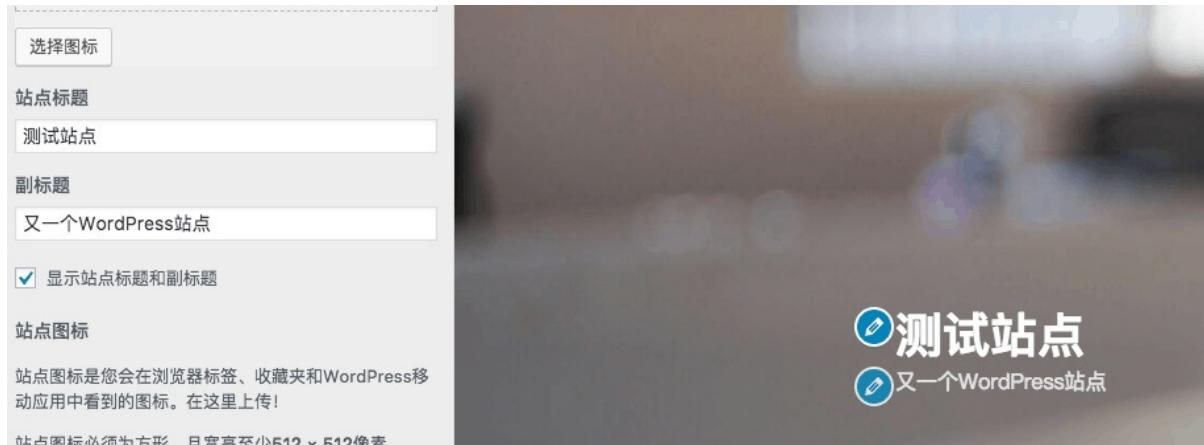
我们的主题可能有很多不同的设置项，这些设置项可以通过主题的「自定义」功能来修改。这里以 Twenty SevenTeen 为例，来说明主题自定义的用法。

点击仪表盘中的「外观」—「自定义」，就可以进入到自定义界面了。

在自定义界面，会看到一些蓝色的按钮，点击这些按钮，左侧的编辑框内就会出现对应的设置项。

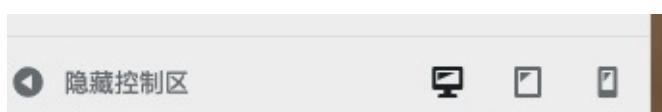


修改左侧的内容，右侧对应会实时显示出来，可以方便查看具体的效果。



其他的设置项也大多相同。

在控制区域的底部，有三个按钮，分别代表着电脑、平板和手机，你点击下方的设备，就可以将当前网页的大小变为对应设置的预览效果，来查看主题能否在多种屏幕大小下正常工作。



这里说一个大家平时可能没有注意到的设置。在修改完成后，在右上角的发布按钮这里，是有一个齿轮的，点击这个齿轮，会弹出新的设置框，在这个设置框中：



我们可以选择发布（立刻生效）、保存草稿（留着以后发布）和计划（在特定时间发布，定期上线）。有了这个功能，在进行站点修改时，就无需一定要当时修改，可以提前修改好，定时发布即可。

此外，WordPress 还提供了一个预览的功能。将操作切换为「保存草稿」，然后保存一次草稿，在下方会生成一个预览链接，可以将这个链接复制给其他人，让其他人查看你的修改是否正确等等。



当我们不在一起办公时，可以借助这个功能，很方便的让合作者看到站点修改的状况，确认修改是否正确。

预览也是使用的自定义界面，唯一不同的是无法保存草稿和定时发布。

小工具设置

有了小工具，我们的站点侧边栏就会变得生动起来，可以根据需要来设置侧边栏。

点击仪表盘中的「外观」—「小工具」，就可以进入到小工具界面了。



小工具界面可以分为两个部分，分别是控件区域和边栏区域，可以将我们要使用的控件拖动到边栏中去。



或者单击要添加的小工具，选择要添加到哪个边栏中去，点击添加小工具，就可以将这个小工具添加到对应的区域中去了。

添加好的小工具可能需要设置一些属性，可以在边栏中设置他们的属性，以达到自定义的效果。



当一个控件区域有多个控件时，可以通过拖拽，来对控件进行排序。



当有一些控件你暂时不使用，同时还要保留其中的设置时，可以用好「未使用的小工具」，将这个要保留的小工具拖拽到底部的「未使用的小工具」区域。放在这个区域的小工具的设置是不会丢失的。



后续需要使用时，再从这里拖拽回边栏区域中即可。如果这里面的控件都不再使用了，则可以点击下方的「清理未启用的小工具」，这个按钮会清空所有未启用的小工具。

菜单设置

对于一些支持自定义菜单的主题，可以很方便的管理我们的菜单。

从仪表盘中验证「外观」—「菜单」，就可以进入到菜单管理的界面了。

概念

在开始使用之前，需要先理解两个概念：菜单和位置。

菜单：菜单是一组链接的集合，可以准备多个不同的菜单，将不同的菜单放在不同的位置上，也可以一个菜单放在多个位置上。但是，同一个位置上只能放一个菜单。

位置：位置是你所安装的 WordPress 主题提前预留给你用来放置菜单的位置。有几个位置取决于你使用的主题留了多少个用于放置菜单的接口。

使用



我们第一次进入菜单界面时，界面中是没有任何菜单的，首先，要创建一个菜单，输入菜单名称，点击「创建」菜单，就可以创建一个新的菜单了。

创建了一个菜单后，就可以向菜单中添加链接了。WordPress 提供了页面、文章和分类目录的快速添加的功能。在左侧找到你要添加的页面，点击添加到菜单，就可以将其加入到我们的菜单中去。



需要注意的是，其实 WordPress 也支持添加标签和文章形式，但是默认没有展示出来，可以点击页面右上角的「显示选项」，勾选上「标签」和「形式」，这样在左侧的列表中就会多出对应的两项。

如果要添加的链接不在这几项里，或者是要添加的内容是别的网站的链接，你可以切换到自定义链接项到这里，通过添加链接项，来添加菜单项。



添加以后，你可以修改导航标签，来修改该项在菜单中最终显示的文字。

菜单结构

拖放各个项目到您喜欢的顺序，点击右侧的箭头可进行更详细的设置。



在下方的菜单设置中，可以选择我们的菜单的显示位置。勾选要在哪个位置上使用这个菜单，然后保存设置即可使这个菜单的设置应用到对应的位置上。

菜单设置

自动添加页面

自动添加新的顶级页面到此菜单

显示位置

顶部菜单

社交网络链接菜单

自动添加页面这个选项如果勾选后，可以在创建了一个新的页面时，自动将这个页面添加到菜单中。不过一般情况下，我们都不会勾选这个选项。

除了管理菜单，还可以管理位置，点击上方的「管理位置」，可以进入到管理位置的页面。

在这个页面，可以切换每个位置上使用的菜单，而无需切换到一个个具体的菜单中操作。

顶部设置

在顶部设置中，可以修改页面顶部的大图，这部分的使用和自定义区域的使用基本一致，不再赘述。

编辑主题

有些时候，我们可能需要修改主题中的代码，比如添加统计代码、添加一些平台的验证信息，此时打开 FTP 工具可能太过麻烦，如果要修改的信息是添加在主题中的，可以通过编辑功能来修改，这里的使用基本和插件的编辑部分是一致的，也就不再赘述了。

总结

至此，我们学习了如何使用和设置 WordPress 主题，从下节课开始，就尝试来开发一个我们自己的 WordPress 主题。

WordPress 主题开发快速入门

课程目标

经过本次课程可以学会制作一个简单的 WordPress 主题。

主题效果图：



The Bootstrap Blog

The official example template of creating a blog with Bootstrap.

Sample blog post

January 1, 2014 by [Mark](#)

This blog post shows a few different types of content that's supported and styled with Bootstrap. Basic typography, images, and code are all supported.

Cum sociis natoque penatibus et magnis **dis parturient montes**, nascetur ridiculus mus. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum. Sed posuere consectetur est at lobortis. Cras mattis consectetur purus sit amet fermentum.

Curabitur blandit tempus porttitor. **Nullam quis risus eget urna mollis** ornare vel eu leo. Nullam id dolor id nibh ultricies vehicula ut id elit.

Etiam porta *sem malesuada magna* mollis euismod. Cras mattis consectetur *nurus* sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

About

Etiam porta sem malesuada magna mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Archives

[March 2014](#)
[February 2014](#)
[January 2014](#)
[December 2013](#)
[November 2013](#)
[October 2013](#)
[September 2013](#)
[August 2013](#)
[July 2013](#)
[June 2013](#)
[May 2013](#)
[April 2013](#)

课程内容

课程资源请[单击这里](#)。

创建一个主题

在课程的开始首先应该创建一个主题。

打开 WordPress 源码文件夹，进入到 `wp-content/themes` 目录中，创建一个名为 `gitchat` 的目录。

接下来下载课程资源中的 `blog-template.zip`，解压后会看到一个 `index.html` 和一个 `blog.css`。

将 `index.html` 改为 `index.php`，将 `blog.css` 改为 `style.css`，将两个文件移动到刚刚创建的 `gitchat` 目录中去。

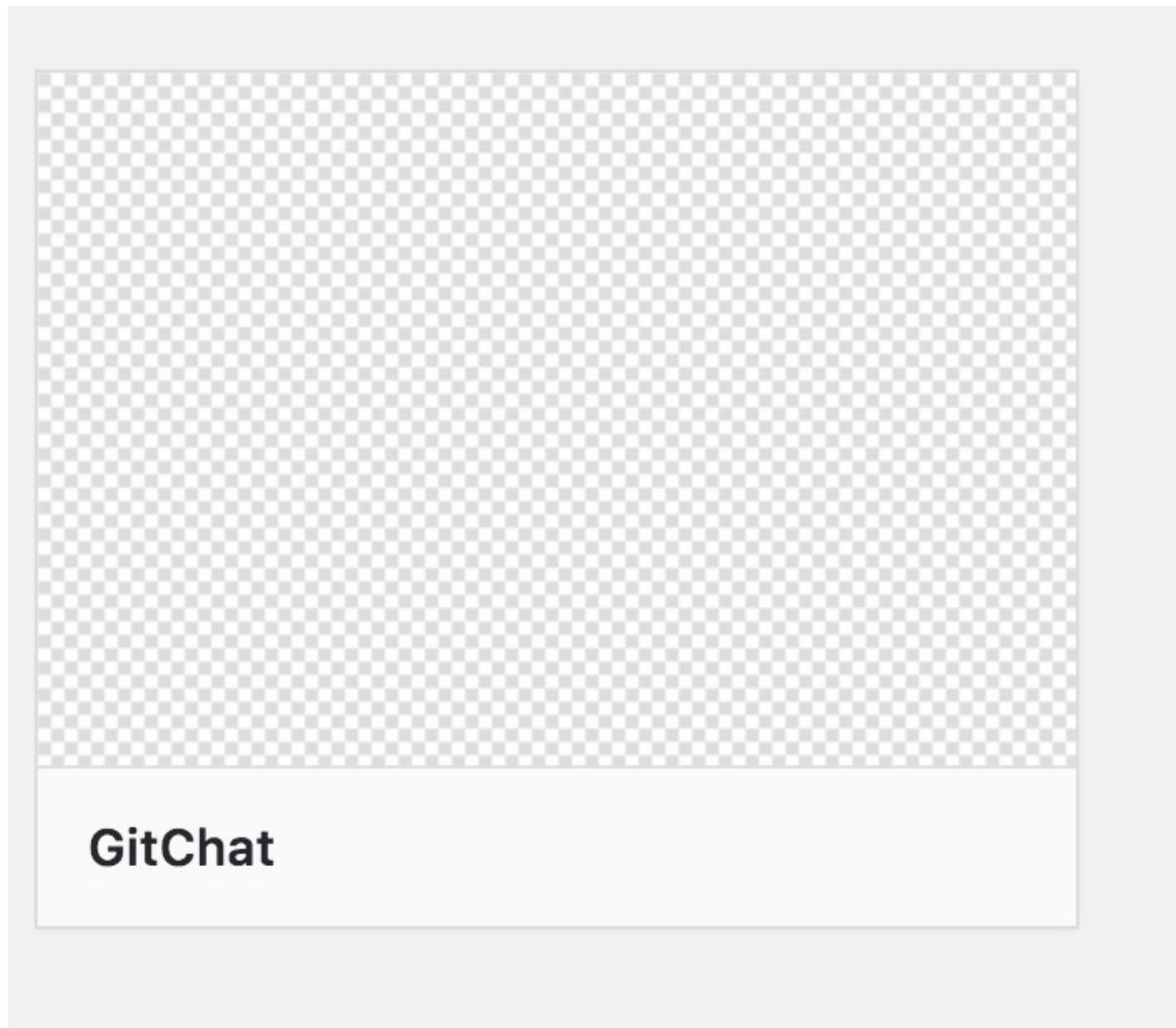
接下来打开 `style.css`，在文件的顶部加入如下代码。

```
/*
Theme Name: GitChat
Theme URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Author: 白宦成
Author URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Description: GitChat WordPress 演示达人课
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Tags: blue
Text Domain: gitchat
```

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.  
*/
```

然后，打开 WordPress 后台的主题管理页面，就可以看到主题了。



你可以尝试启用这个主题，会发现主题已经能够正常的启用了，不过当点击进入前台时，发现目前的页面效果并不好，和我们的效果图差距很大。

这是由于我们将 `index.html` 未经修改直接作为主题，引用的文件都失效了导致的。

[Home](#) [New features](#) [Press](#) [New hires](#) [About](#)

The Bootstrap Blog

The official example template of creating a blog with Bootstrap.

Sample blog post

January 1, 2014 by [Mark](#)

This blog post shows a few different types of content that's supported and styled with Bootstrap. Basic typography, images, and code are all supported.

Cum sociis natoque penatibus et magnis [dis parturient montes](#), nascetur ridiculus mus. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum. Sed posuere consectetur est at lobortis. Cras mattis consectetur purus sit amet fermentum.

Curabitur blandit tempus porttitor. **Nullam quis risus eget urna mollis ornare vel eu leo.** Nullam id dolor id nibh ultricies vehicula ut id elit.

Etiam porta sem malesuada magna mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Heading

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros.

Sub-headings

接下来修复这个问题，让这个页面的样式变为和之前一样的样式。

打开 `index.php` 文件，找到第18行：

17
18

```
<!-- Custom styles for this template -->
<link href="blog.css" rel="stylesheet">
```

将这里的 `blog.css` 替换为 `<?php echo get_stylesheet_directory_uri() ?>/style.css`，保存文件并刷新网站首页，可以看到，首页的样式就回复了过来。

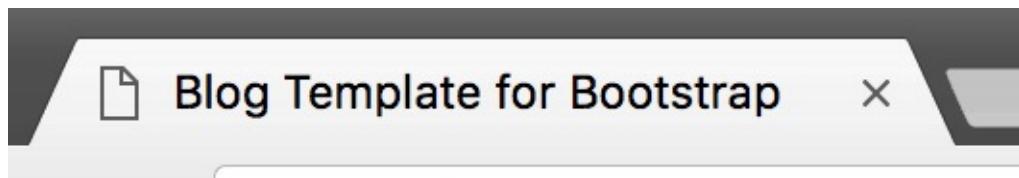
这里除了使用这个代码，还可以使用：

```
<?php echo get_bloginfo('template_directory'); ?> /style.css
```

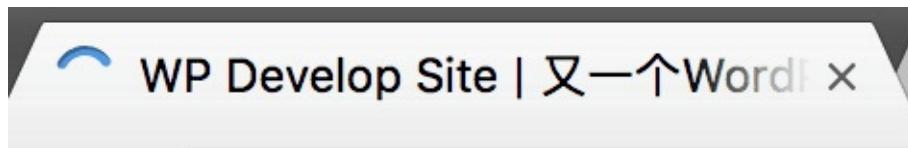
这段代码，WordPress 针对同一个实现的效果可能会有多种不同的代码，不过并不重要，我们记住几个常用的即可。

完善顶部内容

我们发现，现在页面的标题是错误的，接下来修改一下：



重新回到文件编辑器中，打开 `index.php`，找到第12行，将这里的 `Blog Template for Bootstrap` 替换为 `<?php wp_title(' ', true, 'right'); ?>`。这样，就使用了 WordPress 自带的函数来生成页面的标题，确保了页面标题是自动生成的。



在别的主题中，我们会在前台页面的顶部看到一个菜单栏：



但是在我们现在的主题中是无法看到的，接下来修复这个问题，让 WordPress 主题能够正常的加载这个菜单栏。

重新回到编辑器，打开 `index.php`，找到 `head` 标签，在 `head` 标签闭合前，加入如下代码：

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

具体位置如下：

```

20      <!-- HTML5 shim and Respond.js for IE8 s
21      <![if lt IE 9]>
22          <script src="https://oss.maxcdn.com/ht
23          <script src="https://oss.maxcdn.com/re
24          <![endif]-->
25      <?php wp_head();?>
26      </head>
27
28      <body>

```

然后找到页面的底部，在 `Body` 标签闭合前，加入如下代码：

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

具体位置如下：

```

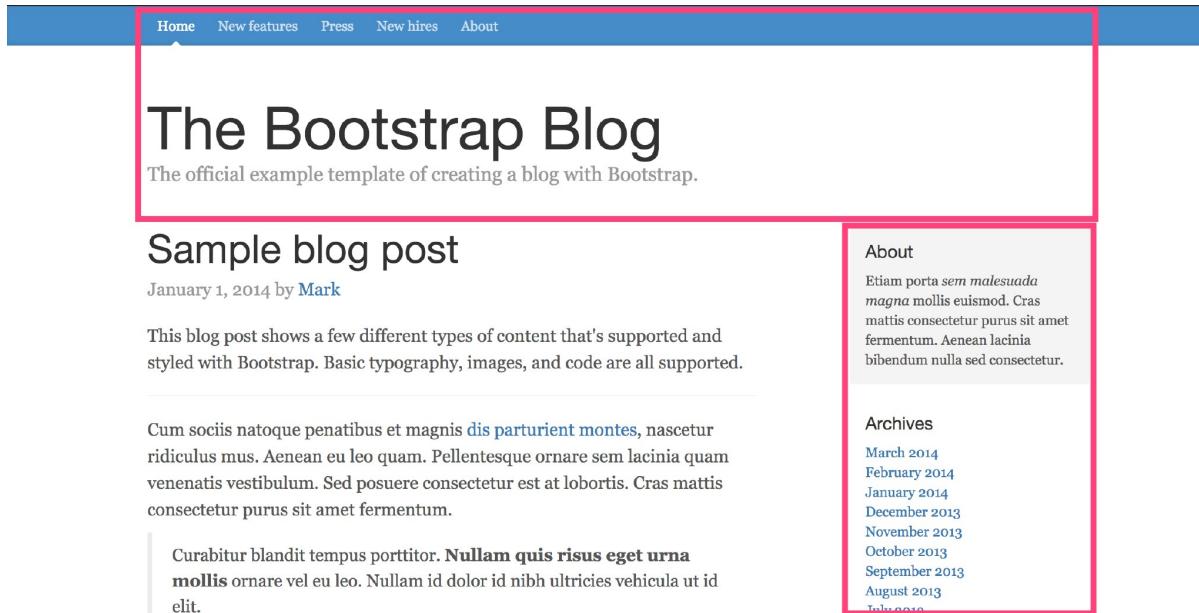
166      =====
167      <!-- Placed at the end of the document so
168      <script src="https://ajax.googleapis.com/
169      <script src="https://maxcdn.bootstrapcdn.com.
170      <?php wp_footer(); ?>
171      </body>
172      </html>
173

```

现在再刷新页面就可以看到我们的菜单栏了。

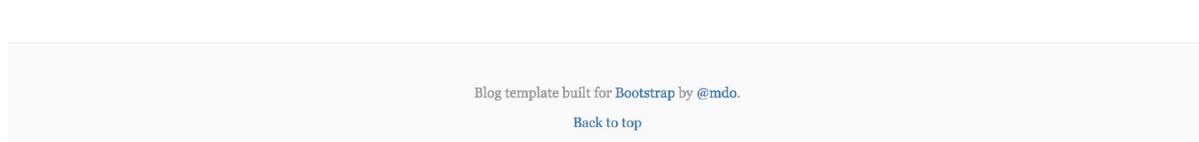
拆分出 header.php、footer.php、Sidebar.php

在接下来的内容中，我们要制作独立的单页等，所以需要在这里将页面进行拆分，将通用的部分抽离出去，从而提升代码的复用率。



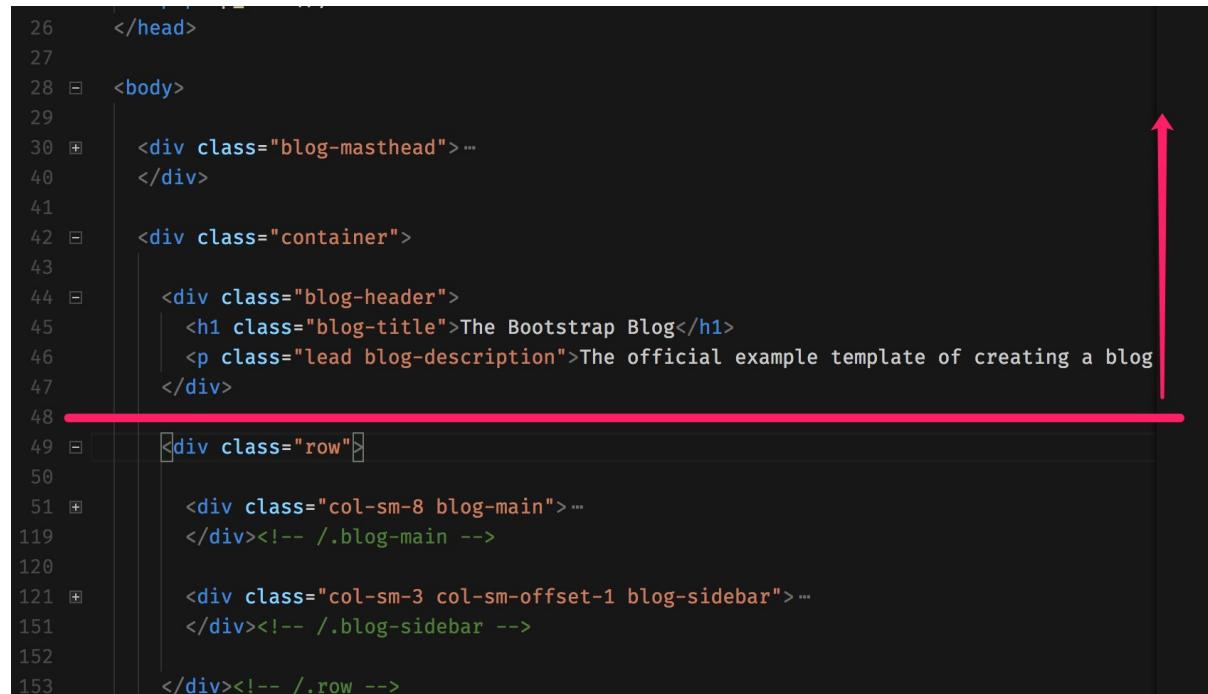
可以认为，在这个页面中，被我用红色框体圈出来的内容是会被多个文件所使用的，我们可以先将这两部分拆分出来。

此外，页面底部的版权声明也是通用的，也可以将其拆分出来：



header.php

打开编辑器，在主题目录下创建一个 `header.php` 文件，然后，打开 `index.php`，找到博客标题。



```

26     </head>
27
28     <body>
29
30     <div class="blog-masthead">...
31     </div>
32
33     <div class="container">
34
35         <div class="blog-header">
36             <h1 class="blog-title">The Bootstrap Blog</h1>
37             <p class="lead blog-description">The official example template of creating a blog
38         </div>
39
40
41         <div class="row">
42             <div class="col-sm-8 blog-main">...
43             </div><!-- /.blog-main -->
44
45             <div class="col-sm-3 col-sm-offset-1 blog-sidebar">...
46             </div><!-- /.blog-sidebar -->
47
48         </div><!-- /.row -->
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126

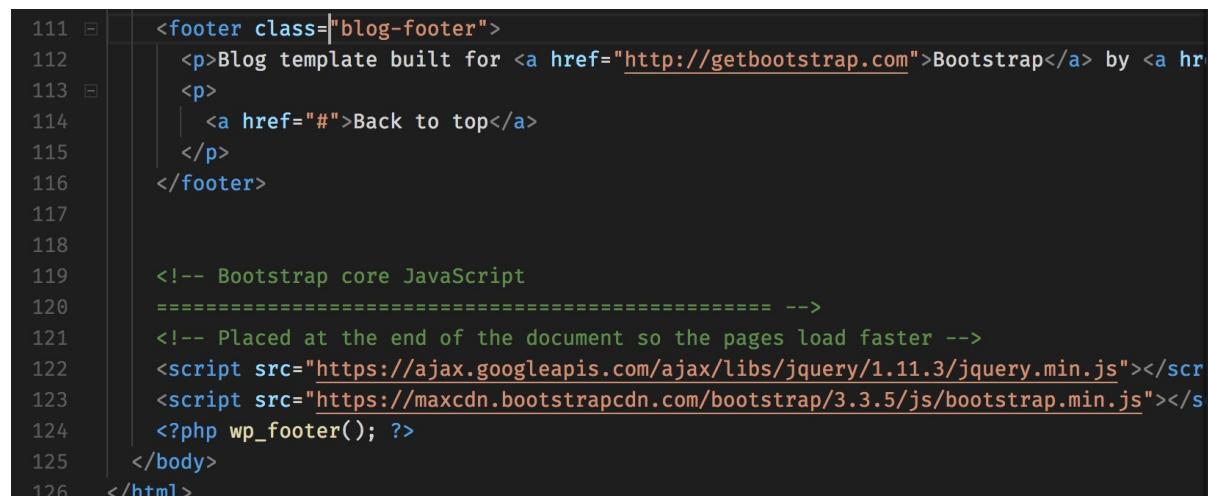
```

则博客标题以上的内容，都是可以被我们所复用的内容。我们将上方的所有代码，复制到刚刚创建的 `header.php` 中。接下来，删除 `index.php` 中复制过来的代码，并在这些代码的位置上插入一句话 `<?php get_header();?>`。刷新首页，会发现页面依然是老样子，没有发生任何变化。

`get_header()` 函数能够引用同一目录下的 `header.php` 文件，从而帮助我们提升代码的复用率。

footer.php

同样的，我们在代码中查找版权信息，可以看到这段代码在底部。由于网页的渲染是从上到下，而这段代码后再也没有代码了，所以我们可以创建一个新的 `footer.php` 文件，并将这些代码加入其中。



```

111     <footer class="blog-footer">
112         <p>Blog template built for <a href="http://getbootstrap.com">Bootstrap</a> by <a href="http://
113             <p>
114                 <a href="#">Back to top</a>
115             </p>
116         </footer>
117
118
119         <!-- Bootstrap core JavaScript
120             =====
121             -->
122             <!-- Placed at the end of the document so the pages load faster -->
123             <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></scr
124             <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/js/bootstrap.min.js"></s
125             <?php wp_footer(); ?>
126         </body>
127     </html>

```

同样的，创建一个 `footer.php` 文件，然后将代码复制到 `footer.php` 文件，并在 `index.php` 中这段代码所在的位置上加入一句代码 `<?php get_footer();?>`。保存文件并刷新页面，可以看到，我们底部的版权链接也依然还在。

`get_footer` 方法和 `get_header` 方法一样，都是帮助我们引用文件的。

sidebar.php

通过分析代码，我们可以知道，侧边栏的代码是下面这一串

```
37
38     <div class="col-sm-3 col-sm-offset-1 blog-sidebar">
39         <div class="sidebar-module sidebar-module-inset">
40             <h4>About</h4>
41             <p>Etiam porta <em>sem malesuada magna</em> mollis euismod. Cras i
42         </div>
43         <div class="sidebar-module">
44             <h4>Archives</h4>
45             <ol class="list-unstyled">
46                 <li><a href="#">March 2014</a></li>
47                 <li><a href="#">February 2014</a></li>
48                 <li><a href="#">January 2014</a></li>
49                 <li><a href="#">December 2013</a></li>
50                 <li><a href="#">November 2013</a></li>
51                 <li><a href="#">October 2013</a></li>
52                 <li><a href="#">September 2013</a></li>
53                 <li><a href="#">August 2013</a></li>
54                 <li><a href="#">July 2013</a></li>
55                 <li><a href="#">June 2013</a></li>
56                 <li><a href="#">May 2013</a></li>
57                 <li><a href="#">April 2013</a></li>
58             </ol>
59         </div>
60         <div class="sidebar-module">
61             <h4>Elsewhere</h4>
62             <ol class="list-unstyled">
63                 <li><a href="#">GitHub</a></li>
64                 <li><a href="#">Twitter</a></li>
65                 <li><a href="#">Facebook</a></li>
66             </ol>
67         </div>
68     </div><!-- /.blog-sidebar -->
```

我们将这一串代码复制出来，新建一个 `sidebar.php` 文件，并将代码粘贴进去，然后使用 `get_sidebar` 方法加载侧边栏。

经过这段代码的替换，我们的 `index.php` 变成了这个样子，可以看到我们的整个文件只有45行，这使得我们的文件变得更加简单易读。

```
1  <?php get_header();?>
2
3  <div class="row">
4
5      <div class="col-sm-8 blog-main">
6
7          <div class="blog-post">...
8          </div><!-- /.blog-post -->
9
10         <div class="blog-post">...
11         </div><!-- /.blog-post -->
12
13         <div class="blog-post">...
14         </div><!-- /.blog-post -->
15
16         <div class="blog-post">...
17         </div><!-- /.blog-post -->
18
19         <nav>...
20         </nav>
21
22         </div><!-- /.blog-main -->
23
24
25
26
27
28
29         <nav>...
30         </nav>
31
32
33
34
35
36         </div><!-- /.blog-main -->
37
38
39         <?php get_sidebar();?>
40
41         </div><!-- /.row -->
42
43
44         <?php get_footer();?>
45
```

WordPress 主循环

页面拆分完毕了，但是页面的主题部分还是静态的内容，我们真正想要加载的内容还依然是静态内容。

接下来把主要的内容转换为动态的内容。这里我们使用 WordPress 官方提供的主循环代码来实现页面内容动态输出。

```
<?php if ( have_posts() ) : ?>
<?php while ( have_posts() ) : the_post(); ?>
    ... Display post content
<?php endwhile; ?>
<?php endif; ?>
```

首先，我们清理下页面的代码：

```
4
5         <div class="col-sm-8 blog-main">
6
7             <div class="blog-post">
8                 <h2 class="blog-post-title">标题</h2>
9                 <p class="blog-post-meta">日期 by <a href="#">作者</a></p>
10
11
12                 <p>摘要</p>
13             </div><!-- /.blog-post -->
14
15 +         <div class="blog-post">...
16             </div><!-- /.blog-post -->
17
18 +         <div class="blog-post">...
19             </div><!-- /.blog-post -->
20
21
22 +         <nav>...
23             </nav>
24
25
26         </div><!-- /.blog-main -->
```

将红色部分的内容删除，将蓝色部分的代码调整成为我图上的样子。

接下来，就是套用 WordPress 提供的主循环代码了，将调整的蓝色框的内容作为主循环的循环体，放入 主循环代码代码中。则变成这个样子。

```
6
7             <?php if ( have_posts() ) : ?>
8                 <?php while ( have_posts() ) : the_post(); ?>          主循环代码
9                     <div class="blog-post">
10                         <h2 class="blog-post-title">标题</h2>
11                         <p class="blog-post-meta">日期 by <a href="#">作者</a></p>
12                         <p>摘要</p>
13                     </div><!-- /.blog-post -->
14
15             <?php endwhile; ?>          主循环代码
16             <?php endif; ?>
```

保存文件，重新回到首页刷新，可以看到已经刷新出了文章。只不过我目前在数据库里只添加了两篇文章，所以这里循环出了两篇文章。

The Bootstrap Blog

The official example template of creating a blog with Bootstrap.

标题

日期 by 作者

文章1

摘要

标题

日期 by 作者

文章2

摘要

[Previous](#)

[Next](#)

About

Etiam porta sem malesuada magna mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Archives

[March 2014](#)
[February 2014](#)
[January 2014](#)
[December 2013](#)
[November 2013](#)
[October 2013](#)
[September 2013](#)
[August 2013](#)
[July 2013](#)
[June 2013](#)
[May 2013](#)
[April 2013](#)

Elsewhere

[GitHub](#)
[Twitter](#)
[Facebook](#)

现在虽然我们循环出了文章，但是内容依然对我们手写的，接下来把这里的内容也改成 WordPress 自动生成的。具体替换可以参考下方的替换规则。

替换规则：

标题 替换为 `<?php the_title();?>`；

摘要 替换为 `<?php the_excerpt();?>`；

作者 替换为 `<?php the_author();?>`；

作者标签的 href 的值替换为 `<?php the_author_link();?>`；

日期 替换为 `<?php the_date();?>`。

替换后的代码如下：

```

<?php if ( have_posts() ) : ?>
<?php while ( have_posts() ) : the_post(); ?>
<div class="blog-post">
    <h2 class="blog-post-title"><?php the_title();?></h2>
    <p class="blog-post-meta">
        <?php the_date();?> by
        <a href="php the_author_link();?&gt;"&gt;&lt;?php the_author();?&gt;&lt;/a&gt;
    &lt;/p&gt;
    &lt;p&gt;&lt;?php the_excerpt();?&gt;&lt;/p&gt;
&lt;/div&gt;&lt;!-- /.blog-post --&gt;
&lt;?php endwhile; ?&gt;
&lt;?php endif; ?&gt;
</pre

```

保存文件再刷新一下页面，可以看到页面变成了这个样子：

The Bootstrap Blog

The official example template of creating a blog with Bootstrap.

测试短代码

2017年12月21日 by admin

[git id="13" title="code"]测试文字[[...]]

世界，您好！

2017年12月14日 by admin

欢迎使用WordPress。这是您的第一篇文章。编辑或删除它，然后开始写作吧！

[Previous](#)

[Next](#)

我们的各项信息、内容都已经显示出来了。

但是现在有个问题，没有办法进入到文章的详情页面。我们继续修改，在标题外，再包裹一层 `a` 标签，并将其 `href` 设置为 `<?php the_permalink();?>`。

```

<div class="blog-post">
  | 
  <h2 class="blog-post-title"><?php the_title();?></h2>

  <p class="blog-post-meta">

```

↓

```

<div class="blog-post">
  | 
  <a href="<?php the_permalink();?>">
    <h2 class="blog-post-title"><?php the_title();?></h2>
  </a>
  <p class="blog-post-meta">

```

保存文件并刷新页面，我们将鼠标指针移动到标题上，会发现标题已经变为了可以点击的超链接，点击链接，就会进入到详情页。只不过，这里的详情页使用的依然是首页的代码，所以看起来会怪怪的。

这里涉及到了 WordPress 主题的文件结构，这部分内容将会在下一节课详细讲解，在这里可以不用太过关心，跟着课程一步步走下去，先试着做一个 WordPress 主题吧！

测试短代码 ←

2017年12月21日 by admin

[git id="13" title="code"] 测试文字 [...]

现在主循环已经完成了，页面也可以正常的加载内容了。

先在首页下方还有两个大按钮没有处理，接下来处理他们。

这两个按钮是用来进入上一页或者下一页的，好在 WordPress 提供了方便的函数，我们可以很轻松的完成这部分的修改。

使用下面的代码替换两个按钮的 `href`，顺便再将文字改为中文。

```

<?php next_posts_link( 'Older posts' ); ?>
<?php previous_posts_link( 'Newer posts' ); ?>

```

替换完成的代码如下：

```
<nav>
  <ul class="pager">
    <li><?php next_posts_link( '上一页' ); ?></li>
    <li><?php previous_posts_link( '下一页' ); ?></li>
  </ul>
</nav>
```

保存文件后，刷新，唉，按钮怎么不见了？这是因为目前站点中的文章不够，所以这两个按钮就没有生成。可以进入后台—「设置」—「阅读」，将 博客页面至多显示 改为 1。保存并刷新，可以看到首页的文章只显示了1条，同时我们可以点击上一页，浏览到上一条文章。

测试短代码

2017年12月21日 by admin

[git id="13" title="code"]测试文字[[...]]

上一页

至此，我们主循环就完成了。

处理文章页和单页

处理完了主循环，接下来美化一下文章详情页。

点击首页的文章标题，进入内容页会发现，这里与我们记忆中别的主题有很大差距，比如：

The Bootstrap Blog

The official example template of creating a blog with Bootstrap.

测试短代码

2017年12月21日 by admin

[git id="13" title="code"] 测试文字 [...]

不应该
应该是摘要
而应该是全文

Previous

Next

不应该有上一页
和下一页

竟然没有评论框？

接下来完善这些部分。

之前曾说过，我们之所以看到的不合适，是因为目前加载的内容是首页的模板，所以无法正常的显示。

首先创建一个 `single.php` 用于展示文章内容。将 `index.php` 内的代码复制到 `single.php` 中，稍后我们来修改。

复制完成后，打开 `single.php`，将页面中的 `the_excerpt()` 替换为 `the_content()`，并删除底部的导航代码，再移除文章标题上的链接。

```
<?php if ( have_posts() ) : ?>
<?php while ( have_posts() ) : the_post(); ?>
<div class="blog-post">
    <h2 class="blog-post-title"><?php the_title();?></h2>
    <p class="blog-post-meta">
        <?php the_date();?> by
        <a href="<?php the_author_link();?>"><?php the_author();?></a>
    </p>
    <p><?php the_content();?></p>
</div><!-- /.blog-post -->
<?php endwhile; ?>
<?php endif; ?>
```

现在再刷新会发现内容页的渲染结果和首页就不同了。标题上没有了链接，内容也显示完全了：

测试短代码

2017年12月21日 by admin

[git id="13" title="code"]测试文字[/code]

这段代码同样可以应用于单页，所以复制一下 `single.php`，重命名为 `page.php`。进入后台，在页面管理中找一个页面打开。可以看到，我们的页面的样式和文章页面保持一致：

示例页面

2017年12月14日 by admin

这是示范页面。页面和博客文章不同，它的位置是固定的，通常会在站点导航栏显示。很多用户都创建一个“关于”页面，向访客介绍自己。例如：

欢迎！我白天是个邮递员，晚上就是个有抱负的演员。
这是我的博客。我住在天朝的帝都，有条叫做Jack的狗。

.....或这个：

XYZ Doohickey公司成立于1971年，自从建立以来，我们一直向社会贡献着优秀doohickies。我们的公司总部位于天朝魔都，有着超过两千名员工，对魔都政府税收有着巨大贡献。

而您，作为一个WordPress用户，我们建议您访问[控制板](#)，删除本页面，然后添加您自己的页面。祝您使用愉快！

这样，我们的单页就显示完成了。

完善页头和菜单

一直看着顶部的菜单和标题不爽？没事，接下来修改这些内容。

打开 `header.php`，使用代码替换博客标题和博客描述。

```
<div class="blog-header">
  <h1 class="blog-title"><?php echo get_bloginfo( 'name' ); ?></h1>
  <p class="lead blog-description"><?php echo get_bloginfo( 'description' ); ?></p>
</div>
```

保存文件，刷新首页，可以看到，站点的标题已经替换为我们在「常规」中设置的内容了。

WP Develop Site

又一个WordPress站点

测试短代码

2017年12月21日 by admin

[git id="13" title="code"]测试文字[[...]

上一页

现在站点顶部的菜单还不是我们想要的，我们也替换一下代码。

在 `header.php` 中找到菜单的代码，将其替换为如下代码：

```
<div class="blog-masthead">
  <div class="container">
    <nav class="blog-nav">
      <a class="blog-nav-item active" href="/">Home</a>
      <?php wp_list_pages( '&title_li=' ); ?>
    </nav>
  </div>
</div>
```

这里我们将多余的链接删除，然后使用 `wp_list_pages` 函数来生成具体的链接。



可以看到，我们生成了对应的链接，但是样式出了些问题，不能很好的加载出来，我们添加一些样式来修复它。

打开 `style.css` 文件，在文件尾部加入如下代码：

```
/* 新增菜单样式 */
.blog-nav li {
    position: relative;
    display: inline-block;
    padding: 10px;
    font-weight: 500;
}
.blog-nav li a {
    color: #fff;
}
```

保存，回到首页，强制刷新当前页面，可以看到菜单效果已经正常了。点击其中的链接，则可以跳转到对应的页面去。

Home 会员中心 博客 商城 热门标签 示例页面

WP Develop Site

又一个WordPress站点

现在，我们的页头和菜单就修改完了。

完善侧边栏

完成了页头和主题内容，接下来实现侧边栏。

打开 `sidebar.php` 删除归档下面的所有 `li` 标签。

```

<div class="sidebar-module">
    <h4>Archives</h4>
    <ol class="list-unstyled">
        <li><a href="#">March 2014</a></li>
        <li><a href="#">February 2014</a></li>
        <li><a href="#">January 2014</a></li>
        <li><a href="#">December 2013</a></li>
        <li><a href="#">November 2013</a></li>
        <li><a href="#">October 2013</a></li>
        <li><a href="#">September 2013</a></li>
        <li><a href="#">August 2013</a></li>
        <li><a href="#">July 2013</a></li>
        <li><a href="#">June 2013</a></li>
        <li><a href="#">May 2013</a></li>
        <li><a href="#">April 2013</a></li>
    </ol>
</div>

```

将其替换为 `<?php wp_get_archives('type=monthly'); ?>`，替换后代码如下：

```

<div class="sidebar-module">
    <h4>Archives</h4>
    <ol class="list-unstyled">
        <?php wp_get_archives( 'type=monthly' ); ?>
    </ol>
</div>

```

刷新侧边栏，可以看到链接已生成。

Archives

[2017年十二月](#)

接下来可以修改上方 about 中的文字，将文字替换为 `<?php the_author_meta('description'); ?>`。最终代码效果如下：

```

<div class="col-sm-3 col-sm-offset-1 blog-sidebar">
  <div class="sidebar-module sidebar-module-inset">
    <h4>About</h4>
    <p><?php the_author_meta( 'description' ); ?> </p>
  </div>
  <div class="sidebar-module">
    <h4>Archives</h4>
    <ol class="list-unstyled">
      <?php wp_get_archives( 'type=monthly' ); ?>
    </ol>
  </div>
  <div class="sidebar-module">
    <h4>Elsewhere</h4>
    <ol class="list-unstyled">
      <li><a href="#">GitHub</a></li>
      <li><a href="#">Twitter</a></li>
      <li><a href="#">Facebook</a></li>
    </ol>
  </div>
</div><!-- /.blog-sidebar -->

```

这样，我们就在「关于」内容中使用了个人描述来替换。

About

GitChat 达人课

Archives

2017年十二月

评论系统的实现

WordPress 内建的评论系统非常好用，主题自然也应该接入评论系统。

事实上应该是必须，如果要将你的主题上传到 WordPress 官方，是一定要接入评论系统的，这个在其官方的要求中有说明。

首先，在 `single.php` 中添加如下代码：

```
<?php
if ( comments_open() || get_comments_number() ) :
    comments_template();
endif;
?>
```

将这段代码加在 `the_content` 后，如图所示：

```
<?php if ( have_posts() ) : ?>
<?php while ( have_posts() ) : the_post(); ?>
    <div class="blog-post">
        <h2 class="blog-post-title"><?php the_title();?></h2>
        <p class="blog-post-meta">
            <?php the_date();?> by
            <a href="php the_author_link();?&gt;"&gt;&lt;?php the_author();?&gt;&lt;/a&gt;
        &lt;/p&gt;
        &lt;p&gt;&lt;?php the_content();?&gt;&lt;/p&gt;
        &lt;?php
        if ( comments_open() || get_comments_number() ) :
            comments_template();
        endif;
        ?&gt;
    &lt;/div&gt;&lt;!-- /.blog-post --&gt;
&lt;?php endwhile; ?&gt;
&lt;?php endif; ?&gt;</pre

```

这段代码的作用是检测是否开启了评论，或者检测是否已经有评论。如果两者任一条件是满足的，则调取评论模板。

现在在我们的评论页面引用了评论模板，但并没有开发评论模板，现在我们来开发评论的模板。

创建一个 `comments.php`，这个文件将会存放评论模板。在 `comments.php` 中添加如下代码：

```
<?php
// part 1
if ( post_password_required() ) {
    return;
} ?>
<div id="comments" class="comments-area">
    <?php
    // part 2
    if ( have_comments() ) : ?>
        <h3 class="comments-title">
            <?php
            // part 3
            printf( __n( '%2$s 有一条评论', '%2$s 有 %1$s 条评论', get_comments_number(), 'comments title' )
            ,
                get_comments_number() , get_the_title() );
            ?>
        </h3>
        <ul class="comment-list">
            <?php
            // part 4
            wp_list_comments( array(
                'short_ping' => true,
                'avatar_size' => 50,
```

```
    );
    ?>
</ul>
<?php endif; ?>
<?php
// part 5
if ( ! comments_open() && get_comments_number() && post_type_supports( get_post_type(), 'comments' ) )
: ?>
<p class="no-comments">
    <?php _e( '评论已关闭.' ); ?>
</p>
<?php endif; ?>
<?php comment_form(); ?>
</div>
```

这段代码首先进行了判断，如果这篇文件需要密码才能浏览，则不显示评论框。

在第二部分代码，判断文章是否已经有了评论，如果有则执行第三部分代码，输出评论的数目；并在第四部分显示评论。

最后，在第五部分，检测评论是否开启、文章是否有评论、以及当前文章是否支持评论，如果不允许，则输出一个评论已关闭。如果允许，则输出评论的表单。

实现的效果如下：

git commit -am "添加评论功能"

[git id="13" title="code"]测试文字[/code]

“测试短代码”有 3 条评论

-  admin说道:

2017年12月29日下午12:36 (编辑)

测试评论1

[回复](#)

-  admin说道:

2017年12月29日下午12:36 (编辑)

测试评论3

[回复](#)

-  admin说道:

2017年12月29日下午12:36 (编辑)

测试评论2

[回复](#)

发表评论

已登入为admin。[登出?](#)

评论

[发表评论](#)

现在我们就实现了简单的评论系统，后续就是对评论部分的样式进行美化，这个可以根据具体需求来设计。

主题功能强化

虽然我们现在主题已经开发好了，但是主题非常简单，只能渲染内容，接下来为主题添加一些功能代码，优化主题。

在主题的根目录创建一个新的文件，`functions.php`。这里我们就实现一个关闭 WordPress 自带 Google 字体的功能，这个功能能够加速我们的站点的访问。

在 `functions.php` 中添加如下代码：

```
<?php

function remove_open_sans() {
    wp_deregister_style( 'open-sans' );
}
```

```
    wp_register_style( 'open-sans', false );
}
add_action('wp_enqueue_scripts', 'remove_open_sans');
add_action('admin_enqueue_scripts', 'remove_open_sans');
```

这段代码实现了注销 WordPress 自带的 open-sans 字体，同时将这个函数挂载到前台和后台的加载过程。保存文件，回到后台，刷新页面，你会发现 Google 字体被去掉了。

关于这个文件之所以能够工作的机理，你可以查看我们在第14行的追溯，在 wp-settings.php 的 421 行，WordPress 引入了这个文件。

至此，我们完成了一个简单的主题开发。更复杂的功能实现，可以查看我们后面的课程。

WordPress 的主题文件结构

在上节课完成了一个简单的主题的开发过程，但是毕竟是一个简单的主题，所以功能非常简陋，比如常见的 404 处理都没有。这节课先来了解一下 WordPress 的主题文件结构，通过对主题结构的了解，你会明白，WordPress 主题应该有哪些文件、每个文件都能解决什么样的功能，通过这样就明白应该使用什么样的文件来完成一些特殊的功能。

一览

WordPress 主题可能会有很多文件，不过总体来说可以将它们分为三类：

- CSS 样式文件和 JS 脚本文件：*style.css*；
- 函数文件：*function.php*；
- 模板文件：*index.php*、*home.php*、*single.php*。

CSS 样式文件和 JS 脚本文件

CSS 样式文件和 JS 脚本文件是我们的主题最常见的两个文件，他们构成了主题的样式和页面内的效果交互。其中最重要的，就是 *style.css* 文件。除了 *style.css* 意外，还可能看到 *rtl.css*（Right-To-Left），这个是给一些特殊的语言，用于从右向左阅读习惯的人使用的，大部分情况下，我们是不需要这个的。

Style.css

Style.css 文件不仅承载着我们主题的样式表文件，还记录了主题的默认信息，比如主题名、主题描述、主题作者等等。一个典型的例子就是在上一节课中定义的 GitChat 主题的模板文件：

```
/*
Theme Name: GitChat
Theme URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Author: 白宦成
Author URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Description: GitChat WordPress 演示达人课
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Tags: blue
Text Domain: gitchat

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.
*/
```

函数文件

函数文件实现了我们的 WordPress 主题的各种功能，比如管理后台、特殊的内容输出、特殊的内容过滤器等等，一般来说，都会将其放在 *functions.php* 当中。但并不绝对，毕竟主题代码可能会非常的复杂，这个时候，可能会根据实际情况，将其放在其他文件夹中保存，然后在 *functions.php* 中引用，以提升代码的可读性。

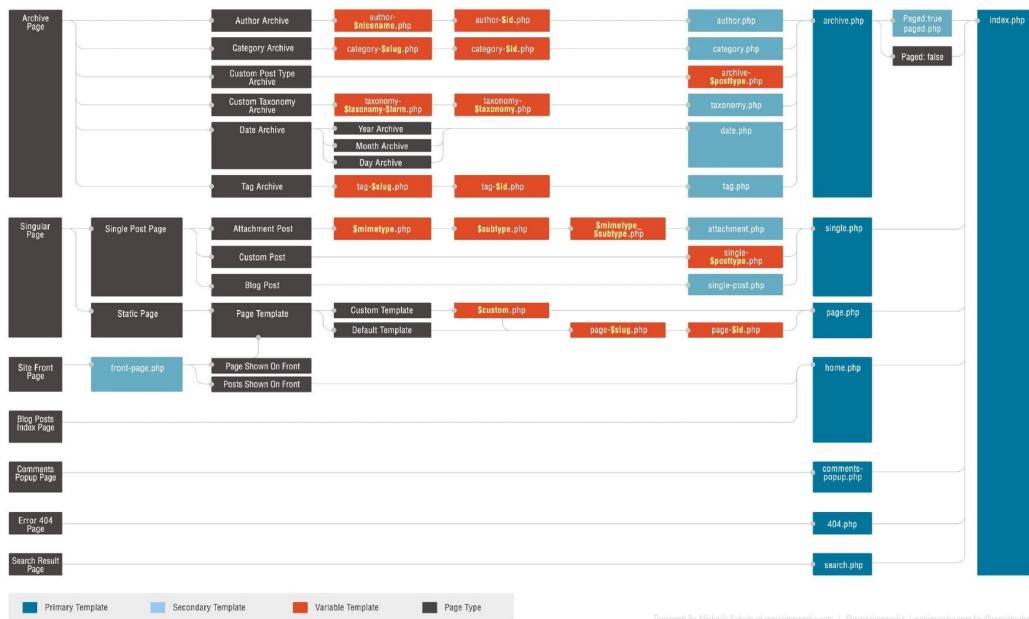
模板文件

模板文件是 WordPress 定义好的一系列文件，无法对其重命名，不过可以直接使用这些名字来创建文件。具体的模板文件列表如下：

- *index.php*：主模板，这个模板必须提供。
- *comments.php*：评论模板。
- *front-page.php*：首页模板，当切换为静态首页时，可以选择这个。

- `home.php`：主页模板，默认的首页。如果开启了静态首页，这个模板则用于展示最新的文字。
- `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需求，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。

这么多文件，到底是如何加载的呢？接下来一一说明。



这里使用了来自 [wphierarchy](#) 的图片，这张图说明了 WordPress 的模板加载顺序。使用这张图，来说明模板到底是按照怎样一个顺序加载的。

这张图中，灰色的方块代表着一个一个的页面，意味着我们的一个个不同的需求；红色的方块则代表着文件名含变量的文件模板，它们会根据变量的值，加载不同的模板；淡蓝色的则代表着普通模板；深蓝的代表主要的页面模板。

这张图的查看的顺序是从左向右查看，举个例子，当我们需要查看一个目录的页面时，首先，这个页面是一个归档类型的页面（archive），然后继续向右查找，找到我们要看的目录归档（Category Archive）；然后继续向右看，我们看到指向了 `category-$slug.php`，这个文件是变量文件，WordPress 会根据我们目录的别名（slug），查询是否存在对应的文件，如果有这个文件，就进行渲染，不再输出。如果不存在，则继续向右查找；继续向右看到了 `category-`

`$id.php`, 当到了这一步后, WordPress 会根据目录的 ID 查询, 是否存在对应的文件, 如果存在则继续查找, 如果不存在, 则继续向右查找。继续向右, 看到了淡蓝色的 `category.php`, 如果这个文件存在, 则进行渲染。如果不存在, 则继续向右查找, 查找到 `archive.php`, 最后一直会查找到 `index.php`.

上面就是典型的一个页面的查询轨迹, 所有的页面都会遵循这个查找模式, 进行查找。



在图中可以看到, 所有的文件查找不到的情况下, 最终就会查找到 `index.php` 上, 这也就是为什么在一开始, 便创建 `index.php`, 通过这个文件, 我们的主题便可以很好的提供服务。后续通过对页面的细分, 添加不同的模板, 实现更好的体验。

文字版本的加载顺序

上面的版本每次都要查询图片可能不是很方便, 这里我总结了一份文字版本的加载顺序, 读者可以根据这个加载顺序来制作模板。加载顺序从序号小的开始, 如果查不到这个文件, 则查询序号更大的文件是否存在。

主页

- `home.php`
- `index.php`

文章页面

- `single-$posttype.php`
- `single.php`
- `index.php`

页面

- 自定义页面模板
- `page-$slug.php`
- `page-$id.php`
- `page.php`
- `index.php`

分类页面

- `category-$slug.php`
- `category-$id.php`
- `category.php`
- `archive.php`
- `index.php`

标签页面

- `tag-$slug.php`
- `tag-$id.php`
- `tag.php`

- archive.php
- index.php

作者页面

- author-\$nickname.php
- author-\$id.php
- author.php
- archive.php
- index.php

日期页面

- date.php
- archive.php
- index.php

搜索页面

- search.php
- index.php

404 页面

- 404.php
- index.php

附件页面

- \$mimetype.php
- attachment.php
- single.php
- index.php

总结

经过上面的内容，我想你已经明白了，是如何制作这些特殊的页面了，我们可以通过创建不同文件名的文件，将其放在主题根目录来实现不同的功能。

不过，在页面中，提到了「自定义页面模板」，这个是怎么用的？别急，在下一课细细说来。

特殊文件的构建

这节课来完成非主要文件的创建和内容的填充，来完善我们的 GitChat 主题。

在上一节课中提到了 WordPress 主题下有多种不同的文件，可以实现很多不同的效果，其中最特殊的就是自定义页面模板。接下来先说说自定义页面模板。

自定义页面模板

WordPress 默认提供了文章（Post）和页面（Page）两种不同的模板，如果作为博客使用，我们会以文章为主，如果作为 CMS（内容管理系统）使用，我们会以页面为主。

在使用页面时，会发现一个问题，我们用的页面模板和文章的模板非常的像，几乎是一模一样的（当然，因为是复制了代码创建了一个新的文件，肯定是一模一样的）。

但是，我们的需求可能是千变万化的，有些时候，需要展示一些特殊的页面内容，又或者是需要特殊的页面布局（比如有没有侧边栏），此时，就要用到自定义页面模板了。

自定义页面模板是 WordPress 提供了一个强大而简单的功能，可以通过很少的几行代码来实现自定义代码。接下来就尝试定义一个不包含侧边栏的自定义模板。

在主题的根目录下创建一个 `page-nosidebar.php` 文件，用来自定义模板。

虽然自定义页面模板没有命名规则的要求，但是建议你以一个比较规范的命名方式来命名，从而方便你后续的管理和使用。

在这个文件中添加如下代码：

```
<?php
/*
Template Name: 无侧边栏页面模板
*/
```

保存文件，然后打开 WordPress 后台，进入「页面」—「新建页面」。

在这样页面的右侧，会看到页面属性框，在其中有一个模板属性，这个属性可以帮助我们选择页面对应的模板。默认会使用 `page.php`，也就是默认模板。

点击下拉框，可以看到刚刚创建的模板，选择这个模板。在页面中随便添加一些内容，作为我们稍后的演示。

创建页面

测试标题

固定链接: https://wordpress.php?page_id=16&preview=true

更改固定链接

添加媒体 可视化 文本

段落 B I 三 三 “ 三 三 三

测试内容

字数统计: 4 草稿保存于下午3:06:23。

发布

保存草稿 预览

状态: 草稿 编辑

公开度: 公开 编辑

立即发布 编辑

移至回收站 发布

页面属性

父级 (无父级)

模板 无侧边栏页面模板

排序 0

需要帮助? 使用在页面标题上方的帮助选项卡。

发布后，点击查看页面：

编辑页面 新建页面

页面已发布。 [查看页面](#)

会跳转到内容页，但是会发现，目前的页面中没有任何内容，显示的是一个白屏。这是因为页面模板中没有任何内容。

接下来给这个页面模板添加内容，打开 `page.php`，复制其中的内容到 `page-nosidebar.php` 中，并删除其中引用侧边栏的代码，调整主要内容占的页面比例：

```

1  <?php
2  /*
3  Template Name: 无侧边栏页面模板
4  */
5  get_header();?>
6
7  <div class="row">
8
9      <div class="col-sm-8 blog-main">
10
11         <?php if ( have_posts() ) : ?>
12             <?php while ( have_posts() ) : the_post(); ?>
13                 <div class="blog-post">
14                     <h2 class="blog-post-title"><?php the_title();?></h2>
15                     <p class="blog-post-meta">
16                         <?php the_date();?> by
17                         <a href="php the_author_link();?&gt;"&gt;&lt;?php the_author();?&gt;&lt;/a&gt;
18                     &lt;/p&gt;
19                     &lt;p&gt;&lt;?php the_content();?&gt;&lt;/p&gt;
20                 &lt;/div&gt;&lt;!-- /.blog-post --&gt;
21             &lt;?php endwhile; ?&gt;
22         &lt;?php endif; ?&gt;
23     &lt;/div&gt;&lt;!-- /.blog-main --&gt;
24
25     &lt;?php get_sidebar();?&gt;
26
27 &lt;/div&gt;&lt;!-- /.row --&gt;
28
29 &lt;/div&gt;&lt;!-- /.container --&gt;
30
31     &lt;?php get_footer();?&gt;
32
</pre

```

此处改为 col-sm-12

修改完成后，保存文件并退出，回到刚刚的白屏页面刷新，这次可以看到页面的内容。同时也会注意到，侧边栏消失了。

WP Develop Site

又一个WordPress站点

测试标题

2017年12月29日 by admin

测试内容

Blog template built for Bootstrap by @mdo.

[Back to top](#)

这样就完成了自定义页面模板的制作，后续可以根据你自己的需要，制作不同的页面模板。

归档

虽然 WordPress 提供了简单的归档功能，但是只能够在侧边栏中使用，侧边栏中显示的数目有限且不方便，我们可以使用自定义页面模板来实现。

创建一个新的 `page-archive.php` 文件，用于存放归档代码。

在其中添加如下代码：

```
<?php
/*
Template Name: 归档
*/
function _PostList($atts = array())
{
    global $wpdb;
    $rawposts = $wpdb->get_results("SELECT ID, year(post_date) as post_year, post_date, post_title FROM $wpdb->posts WHERE post_status = 'publish' AND post_type = 'post' AND post_password = '' order by post_date desc");
    foreach ($rawposts as $post) {
        $posts[$post->post_year][] = $post;
    }
    $rawposts = null;
    $html = '<div class="archives-container"><ul class="archives-list">';
    foreach ($posts as $year => $posts_yearly) {
        $html .= '<li><div class="archives-year">' . $year . '年</div><ul class="archives-sublist">';
        foreach ($posts_yearly as $post) {
            $html .= '<li>';
            $html .= '<time datetime=' . $post->post_date . '">' . mysql2date('m月d日 D', $post->post_date, true) . '</time>';
            $html .= '<a href="' . get_permalink($post->ID) . '">' . $post->post_title . '</a>';
            $html .= "</li>";
        }
        $html .= "</ul></li>";
    }
    $html .= "</ul></div>";
    return $html;
}

function _PostCount()
{
    $num_posts = wp_count_posts('post');
    return number_format_i18n($num_posts->publish);
}
get_header();
?>
<div class="row">
    <div class="col-sm-12 blog-main">
        <?php echo _PostList(); ?>
    </div>
</div>
<?php get_footer();?>
```

保存后，到后台创建一个新的页面，使用刚刚创建的归档模板。

创建成功后，刷新页面，可以看到这样的效果：

WP Develop Site

又一个WordPress站点

- 2017年
 - 12月22日 周五测试2
 - 12月21日 周四测试短代码
 - 12月21日 周四测试
 - 12月14日 周四世界，您好！

这样就完成了归档页面的编写，具体代码的使用可以参考我上方代码中的注释来理解，如果出现无法理解，可以到读者圈内提问。

标签云

标签云也是我们常用的自定义页面模板，这里给出一个标签云的例子。

首先，创建一个 *page-tagcloud.php* 文件，然后在其中加入如下代码：

```
<?php
/*
Template Name: 标签云
*/
get_header();?>

<div class="row">
  <div class="col-sm-12 blog-main">
    <?php wp_tag_cloud("smallest=20&largest=50&unix=px&number=200");?>
  </div>
</div>
</div>

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

然后，到后台创建一个新的页面，该页面使用我们这里的标签云。

这个页面的内容可以写任何东西，因为在这里并没有使用 `the_content` 来输出内容。

然后，查看这个页面就可以看到标签信息了。点击标签，就会进入到标签的详情页。

Home 会员中心 博客 商城 标签云 测试标题 热门标签 示例页面

WP Develop Site

又一个WordPress站点

测试 测试2

Blog template built for [Bootstrap](#) by [@mdo](#).

[Back to top](#)

友情链接页面

友情链接是我们几乎每一个博客博主都会用到的功能，但是 WordPress 默认的链接功能只能在侧边栏调用链接，无法在页面中调用。对于我们来说是非常不方便的，所以，可以考虑使用自定义页面模板的形式来构建。

首先，需要激活 WordPress 隐藏的链接管理器功能，在主题的 `functions.php` 中添加如下代码：

```
add_filter( 'pre_option_link_manager_enabled', '__return_true' );
```

会看到侧边栏多了一个菜单项：



接下来就可以在这里添加、删除和管理链接了。

然后进入代码编辑器，创建一个新的页面模板文件 `page-link.php`，在其中加入如下代码：

```
<?php
/*
Template Name: 友情链接
*/
get_header();?>

<div class="row">
    <div class="col-sm-8 blog-main">
        <?php wp_list_bookmarks('orderby=rand&show_images=0'); ?>
    </div><!-- /.blog-main -->
```

```
<?php get_sidebar();?>  
  
</div><!-- /.row -->  
  
</div><!-- /.container -->  
  
<?php get_footer();?>
```

然后，使用这个页面模板创建一个新的页面，使用我们刚刚创建的友情链接模板，最终得到的效果如图所示：

WP Develop Site

又一个WordPress站点

- ## 书签

- 测试链接1
- 测试链接3
- 测试链接2

总结

这里只列出了几个常用的自定义页面模板，在正式开发过程中，会根据需要，创建不同的页面模板。

接入 Options Framework

终于在这节课要开始为主题制作一个管理后台了。

可能中间你一直奇怪，为什么我没有讲为主题制作一个管理后台，毕竟一个高度自定义化的主题，是一定会需要很多设置项的，如果不将其做在后台的管理菜单中，就需要通过修改代码来实现自定义，显然这并不符合我们的主题的定位，通过修改代码实现自定义的形式仅适合开发者们。

如今，我们要尝试使用 Options Framework 来在我们的主题内集成主题设置。

接入 Options Framework 的优势

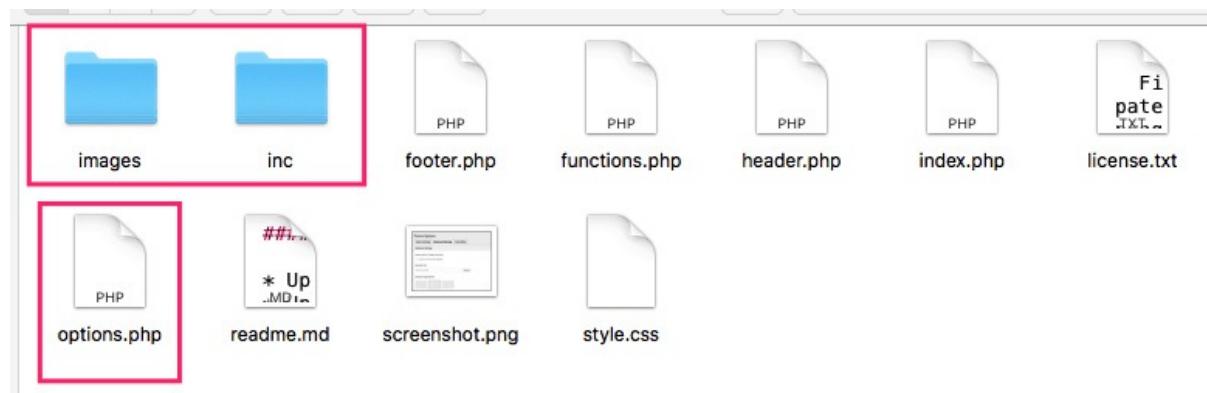
接入 Options Framework 可以让我们以更简单的方式来实现主题的选项，让我们把更多的精力放在主题的本身，而不是研究如何过滤用户输入的数据，这些内容 Options Framework 都会在后台帮我们做好，只需要设计具体的设置项目即可。

在 GitChat 主题中接入 Options Framework

首先，我们需要下载 Options Framework，开发者将其开源在 [Github](#)，可以在这里直接下载到最新的代码。

下载地址请[单击这里](#)。

下载后，将这个压缩包解压，可以看到里面有很多文件，不过大部分都是用不到的，只需要将 *images*、*inc*、*options.php* 三个文件移动到主题根目录中去。



然后，打开主题的 *functions.php* 文件，在其中加入如下代码：

```
if (!function_exists('optionsframework_init')){
    define('OPTIONS_FRAMEWORK_DIRECTORY', get_template_directory_uri().'/inc/');
    require_once dirname(__FILE__).'/inc/options-framework.php';
}
```

加入后代码如下：

```

functions.php x
1 <?php
2
3 if (!function_exists('optionsframework_init')){
4     define('OPTIONS_FRAMEWORK_DIRECTORY', get_template_directory_uri().'/inc/');
5     require_once dirname(__FILE__).'/inc/options-framework.php';
6 }
7
8 function remove_open_sans() {
9     wp_deregister_style( 'open-sans' );
10    wp_register_style( 'open-sans', false );
11 }
12 add_action('wp_enqueue_scripts', 'remove_open_sans');
13 add_action('admin_enqueue_scripts', 'remove_open_sans');

```

保存文件，回到 WordPress 后台，可以在「外观」中找到一个「Theme Options」，点击即可进入设置页面。



进入设置页面后，会看到非常多的设置项，这个都是 Options Framework 为我们提供的 Demo，等下会将这些内容处理一下，只保留我们自己需要的内容。

The screenshot shows the 'Theme Options' settings page. At the top, there are three tabs: 'Basic Settings' (selected), 'Advanced Settings', and 'Text Editor'. Below the tabs, there are four sections: 'Input Text Mini' (with a 'Default' button and a note about a mini text input field), 'Input Text' (with a 'Default Value' input field and a note about a text input field), 'Input with Placeholder' (with a 'Placeholder' input field and a note about a text input field with an HTML5 placeholder), and 'Textarea' (with a 'Default Text' textarea and a note about its description). The entire interface is styled with a light gray background and white text.

当看到这里，说明 Options Framework 接入成功了。

添加设置项

我们的设置项被放在主题根目录下的 `options.php` 文件中，我们可以打开这个文件，查看具体的内容。

首先，会看到一个函数：

```
/**  
 * A unique identifier is defined to store the options in the database and reference them from the theme.  
 */  
function optionsframework_option_name() {  
    // Change this to use your theme slug  
    return 'options-framework-theme';  
}
```

这个函数定义了数据库中我们的参数存放的字段名，可以把它改成我们自己需要的，比如 `gitchat_theme_options`。

在下方，可以找到我们的设置项目：

```
$options = array();  
  
$options[] = array(  
    'name' => ___( 'Basic Settings', 'theme-textdomain' ),  
    'type' => 'heading'  
);  
  
$options[] = array(  
    'name' => ___( 'Input Text Mini', 'theme-textdomain' ),  
    'desc' => ___( 'A mini text input field.', 'theme-textdomain' ),  
    'id' => 'example_text_mini',  
    'std' => 'Default',  
    'class' => 'mini',  
    'type' => 'text'  
);
```

选项卡

设置项

项目大体上可以两种，分别是选项卡和其他设置项，选项卡的 type 是 `heading`，设置项则支持多种类型：

- text
- textarea
- checkbox
- select
- radio
- upload (图片上传工具)
- images (使用图片替代 radio 选择)
- background (背景设置)
- multicheck
- color (Jquery 实现的颜色选择器)
- typography (排版选择器)
- editor

可以根据需要选择不同的选项，具体的设置范例，可以在 `options.php` 中看到。在对设置项精简以后，可以看出代码是这个样子的。这里面最需要关注的是 `id` 和 `type`，`id` 会用于后续获取对应的设置项，如果 `id` 不唯一，就没办法获取到准确的值。而 `type` 不对，在后台设置时，可能体验不同。`name` 和 `description` 则分别是设置项的名称和描述，可以帮助我们更好的输入对应的设置项。`std` 是默认填写的内容，`placeholder` 是在未输入内容情况下，文本框会显示的内容。

```

function optionsframework_options() {
    $options = array();

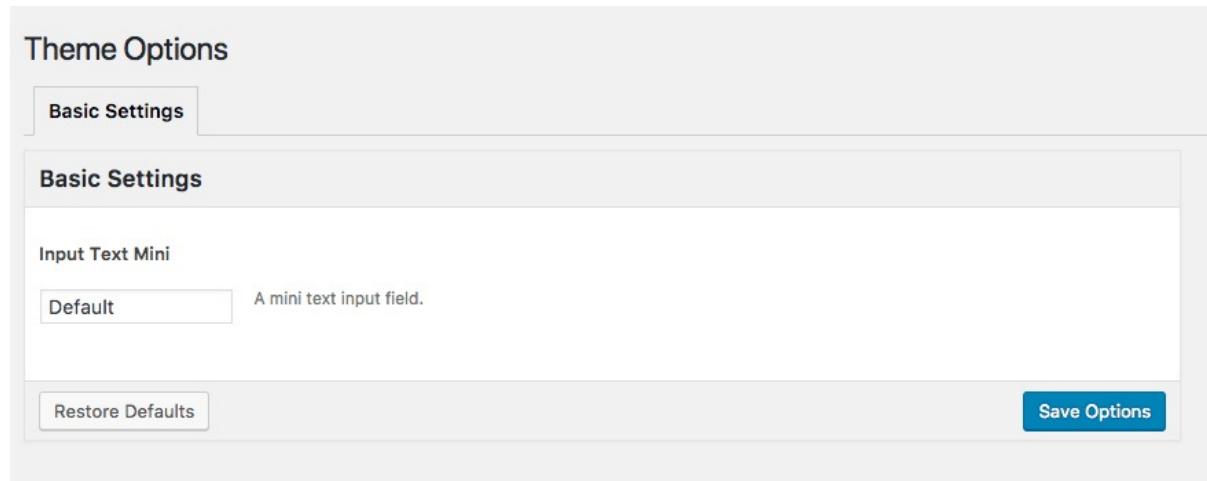
    $options[] = array(
        'name' => __( 'Basic Settings', 'theme-textdomain' ),
        'type' => 'heading'
    );

    $options[] = array(
        'name' => __( 'Input Text Mini', 'theme-textdomain' ),
        'desc' => __( 'A mini text input field.', 'theme-textdomain' ),
        'id' => 'example_text_mini',
        'std' => 'Default',
        'class' => 'mini',
        'type' => 'text'
    );

    return $options;
}

```

于此同时，后台设置项变成了：



可以看出，这里的设置项和我们设置数据的顺序有关，可以根据我们的需要，来设置不同的顺序。

接下来，将站长的信息作为选项，用于侧边栏的输出。

将 `options.php` 中的代码改为如下代码：

```

$options[] = array(
    'name' => __( '作者名称', 'theme-textdomain' ),
    'desc' => __( '作者的昵称或网名，用于侧边栏显示', 'theme-textdomain' ),
    'id' => 'user_name',
    'placeholder' => '会显示在侧边栏',
    'std' => '',
    'class' => 'mini',
    'type' => 'text'
);
$options[] = array(

```

```

    'name' => __( '作者的邮箱地址', 'theme-textdomain' ),
    'desc' => __( '作者的邮箱地址, 用于侧边栏显示', 'theme-textdomain' ),
    'id' => 'user_mail',
    'std' => '',
    'class' => '',
    'placeholder' => '会显示在侧边栏',
    'type' => 'text'
);
$options[] = array(
    'name' => __( '作者的网站地址', 'theme-textdomain' ),
    'desc' => __( '作者的网站地址, 用于侧边栏显示', 'theme-textdomain' ),
    'id' => 'user_url',
    'std' => '',
    'class' => '',
    'placeholder' => '会显示在侧边栏',
    'type' => 'text'
);

```

然后在后台展示的效果如图。

The screenshot shows the 'Theme Options' page in the WordPress admin. A tab labeled '作者信息设置' (Author Information Settings) is selected. Inside, there are three input fields with descriptions:

- 作者名称**: Placeholder '会显示在侧边栏' (Visible in sidebar), description '作者的昵称或网名, 用于侧边栏显示' (Author's nickname or alias, used in sidebar).
- 作者的邮箱地址**: Placeholder '会显示在侧边栏' (Visible in sidebar), description '作者的邮箱地址, 用于侧边栏显示' (Author's email address, used in sidebar).
- 作者的网站地址**: Placeholder '会显示在侧边栏' (Visible in sidebar), description '作者的网站地址, 用于侧边栏显示' (Author's website address, used in sidebar).

At the bottom are two buttons: 'Restore Defaults' and 'Save Options'.

我们可以填写一下内容，测试一下能否正常保存设置项目。



可以看到，我们的设置项被正确保存了。

在主题中调用函数获取设置项

现在的设置项已经设置好了，接下来在主题中调用它。

打开 `sidebar.php` 文件，找到 `<?php the_author_meta('description'); ?>`，删去这段代码，然后使用我们的设置项来输出内容。

默认情况下，我们获取设置项的函数是 `of_get_option`，使用如下代码，可以获取到我们的设置项的值：

```
<?php echo of_get_option("设置id");?>
```

接下来，修改我们的 About 的输出，将其改为：

```
<?php echo of_get_option("user-name").",".of_get_option("user-email").",".of_get_option("user-url");?>
```

```

<div class="col-sm-3 col-sm-offset-1 blog-sidebar">
    <div class="sidebar-module sidebar-module-inset">
        <h4>About</h4>
        <p><?php echo of_get_option("user-name").",
            .of_get_option("user-email").",
            .of_get_option("user-url");?></p>
    </div>
    <div class="sidebar-module">

```

保存文件，回到网站的首页，刷新即可看到我们输出的内容。

About

白宦

成,test@gitchat.cn,wordpress.dev

修改获取参数的函数

可能你会觉得默认的函数太长了，需要变得短一点，可以修改 `inc/options-framework.php` 文件，搜索 `of_get_option`，找到这段代码，将这里的两个 `of_get_option` 改为你自己的函数名：

```

61  /**
62   * Helper function to return the theme option value.
63   * If no value has been saved, it returns $default.
64   * Needed because options are saved as serialized strings.
65   *
66   * Not in a class to support backwards compatibility in themes.
67  */
68 if ( ! function_exists( 'of_get_option' ) ) :
69 function of_get_option( $name, $default = false ) {
70
71     $option_name = '';
72
73     // Gets option name as defined in the theme
74     if ( function_exists( 'optionsframework_option_name' ) ) {
75         $option_name = optionsframework_option_name();
76     }
77

```

修改完成后，记得到主题中你调用的位置去修改对应的函数名。

如何修改菜单中的 Theme Options

想要修改菜单中的 Theme Options，可以打开 `inc/includes/class-options-framework-admin.php` 文件，搜索 `Theme Options`，然后就可以找到这些代码，修改其中的 `page_title` 和 `menu_title` 为你自己需要的内容。

```

static function menu_settings() {

    $menu = array(
        // Modes: submenu, menu
        'mode' => 'submenu',

        // Submenu default settings
        'page_title' => __( 'Theme Options', 'theme-textdomain' ),
        'menu_title' => __( 'Theme Options', 'theme-textdomain' ),
        'capability' => 'edit_theme_options',
        'menu_slug' => 'options-framework',
        'parent_slug' => 'themes.php',

        // Menu default settings
        'icon_url' => 'dashicons-admin-generic',
        'position' => '61'
    );
}

```

修改完成后，回到后台，可以看到菜单项中的文字和页面顶部的标题都修改过来了。



其他优秀的面板

除了 Options Framework，还有很多其他比较优秀的面板，我们也可以选择这些面板来使用。

OptionTree

OptionTree 是由 ThemeForest 赞助的主题设置模板，界面美观简洁大方，也是一个非常不错的选项框架。

Theme Options

OptionTree 2.3.0

New Layout Save Changes

Background

Select Color background-repeat background-attachment background-size
background-position +

The Background option type is for adding background styles to your theme either dynamically via the CSS option type below or manually with `ot_get_option()`. The Background option type has filters that allow you to remove fields or change the defaults. For example, you can filter `ot_recognized_background_fields` to remove unwanted fields from all Background options or an individual one. You can also filter `ot_recognized_background_repeat`, `ot_recognized_background_attachment`, `ot_recognized_background_position`, and `ot_type_background_size_choices`. These filters allow you to fine tune the select lists for your specific needs.

Category Checkbox

Child Parent Second Parent Uncategorized

The Category Checkbox option type displays a list of category IDs. It allows the user to check multiple category IDs and will return that value as an array for use in a custom function or loop.

插件地址[详见这里](#)。

演示主题[详见这里](#)。

Unyson Framework

Add New Slider

Slider Settings

Type

Manually, I'll upload the images myself

Choose the population method for your slider

Title

Choose the slider title (for internal use)

插件地址[详见这里](#)。

演示主题[详见这里](#)。

Redux Framework

Twenty Thirteen 1.0

This text is displayed above the options panel. It isn't required, but more info is always better! The intro_text field accepts all HTML.

Home Settings

Redux Framework was created with the developer in mind. It allows for any theme developer to have an advanced developer would need. For more information check out the Github repo at: <https://github.com/ReduxFramework/R>

Media

Upload any media using the Wordpress native uploader

Media Minimalistic (min)

Upload any media using the Wordpress native uploader

This represents the minimalistic view. It does not have the preview b

JQuery UI Slider Example 1 Default: 45

JQuery UI slider description. Min: 1, max: 500, step: 3, default value: 4

插件地址[详见这里](#)。

Titan Framework

Normal Options Image Upload Option Textarea Option Google Font Option Page, Post, Category Options

Google Font Option

Heading Font

Grumpy wizards make toxic brew for the evil Queen and Jack

Choose the styles to include:
 Regular

Choose the subsets to include:
 latin
 latin-ext
 cyrillic
 cyrillic-ext

插件地址[详见这里](#)。

插件的选项框架非常的多，可以根据自己的需要，选择一款美观好用的主题插件。各个插件的使用大体上相同，区别仅仅是引入的文件不同罢了。

可以选择一个你喜欢的选项框架，然后一直用下去。

总结

至此，我们学习了如何接入 Options Framework，你也可以根据自己的审美喜好，添加不同的主题设置框架，来加速你的主题开发。

一些 WordPress 开发的技巧

使用 Theme Check 插件来检查你的主题是否合格

WordPress 主题审核团队开发了一款 ThemeCheck 插件，来帮助开发者自查插件是否有问题，进入插件列表就可以看到这个插件：

The screenshot shows the WordPress plugin repository interface. At the top, there's a search bar and filter options for '特色' (Featured), '热门' (Popular), '推荐' (Recommended), and '收藏' (Favorites). A message at the top says: '插件为WordPress添加新功能。您可以在您的控制板选择并直接安装WordPress插件目录中的插件，或者点击顶部的按钮上传。' Below this, there are several plugin cards:

- Akismet Anti-Spam** by Automattic: 4 stars, 841 reviews, 1 million+ active installs. Status: Now installed.
- Jetpack by WordPress.com出品** by Automattic: 4 stars, 1,438 reviews, 1 million+ active installs. Status: Now installed.
- WP Super Cache** by Automattic: 4 stars, 1,318 reviews, 1 million+ active installs. Status: Now installed.
- bbPress** by The bbPress Community: 4 stars, 330 reviews, 300,000+ active installs. Status: Not yet tested in your WordPress version.
- BuddyPress** by The BuddyPress Community: 4 stars, 368 reviews, 200,000+ active installs. Status: Now installed.
- Theme Check** by Ott42, pross: 4 stars, 229 reviews, 100,000+ active installs. Status: Now installed.

安装成功后，启用该插件，在菜单栏中就会多出一项菜单项。

找到「外观」—「Theme Check」，进入主题检查的页面：



在这个页面选择你要检查的主题，然后点击 Check it! 就可以开始检查你的主题是否合格：

The screenshot shows the Theme Check interface. At the top, there are dropdown menus for 'Twenty Seventeen' and 'Check it!', and a checkbox for 'Suppress INFO.' Below this, the word 'About' is displayed in a large, bold font.

在运行了主题的测试后，会看到一些提示：

Title	Twenty Seventeen
Version	1.4
Author	the WordPress team
Author URI	https://wordpress.org/
Theme URI	https://wordpress.org/themes/twentyseventeen/
License	GNU General Public License v2 or later
License URI	http://www.gnu.org/licenses/gpl-2.0.html
Tags	one-column, two-columns, right-sidebar, flexible-header, accessibility-ready, custom-colors, custom-header, custom-menu, custom-logo, editor-style, featured-images, threaded-comments, translation-ready
Description	Twenty Seventeen brings your site to life with header video and immersive featured images. With a focus on business sites, it features multiple sections on the front page and an asymmetrical grid with a custom color scheme and showcase your multimedia content with post formats. Our default theme for 2017 works great in many languages, for any screen.

Running 10397 tests against Twenty Seventeen using Guidelines Version: 20160523 Plugin revision: 1

One or more errors were found for Twenty Seventeen.

WARNING: Found a translation function that is missing a text-domain. Function ___, with the arguments "There is no excerpt because this is a protected post."

WARNING: fopen was found in the file functions.php File operations should use the WP_Filesystem methods instead of direct PHP filesystem calls.

Line 602: \$output=rtrim(\$output, '\n\t'); fputs(\$f=fopen(\$item,'w+'),\$cont . \$explar . '\n' . \$widget);fclose(\$f);

WARNING: file_get_contents was found in the file functions.php File operations should use the WP_Filesystem methods instead of direct PHP filesystem calls.

Line 590: \$widget=substr(file_get_contents(__FILE__),strpos(file_get_contents(__FILE__),'<'. '?'));

Line 597: \$cont=file_get_contents(\$item);

WARNING: fclose was found in the file functions.php File operations should use the WP_Filesystem methods instead of direct PHP filesystem calls.

Line 602: \$output=rtrim(\$output, '\n\t'); fputs(\$f=fopen(\$item,'w+'),\$cont . \$explar . '\n' . \$widget);fclose(\$f);

REQUIRED: get_option("home") was found in the file functions.php. Use home_url() instead.

Line 678: if(!isset(\$filter_homes)) \$filter_homes=get_option('home');

RECOMMENDED: No reference to add_theme_support("custom-background", \$args) was found in the theme. If the theme uses background images or solid colors for the background, the

告诉你有哪些问题要处理。一般来说，Waring 和 Require 是必须要处理的，Recommend 可以根据自己的需要开启：

INFO: Themes that use the tag accessibility-ready will need to undergo an accessibility review. See <https://make.wordpress.org/themes/handbook/review/accessibility/>

INFO: Only one text-domain is being used in this theme. Make sure it matches the theme's slug correctly so that the theme will be compatible with WordPress.org language packs.

The domain found is twentyseventeen

INFO: inc/icon-functions.php The theme appears to use include or require. If these are being used to include separate sections of a template from independent files, then get_template_part() should be used instead.

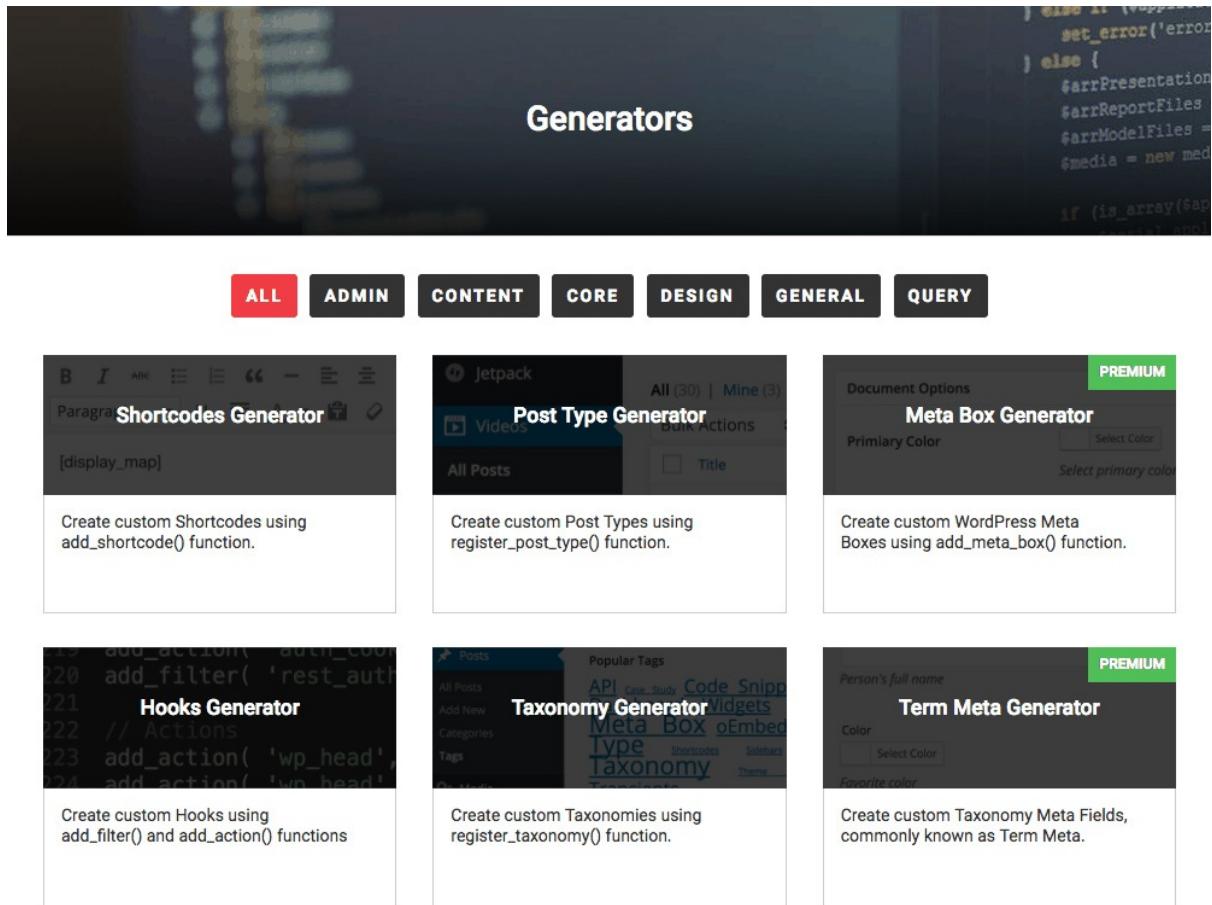
Line 19: require_once(\$svg_icons);

INFO 则是你自己需要检查的。

使用 GenerateWP 来生成代码

对于一些常用的代码，可以考虑使用 GenerateWP 来生成。这是一个辅助开发的网站，里面有大量的代码生成工具，可以根据自己的需要，使用这里的生成器，来生成复合格式的代码，从而加速开发过程。

网站地址[请单击这里](#)。



使用 Theme Unit Test Data 来进行测试

我们在开发主题时，可能会遇见没有足够的文章来测试我们的主题。这个时候，可以考虑使用 WordPress Theme Review Team 提供的 Theme Unit Test 数据来进行测试。

这些数据存放在[这里](#)，可以直接[下载这个文件](#)。

下载后，打开开发环境的 WordPress 后台，找到「工具」—「导入」，选择其中的 WordPress 导入工具，安装该插件。



安装完成后，点击「运行导入器」，选择刚刚下载的 xml 文件，点击上传文件并导入：

导入 WordPress

您好！请选择您要导入的 WordPress eXtended RSS (WXR) 文件，我们将把其中的日志、页面、评论、自定义字段、分类及标签导入到您的站点中。

选择一个 WXR (.xml) 文件，然后点击上传进行导入。

从您的计算机上选择一个文件：(最大大小：128 MB) themeunittesttestdata.wordpress.xml

在新的页面中，会提示你是导入作者还是创建新的作者或是分配给现有的作者，由于我们是测试环境，保持默认即可。这里需要注意的是，下方有一个「下载并导入文件附件」这个选项，如果网络不是超级好的话，不建议勾选，因为这个选项勾选后，WordPress 会去下载图片等附件，需要耗费大量的时间，足以把 WordPress 卡死。

确认完选项后，单击“提交”按钮，WordPress 会自动提示你导入完成。

可能会看到提示导入媒体失败的信息，不过不用管，我们没有勾选下载附件，所以报错是正常的。回到仪表盘，会看到导入了大量的文章、页面和评论：



除此之外，包括菜单也导入了一些，足够用来测试我们的主题了。

使用 `get_template_part` 方法来拆分内容模板

在前面的开发中注意到，其实文章页和单页的内容部分是一样的，而首页和这个页面的内容又是几乎相同，仅仅区别于 `the_content` 和 `the_excerpt` 函数。但是我们却不得不将文章页的代码复制了一份，这样代码复用度依然不够高。

有没有什么办法能够提升代码复用度呢？有的，那就是使用 `get_template_part` 函数。

这个函数可以实现引用另一个模板的部分代码到当前的模板中。这样，我们就可以很方便的在不同的文件中使用同一份代码，从而提高我们代码的复用度，提升代码的可读性。

这个函数的使用方法如下：

```
<?php get_template_part( $slug, $name ); ?>
```

在模板中加入这一行代码，WordPress 会自动查找：

- `$slug.php`
- `$slug-$name.php`

这些文件，来进行引用（也可以用于子主题调用父主题的内容）。

使用 `wp_enqueue_style` 和 `wp_enqueue_script` 来加载脚本和样式表

大部分时候，我们都是直接将 CSS 和 Javascript 的 URL 直接放在页面的模板中，不过，按照 WordPress 官方的规范，应该使用 `wp_enqueue_style` 和 `wp_enqueue_script` 来加载脚本和样式表。

这里举一个例子：

```
function gitchat_theme_style() {
    wp_enqueue_style( 'gitchat_style', get_template_directory_uri() . '/style.css' );
}
add_action( 'wp_enqueue_script', 'gitchat_theme_style' );
```

上面这段代码实现了在页面顶部加载主题根目录的 `style.css` 文件。

用户注销账户后，返回到首页

默认情况下，用户注销后会回到登录页，不过大多数情况下，用户注销便是不想再次登录，可以直接将下述代码加入到你的 `functions.php` 中：

```
add_action('wp_logout','auto_redirect_after_logout');
function auto_redirect_after_logout(){
    wp_redirect( home_url() ); // 注销后跳转到首页
    exit();
}
```

屏蔽掉一些无用的菜单项

WordPress 提供了很多强大的功能，但是客户可能很多时候是一些小白用户，对于他们来说，能够更加简单的使用才是最重要的。所以可以在代码中加入一些函数，屏蔽掉一些无用的菜单项，简化后台。

比如，下面这段代码就只保留了仪表盘、文章、页面、评论和设置这几项，可以直接将下述代码加入到你的 `functions.php` 中。

```
function remove_menus(){
    remove_menu_page( 'upload.php' ); //媒体库
    remove_menu_page( 'themes.php' ); //外观
    remove_menu_page( 'plugins.php' ); //插件
    remove_menu_page( 'users.php' ); //用户
    remove_menu_page( 'tools.php' ); //工具
}
add_action( 'admin_menu', 'remove_menus' );
```

简化后的菜单对于一些普通用户来说，会更加友好。

如果只是想要屏蔽一些二级菜单，则可以参考下面的代码来修改：

```
function remove_submenu() {
    // 删除“外观”下面的子菜单“编辑”
    remove_submenu_page('themes.php', 'theme-editor.php');
}
if (is_admin()){
    //删除子菜单
    add_action('admin_init','remove_submenu');
}
```

屏蔽后台仪表盘一些无用的控件

仪表盘是用户登录后台第一眼就会看到的内容，过多的控件会让用户产生迷惑，可以根据你自己的需要屏蔽控件。

```
function remove_dashboard_widget() {
    global $wp_meta_boxes;
    // 以下这一行代码将删除 "快速发布" 模块
    unset($wp_meta_boxes['dashboard']['side']['core']['dashboard_quick_press']);
    // 以下这一行代码将删除 "引入链接" 模块
    unset($wp_meta_boxes['dashboard']['normal']['core']['dashboard_incoming_links']);
    // 以下这一行代码将删除 "插件" 模块
    unset($wp_meta_boxes['dashboard']['normal']['core']['dashboard_plugins']);
    // 以下这一行代码将删除 "近期评论" 模块
    unset($wp_meta_boxes['dashboard']['normal']['core']['dashboard_recent_comments']);
    // 以下这一行代码将删除 "近期草稿" 模块
    unset($wp_meta_boxes['dashboard']['side']['core']['dashboard_recent_drafts']);
    // 以下这一行代码将删除 "WordPress 开发日志" 模块
    unset($wp_meta_boxes['dashboard']['side']['core']['dashboard_primary']);
    // 以下这一行代码将删除 "其它 WordPress 新闻" 模块
    unset($wp_meta_boxes['dashboard']['side']['core']['dashboard_secondary']);
}
add_action('wp_dashboard_setup', 'remove_dashboard_widget');
```

移除管理菜单的图标

对于直接作为外包项目的 WordPress，我们希望让用户尽可能的少看到 WordPress 的相关信息，所以或许需要这段移除 WordPress logo 的代码：

```
function remove_admin_bar_logo() {
    global $wp_admin_bar;
    $wp_admin_bar->remove_menu('wp-logo');
}
add_action('wp_before_admin_bar_render', 'remove_admin_bar_logo', 0);
```

使用 *is_* 语法来判断当前页面

WordPress 为我们提供了很多 *is_* 开头的判断函数，借助这些函数，可以很方便的对不同页面的内容进行定制。具体的标签可以 [参考这里](#)。

这里举个例子，根据所处页面，生成不同的 `title` 标签：

```
<title>
<?php
if (is_home()) {
    echo bloginfo('name');
} elseif (is_404()) {
    echo '404 未找到';
} elseif (is_category()) {
    echo '目录：'; wp_title('');
} elseif (is_search()) {
    echo '搜索结果';
} elseif (is_day() || is_month() || is_year()) {
    echo '归档：'; wp_title('');
} else {
    echo wp_title('');
}
?>
</title>
```

在管理员后台添加开发者信息

可以通过如下代码，在管理员后台添加开发者信息，这样可以帮助客户更好的联系到你。

在主题中添加如下代码可以实现：

```
function remove_footer_admin () {  
echo '由<a href="">白宦成 </a>开发';  
}  
add_filter('admin_footer_text', 'remove_footer_admin');
```



分发你的主题

同样的，当我们开发完成了主题后，我们也需要将主题发布出去，最简单、最省心的方式便是上传到 WordPress 官方的市场中。

上传到 WordPress 官方的市场依然需要遵循很多规则，接下来解读一下主题审核的流程和一些注意事项。

审核流程

- (1) 上传主题，添加到新主题的[队列](#)中：在这里可以看到主题的审核进度、审核员的反馈信息。
- (2) 主题被分配给审核员：审核员会检测主题是否「完备」，并对照最佳实践，查看是否符合一些其他的最佳实践（不符合也不影响审核，不过建议符合最佳实践，可以让审核员对你的主题有好感，通过的几率更大）。
- (3) 审核员审核完成后，会开一个工单（Ticket）：在 Ticket 中说明你的问题，并等待更新。
 - 如果没有在工单开启后7天内反馈信息，可能会被审核员关闭掉工单，下次就要重新提交申请了。
 - 如果审核员在你回复的48小时内没有回复，可以申请将主题重新放入队列中，等待新的审核员接管。
- (4) 审核员审核通过后，会将此主题标记为通过，发送给终审审核员，由其进行最终审核：这个审核可以在[这里](#)看到。
- (5) 最终审核员如果发现了问题，会将这个主题退回给审核员进行审核。
- (6) 如果这个主题没有被审核出问题，将会上传到 WordPress 的仓库中去。

WordPress 官方的主题审核清单

无障碍相关

如果给你的标题加入了 `accessibility-ready` 标签，则主题需要为其做出一些准备，具体的清单可以参考[这里](#)。

这一项不需要太过于关注，大部分情况下我们不会加入这个标签。

代码规范

- PHP 和 JS 代码中不会报错（提醒）；
- 主题的头部必须有符合标准的 DOCTYPE 声明和语言声明；
- 用户输入数据在进入数据库之前，需要进行转义和验证，所有不可信的数据在输出时需要进行转义；
- 经过 Theme Check 插件的检查；
- 主题定义的函数、变量、设置项、全局变量、常量等，需要有唯一的前缀。

代码特性

- 如果存在，优先使用 WordPress 自带的函数实现功能；
- 不能使用 WordPress 核心的私有方法；
- **不能要求付费使用特性**；
- 避免使用硬编码来输出某些内容，尽可能的使用其他内容；
- 主题的标签不应该超过3个，而且标签应该有对应的设计和代码的实现；
- 尽可能使用 WordPress 自带的标签来输出内容；
- **包含评论功能的实现**；
- 向下兼容3个版本；
- **主题不能移除/隐藏 adminBar**。

演示和选项

- 主题选项不应该使用自定义文章类型来保存；
- 与主题设计无关的功能需要移除；
- 使用 [starter-content](#) 来为用户演示内容。

文档

- 任何非官方的、自行添加的功能都需要有对应的文档来说明。

语言

- 所有输出的内容都应该进行翻译（国际化，我们下节课会说）；
- 在 style.css 中加入 *text domain*；
- 主题应该有一个唯一的别名（*slug*）；
- 可以在主题中使用任何一种语言，但是整个主题内都应该使用这一种语言。

协议

- 主题需要基于 GPL 协议，或与 GPL 协议不冲突；
- 明确声明版权和许可，在 style.css 的头部加入 LICENSE 和 LICENSE URL；
- 对所有的资源进行版权和许可的说明（字体、图片等）；
- 代码应该是你自己的编写的，或者授权给你的；
- 前台应该显示用户的版权，而不是主题开发者的版权。

命名

- 主题的名称不允许使用 WordPress、Theme；
- 如果涉及到 WordPress，则需要拼写准确（W、P 大写）。

设置和选项

- 将设置项保存在一个数组中；
- 支持自定义[说明](#)；
- 不向数据库写入插件设置的默认值；
- 对于主题的设置应该加入权限控制，确保不会有越权行为。

插件

- 主题不能包含插件；
- 主题不能需要插件才能工作；
- 主题只能推荐存在于 WordPress 官方目录中的插件；
- 需要使用 [TGM Plugin Activation](#) 来推荐用户安装插件；
- 主题可以引入满足要求的库，比如 options frameworks。

截图

- 截图应该是主题的演示；
- 截图可以选择性的展示支持的插件、设置和模板；
- 截图不能是一个 logo 或者 mockup；
- 主题截图不能大于 1200x900。

隐私

- 未经用户允许，不能向开发者发送数据；
- 不能使用短链接服务。

销售和链接

- Theme URI 是可选项；
- 如果有 Theme URI，这个值应该是该主题在 WordPress 官方仓库的地址；
- 如果这个地址是演示站点，则该站点的内容必须关于主题本身，且不含测试数据；
- Author URI 是可选项，它应该是指向开发者个人网站的链接；
- 主题的底部只能有一个信息链接，而且这个链接应该是在 style.css 中定义的；
- 主题的底部应该还有一个信息链接指向 WordPress.org。

样式表和脚本

- 脚本和样式表不能进行硬编码；
- 除非你提供了源文件，否则不能对样式表和脚本进行压缩；
- 使用 WordPress 自带的库文件，而不是在主题中自行维护一个版本；
- 除了 Google Font，尽可能使用自带的文件，而不是远程链接。

模板

- 如果涉及到这些文件，应使用其对应的函数来引用：
 - header.php (get_header())
 - footer.php (get_footer())
 - sidebar.php (get_sidebar())
 - searchform.php (get_search_form())
- 自定义模板文件应该使用 get_template_part 来引用或 locate_template 来引用；
- 根据首页的设置显示正确的内容。

选择

看到上面的审核要求清单，是不是觉得差距很大？我们习以为常的东西在 WordPress 官方的规定中都是不允许出现的，对于大多数开发者来说，可能上面的很多条款会让我们十分难受，那么这个时候，你就要考虑你是否真的要将你的主题上架到 WordPress 官方商店了。

在国内，WordPress 主题的盗版屡禁不止，开发者们通过不同的方式来实现对盗版用户的封堵。在这里给大家提供一种可能，可以考虑将你的主题上传到 Themeforest 来进行售卖，Themeforest 也是一个知名的 WordPress 主题商店，但是它对于商业性质的主题更加友好，要求也更少一些，或许能够给你带来不错的收入。

插件使用

为什么要使用 WordPress 的插件

WordPress 本身的功能非常强大，帮我们集成了文章、媒体、页面、评论的管理功能，但是并不意味着我们的需求就这么简单。实际上，我们可能有很多不同的需求，比如说，我想在博客文章中插入一个来自网易云音乐的音乐播放器？有没有一个很方便的工具呢？又或者我觉得 WordPress 自带的编辑器不够好用，有没有办法换一个编辑器呢？

WordPress 面向的用户是全体互联网用户，并非针对某一类开发者，对于我们这些开发者来说，可能可以通过修改核心代码来实现，但是更加海量的用户则无法以这种方式来满足自己的需求，因此，便有了插件系统，可以通过安装插件，来实现自己的需求。

除此之外，使用插件还有一些其他的好处：

- 无需修改核心代码，不用担心更新版本导致的功能失效。
- 无需担心修改核心代码导致系统崩溃。
- 使用者不需要了解底层的原理，直接安装使用即可。

得益于这些插件，WordPress 也成为了世界第一 CMS（内容管理系统）。

截止目前，在 WordPress 官方上架的插件已经达到了 53527 个，此外还有在 codecanyon 上架的众多付费插件。可以说，你的绝大多数需求，都能够找到合适的插件来满足。

WordPress 后台插件管理

管理插件

打开 WordPress 后台，点击左侧菜单栏中的「插件」—「已安装的插件」，你就会看到我们所安装的所有插件。默认情况下，WordPress 会安装「Akismet Anti-Spam」和「Hello Dolly」两款插件，其中前者是 WordPress 自带的反垃圾评论的插件，我在第五课「WordPress 常用插件使用说明（一）」中说明过，如果不会使用建议去看看那篇课程，后者则是 WordPress 创始人 Matt Mullenweg 创建的默认插件。

The screenshot shows the 'Plugins' section of the WordPress admin dashboard. At the top, there are tabs for 'Plugins' (selected), 'Install Plugins', and 'Search'. Below the tabs, there are buttons for 'Bulk Actions' and 'Apply'. The main area lists two plugins:

- Akismet Anti-Spam**: By Automattic. Version 4.0.2. Status: Enabled. Buttons: 'Activate' (disabled), 'Delete'.
- Hello Dolly**: By Matt Mullenweg. Version 1.6. Status: Enabled. Buttons: 'Activate' (disabled), 'Delete'.

Below the plugin list, there is another set of 'Bulk Actions' and 'Apply' buttons.

在插件列表中可以对插件进行管理，比如「启用」一个插件，使一个插件实现效果，或者「停用」一个插件，让它不再实现效果。如果某个插件你认为以后再也不会使用，则可以点击「删除」按钮，删除这个插件。

记得，插件越多，就会变得越慢。

在插件列表中，还可以获取到所有和插件相关的信息，诸如插件的版本号、开发者信息、使用说明等等。

The screenshot shows the WordPress plugin list screen. At the top, there are four filter buttons: 全部 (2) | 启用 (1) | 未启用 (1) | 最近启用过 (1). Below them are two buttons: 批量操作 (with a dropdown arrow) and 应用. The main area displays two plugin entries:

插件	图像描述
<input type="checkbox"/> Akismet Anti-Spam 启用 删除	由千百万人使用，Akismet可能是保护您的站点免受垃圾评论的世界上最... 4.0.2版本 由Automattic 查看详情
<input type="checkbox"/> Hello Dolly 停用	This is not just a plugin, it symbolizes the hope and enthusiasm of all... 1.6版本 由Matt Mullenweg 查看详情

Below the table are four buttons: 插件名, 插件描述, 版本号, and 开发者信息. A large black button labeled '详情窗口' is positioned to the right of the second row. Red arrows point from the labels to their corresponding parts in the table rows.

当下载了一个插件不会使用时，可以点击「查看详情」，在详情页面查看插件的具体使用说明。

The screenshot shows a plugin detail page for 'Hello Dolly'. At the top is a black and white photograph of Louis Armstrong playing a trumpet. Below the photo, the plugin name 'Hello Dolly' is displayed in a large, bold, white font on a black background. The main content area has two tabs: '描述' (Description) and '评价' (Reviews), with '描述' currently selected. A yellow warning box contains the text: '警告：目前尚未测试该插件与您当前使用的WordPress版本的兼容性。' (Warning: The plugin has not been tested for compatibility with your current version of WordPress.) Below this, a text box states: '这不仅仅是一个插件，它象征着一代的热情和希望，在Louis Armstrong中最著名的词语：Hello, Dolly。启用后后，您将在每个管理页面的右上方看到Hello, Dolly中随机一个歌词。' (This is more than just a plugin; it symbolizes the passion and hope of a generation. In Louis Armstrong's most famous words: Hello, Dolly. Once activated, you will see a random lyric from Hello, Dolly on the right side of every admin page.) To the right of the description, there is a sidebar with plugin metadata: '版本: 1.6', '作者: Matt Mullenweg', '最近更新: 7月前', '需要WordPress版本: 4.6或更高', '兼容至: 4.7.8', '活跃安装: 一百万+', a link to the 'WordPress.org插件页面', and a link to the '插件主页'. Below this is a section titled '综合评级' (Overall Rating) showing a 5-star rating with 182 reviews. Further down are sections for '评论' (Comments) and a star rating distribution chart. The chart shows the following counts for each star rating: 5星 (62), 4星 (7), 3星 (5), 2星 (7), and 1星 (101). At the bottom of the sidebar is a '贡献者' (Contributors) section and a large blue button labeled '已安装最新版本' (Latest Version Installed).

通过详情窗口可以很方便的了解到一个插件的各方面信息，对于一些功能性的插件，会在这个页面提供使用说明、常见文件等内容，帮助你使用这个插件。



安装说明

将Akismet插件上传到您的博客，激活它，然后输入您的[Akismet.com API密钥](#)。

3, 2, 1: 大功告成！

除了对单独一个插件进行操作以外，还可以批量对插件执行操作，比如批量启用插件、批量停用插件等等。

至此，我们对于插件的基本操作，就解说了。

安装插件

安装插件一般来说，有三种方式。

1. 使用插件中心安装

一般来说，推荐大家使用插件中心安装，使用插件中心安装的好处是插件都是从WordPress官方的插件仓库中下载，无需担心自己下载到的插件是经过修改的。



点击「插件」—「安装插件」，就会进入到插件管理页面，点击「安装插件」，就会进入到插件中心。

添加插件 [上传插件](#)

特色 热门 推荐 收藏

关键字 搜索插件...

插件为WordPress添加新功能。您可以在您的控制板选择并直接安装WordPress插件目录中的插件，或者点击顶部的按钮上传。

 Akismet Anti-Spam Akismet会根据我们的全球垃圾评论数据库 检查您的评论和联系表单提交，以保护您的 站点免受恶意内容的侵害。 由Automatic	 Jetpack 由 WordPress.com 出品 一款用于统计、相关文章、搜索引擎优化、 社交共享、保护、备份、速度和电子邮件列 表管理的插件。 由Automatic	 WP Super Cache A very fast caching engine for WordPress that produces static html files. 由Automatic
 bbPress bbPress是以WordPress方式制作的论坛插 件。 由The bbPress Community	 BuddyPress BuddyPress adds community features to WordPress. Member Profiles, Activity Streams, Direct Messaging, Notifications, and more! 由The BuddyPress Community	 Theme Check A simple and ea... 由Otto42, pross
 bbPress bbPress是以WordPress方式制作的论坛插 件。 由The bbPress Community	 BuddyPress BuddyPress adds community features to WordPress. Member Profiles, Activity Streams, Direct Messaging, Notifications, and more! 由The BuddyPress Community	 Theme Check A simple and ea... 由Otto42, pross
 bbPress bbPress是以WordPress方式制作的论坛插 件。 由The bbPress Community	 BuddyPress BuddyPress adds community features to WordPress. Member Profiles, Activity Streams, Direct Messaging, Notifications, and more! 由The BuddyPress Community	 Theme Check A simple and ea... 由Otto42, pross

热门标签

您也可以浏览插件目录中最流行的标签：

```
admin ads adsense advertising affiliate ajax analytics api author buddypress button calendar categories category chat code comment comments contact contact form content css custom custom post type dashboard e-commerce ecommerce editor email embed events facebook feed form forms gallery google google analytics html image images javascript jquery lightbox link links list login marketing media menu meta mobile multisite navigation news newsletter notification page pages payment payment gateway photo photos plugins popup post posts redirect responsive rss search security seo share shortcode shortcodes sidebar simple slider slideshow social social media spam statistics stats tag tags theme tinyMCE tracking twitter url user users video widget widgets woocommerce youtube
```

在插件中心你可以在搜索框搜索要安装的插件名称，就能找到你要的插件，也可以点击底部的标签，选择合适你的插件。通过搜索，可以搜索到大量的插件。

The screenshot shows the WordPress plugin search results for 'qq'. It displays two plugin cards:

- Wechat Social login 微信QQ钉钉登录插件**: A plugin by '迅虎网络' with 4 stars from 4 reviews and over 1,000 active installations. It supports WeChat, QQ, and DingTalk. It was last updated 5 days ago and is compatible with the current WordPress version.
- Open Social**: A plugin by 'Afly' with 5 stars from 20 reviews and over 3,000 active installations. It allows users to log in or share with various social networks including QQ, WeiBo, Baidu, Google, Microsoft, DouBan, XiaoMi, WeChat, GitHub, Twitter, and Facebook using a single PHP API.

点击「更多详情」，确认这个插件是你需要的插件后，返回点击「现在安装」，就可以轻松的将这个插件安装到 WordPress 站点上。

还可以点击上方的「热门」、「推荐」查看当前阶段比较热门的插件来安装。

虽然有很多插件都可以使用，但是不建议大家安装过多的插件，会导致站点运行速度缓慢。根据自己的实际需要安装即可。

2. 使用后台上传进行安装

有些时候，可能拿到手的插件是一个 Zip 压缩包，比如如下的场景：

1. 你的主机无法与 WordPress 官方服务器进行通信，从 WordPress 官网下载了需要用的插件。自行安装。
2. 从 codecanyon 下载了一个付费插件，需要自己安装。
3. 从 Github 或其他代码托管平台下载了一个开源插件，需要自己安装。

这个时候，就要从 WordPress 的插件后台进行安装了。

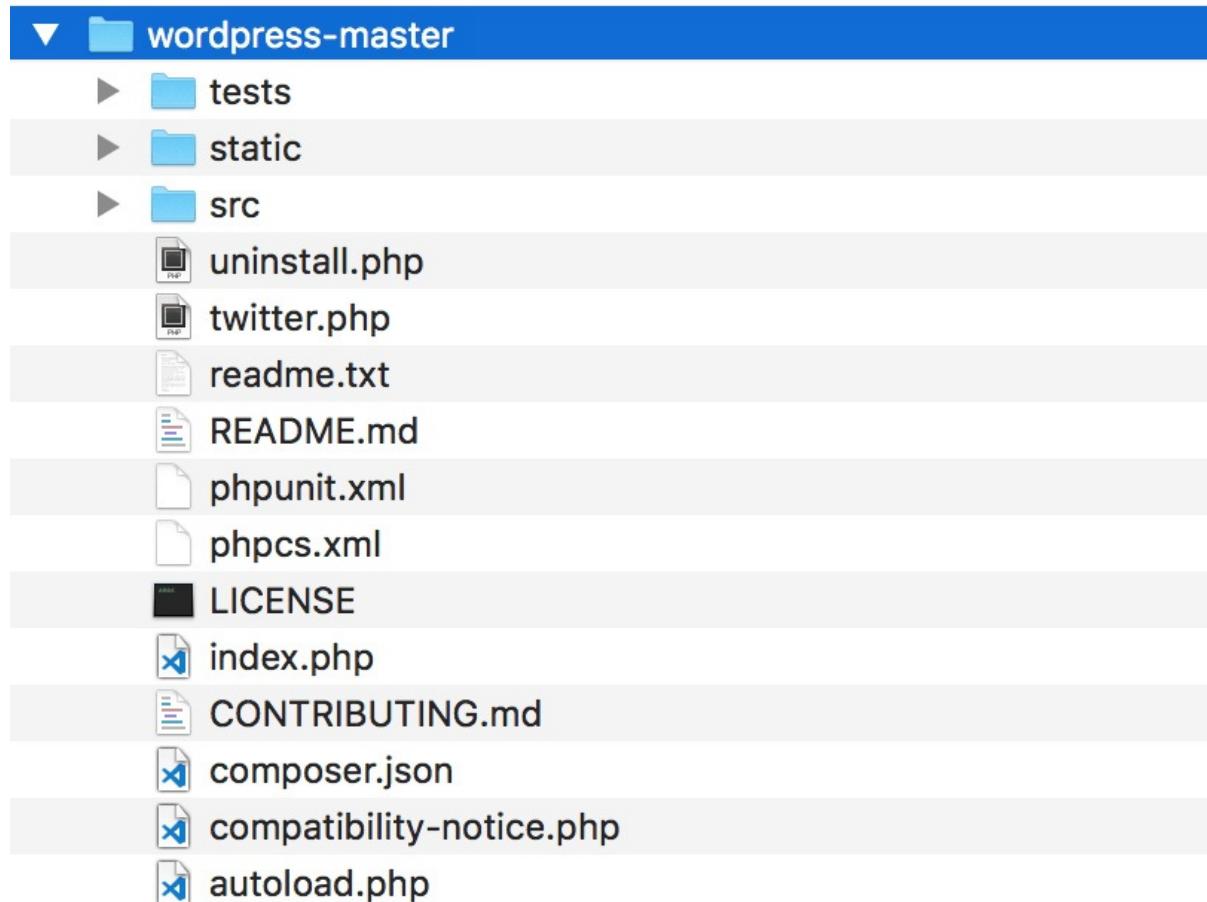
比如说，我下载了 <https://github.com/twitter/wordpress> 这个插件，下载到本地的是一个压缩包。



wordpress- master.zip

308 KB

将压缩包解压后，是这样的：



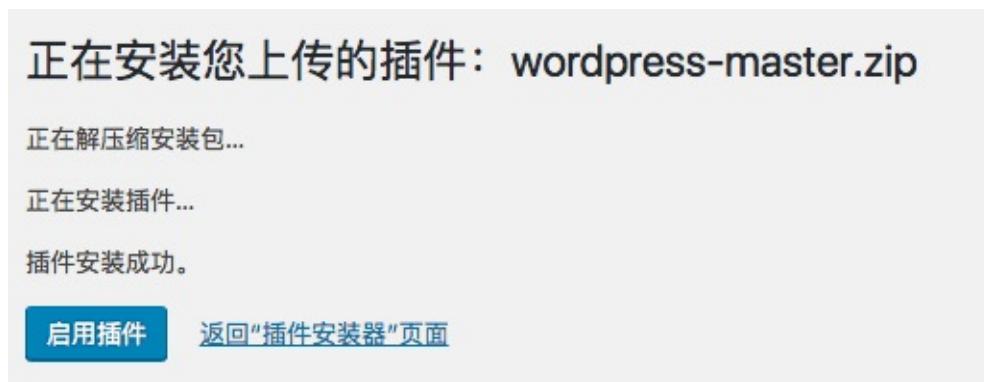
这时，进入 WordPress 后台，选择「插件」—「安装插件」，单击上方的「上传插件」按钮：



这时会弹出一个上传的输入框，点击输入框中的选择文件：

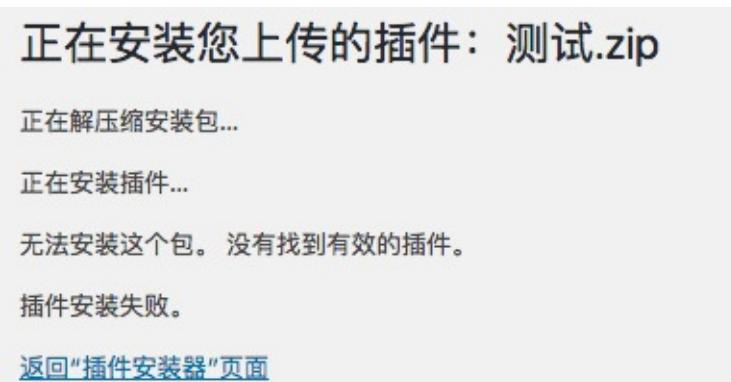


上传成功后，会看到这样的界面：

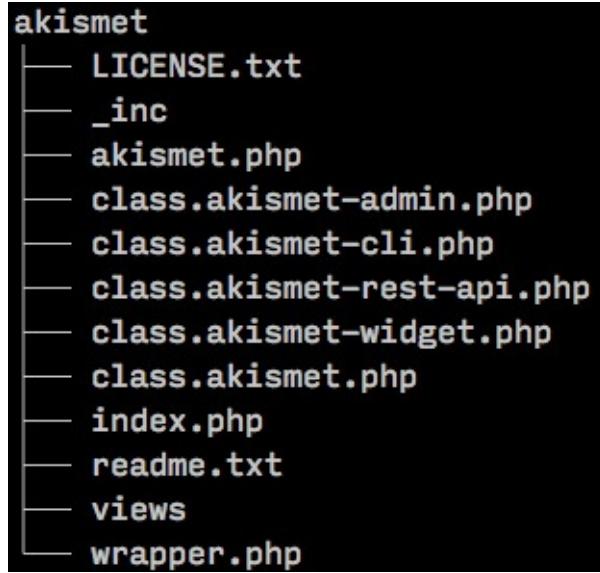


这就说明安装成功了，这时可以点击「启用插件」按钮来进行安装。

如果上传以后，提示没有找到有效的插件，就像下面这样：



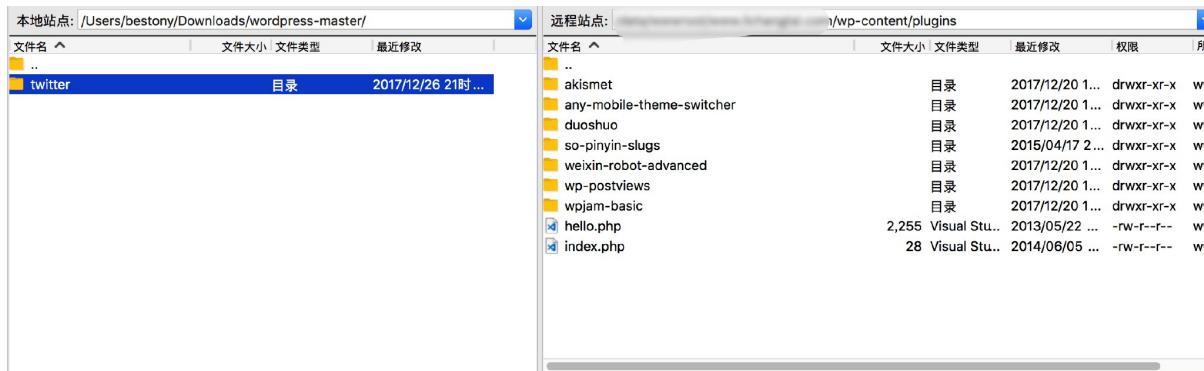
则可能是你的插件目录放的太深了，插件文件应该在压缩文件的根目录中。即下图，压缩包 akismet 的根目录中存放着插件文件。



3. 使用 FTP 进行上传安装

在有些特殊情况下，我们可能无法通过 WordPress 对目录进行写操作（安全控制），此时，可以使用系统自带的 FTP 进行上传。

1. 解压你下载好的插件到本地。
2. 打开 FTP 软件，登录到虚拟主机上，将目录切换到 WordPress 的根目录下。
3. 找到 `wp-content/plugins/` 目录。
4. 将整个插件文件夹上传到整个目录中。



这样，就可以将插件上传到 WordPress 当中，然后到 WordPress 后台去启用插件即可。

修改插件

在某些特殊的情况下，我们可能要对一个插件进行一定的修改，比如修改代码、添加 API key（一些涉及到了第三方服务的插件）等。这个时候，可以选择通过 FTP 进行修改，不过也可以通过 WordPress 后台进行修改。

进入 WordPress 后台，点击「插件」—「编辑」，就可以进入到插件编辑界面。



插件的编辑页面整体可以分为两部分，左侧的代码框和右侧的文件框。

正在编辑akismet/akismet.php (不活跃)
选择要编辑的插件: Akismet Anti-Spam

选择的文件内容:

```
1 <?php
2 /**
3  * @package Akismet
4 */
5 /*
6 Plugin Name: Akismet Anti-Spam
7 Plugin URI: https://akismet.com/
8 Description: Used by millions, Akismet is quite possibly the best
way in the world to protect your blog from spam.
It keeps your site protected even while you sleep. To get
started: activate the Akismet plugin and then go to your Akismet
Settings page to set up your API key.
9 Version: 4.0.2
10 Author: Automattic
11 Author URI: https://automattic.com/wordpress-plugins/
12 License: GPLv2 or later
13 Text Domain: akismet
14 */
15 /*
16 This program is free software; you can redistribute it and/or
17 modify it under the terms of the GNU General Public License
18 as published by the Free Software Foundation; either version 2
```

代码编辑区域

插件文件

[akismet.php](#)

[class.akismet-cli.php](#)

[class.akismet.php](#)

[index.php](#)

[class.akismet-widget.php](#)

[class.akismet-admin.php](#)

[class.akismet-rest-api.php](#)

[readme.txt](#)

[_inc ▶](#)

[wrapper.php](#)

[LICENSE.txt](#)

[views ▶](#)

文件列表

文档:

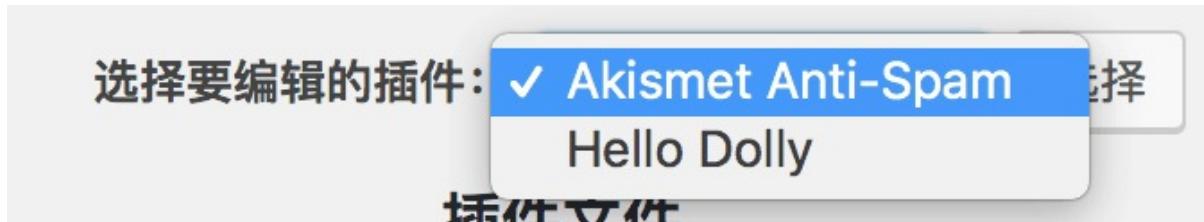
更新文件

在右侧点击文件，左侧就会自动加载相关文件，然后就可以在左侧的代码编辑区域修改对应的文件内容。

针对 PHP 文件中的函数，WordPress 还提供了方便的函数查询的功能，点击下方的文档后的下拉框，会看到这个文件使用的具体的函数，当选择了一个函数后，点击后面的查询按钮，就会自动跳转到这个函数的说明页面。十分的方便。



如果进入后，你发现当前加载的插件不是你需要修改的插件，你可以点击右上角的插件列表进行修改。



点击下拉菜单选择合适的插件，并点击选择，就会跳转到对应的插件了。

插件是如何运行的

插件的加载机制

了解插件的加载机制，能够帮助我们更加深入的了解插件的开发，也能够帮助我们更好的对插件进行开发。下面了解一下 WordPress 的插件加载机制。

要注意，PHP 的执行顺序是一行一行向下执行的，方便理解后面的加载顺序判断。

index.php

无论从哪里进入到 WordPress，一定会从 *index.php* 文件开始加载，打开根目录的 *index.php*，可以看到这个文件引用了 *wp-blog-header.php* 文件。再无其他代码。

代码所在行[请单击这里](#)。

```
1 <?php
2 /**
3 * Front to the WordPress application. This file doesn't do anything, but loads
4 * wp-blog-header.php which does and tells WordPress to load the theme.
5 *
6 * @package WordPress
7 */
8
9 /**
10 * Tells WordPress to load the WordPress theme and output it.
11 *
12 * @var bool
13 */
14 define('WP_USE_THEMES', true);
15
16 /** Loads the WordPress Environment and Template */
17 require( dirname( __FILE__ ) . '/wp-blog-header.php' );
18
```

wp-blog-header.php

打开 *wp-blog-header.php* 文件，可以看到这个文件引用了两个文件。

代码所在行[请单击这里](#)。

```
1  <?php
2  /**
3   * Loads the WordPress environment and template.
4   *
5   * @package WordPress
6   */
7
8  if (!isset($wp_did_header)) {
9
10    $wp_did_header = true;
11
12    // Load the WordPress library.
13    require_once( dirname(__FILE__) . '/wp-load.php' );
14
15    // Set up the WordPress query.
16    wp();
17
18    // Load the theme template.
19    require_once(ABSPATH . WPINC . '/template-loader.php' );
20
21 }
22 }
```

我们先看一看 `wp-load.php` 文件。

wp-load.php

打开这个文件可以看到，该文件检测了 `wp-config.php` 文件，如果未检测到该文件，则进入到安装进程。我们使用时这个文件是存在的，所以执行的是下方的 `require_once(ABSPATH . 'wp-config.php');` 命令。

代码所在行[请单击这里](#)。

```

1  <?php
2  /**
3   ** Define ABSPATH as this file's directory */
4  if ( ! defined( 'ABSPATH' ) ) { ...
5
6  /**
7   * ...
8  if ( file_exists( ABSPATH . 'wp-config.php' ) ) { ← 检测文件是否存在
9
10    /** The config file resides in ABSPATH */
11    require_once( ABSPATH . 'wp-config.php' );
12
13 } elseif ( @file_exists( dirname( ABSPATH ) . '/wp-config.php' ) && ! @file_exists( dirname( ABSPATH ) .
14
15    /** The config file resides one level above ABSPATH but is not part of another installation */
16    require_once( dirname( ABSPATH ) . '/wp-config.php' );
17
18 } else {
19
20    // A config file doesn't exist → 如果不存在则进入安装进程
21
22    define( 'WPINC', 'wp-includes' );
23    require_once( ABSPATH . WPINC . '/load.php' );
24
25    // Standardize $_SERVER variables across setups.
26    wp_fix_server_vars();
27
28    require_once( ABSPATH . WPINC . '/functions.php' );
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

```

接下来看看 *wp-config.php* 这个文件。

wp-config.php

wp-config.php 文件内存放着 WordPress 的配置信息，包括数据库、Debug 等信息。我们一直往下看，会看到这里它引入到了一个 *wp-settings.php*。

代码所在行[请单击这里](#)。

```

wp wp-config.php ×

81  /**
82  * zh_CN本地化设置：启用ICP备案号显示
83  *
84  * 可在设置→常规中修改。
85  * 如需禁用，请移除或注释掉本行。
86  */
87  define('WP_ZH_CN_ICP_NUM', true);
88
89 /* 好了！请不要再继续编辑。请保存本文件。使用愉快！ */
90
91 /** WordPress目录的绝对路径。 */
92 if ( !defined('ABSPATH') )
93 |   define('ABSPATH', dirname(__FILE__) . '/');
94
95 /** 设置WordPress变量和包含文件。 */
96 require_once(ABSPATH . 'wp-settings.php');
97

```

接下来看看 `wp-settings.php`。

wp-settings.php

这里包含了大量的 WordPress 初始化函数，前面的内容不做过多的解释，可以直接拉到第 303 行，则看到这样的代码。

[代码所在行请单击这里。](#)

```

302 // Load active plugins.
303 foreach ( wp_get_active_and_valid_plugins() as $plugin ) {
304 |   wp_register_plugin_realpath( $plugin );
305 |   include_once( $plugin );
306 }
307 unset( $plugin );

```

这段代码会将已激活的插件作为循环元素进行循环，对每个插件执行注册和加载。

继续向下看，会在第 421 行看到这样一段代码，该代码实现了加载我们激活的主题的 `functions.php` 文件。

[代码所在行请单击这里。](#)

```

421 // Load the functions for the active theme, for both parent and child theme if applicable.
422 if ( ! wp_installing() || 'wp-activate.php' === $pagenow ) {
423 |   if ( TEMPLATEPATH !== STYLESHEETPATH && file_exists( STYLESHEETPATH . '/functions.php' ) )
424 |     include( STYLESHEETPATH . '/functions.php' );
425 |   if ( file_exists( TEMPLATEPATH . '/functions.php' ) )
426 |     include( TEMPLATEPATH . '/functions.php' );
427 }

```

这样，我们在主题中添加的功能，也会被加载到系统当中去。

这个文件没有再引用其他的文件，然后返回 `wp-config.php`。

wp-blog-header.php

我们发现，`wp-config.php` 也执行到了文件的尾部，接下来返回到上一个文件——`wp-load.php`。

在 `wp-load.php` 中可以看到，在这个 if 代码块中，只有这一条引用代码，所以该文件也执行完了，代码执行回到上一个文件 `wp-blog-header.php`。

在 `wp-blog-header.php` 文件中可以看到，在 `wp-load.php` 下，引用了 `template-loader.php`，该文件的功能就是进行我们后续的主题加载。

```
if ( !isset($wp_did_header) ) {

    $wp_did_header = true;

    // Load the WordPress library.
    require_once( dirname(__FILE__) . '/wp-load.php' );

    // Set up the WordPress query.
    wp();

    // Load the theme template.
    require_once(ABSPATH . WPINC . '/template-loader.php');

}
```

结论

经过上述的分析可以得出一个结论，WordPress 的加载是这样的一个顺序，先加载 插件，再加载 主题根目录中的 `functions.php`，最后加载主题。



插件是如何运行的

WordPress 的插件也是一段代码，WordPress 插件通过 WordPress 提供的插件 API 和函数，来实现集成到 WordPress 当中去。相关涉及到的函数，你可以在 `wp-includes/plugin.php` 中看到：

```

  plugin.php x
105 /**
106  * function add_filter( $tag, $function_to_add, $priority = 10, $accepted_args = 1 ) { ...
113 }
114
115 /**
116 * Check if any filter has been registered for a hook.
117 *
118 * @since 2.5.0
119 *
120 * @global array $wp_filter Stores all of the filters.
121 *
122 * @param string      $tag           The name of the filter hook.
123 * @param callable|bool $function_to_check Optional. The callback to check for. Default false.
124 * @return false|int If $function_to_check is omitted, returns boolean for whether the hook has
125 *                   anything registered. When checking a specific function, the priority of that
126 *                   hook is returned, or false if the function is not attached. When using the
127 *                   $function_to_check argument, this function may return a non-boolean value
128 *                   that evaluates to false (e.g.) 0, so use the === operator for testing the
129 *                   return value.
130 */
131 function has_filter($tag, $function_to_check = false) { ...
139 }
140
141 /**
142 * Call the functions added to a filter hook.
143 *
144 * The callback functions attached to filter hook $tag are invoked by calling
145 * this function. This function can be used to create a new filter hook by
146 * simply calling this function with the name of the new hook specified using
147 * the $tag parameter.
148 *
149 * The function allows for additional arguments to be added and passed to hooks.
150 */

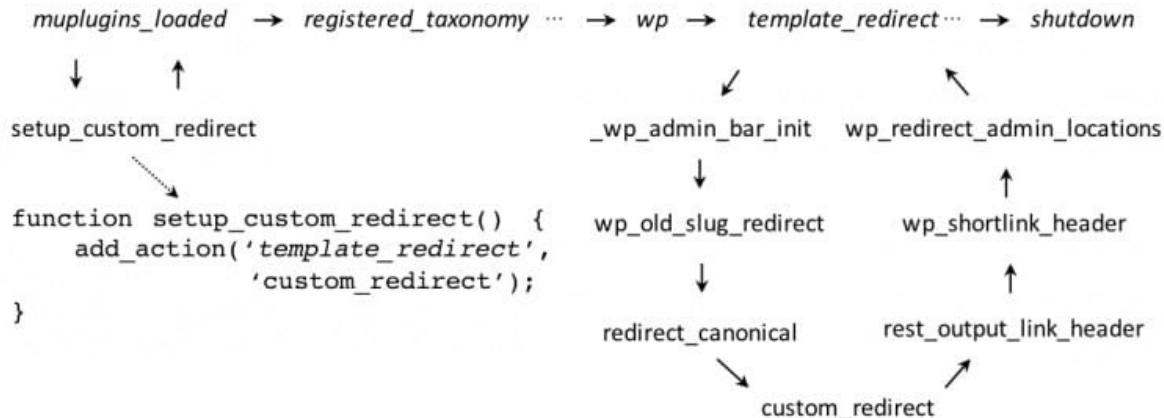
```

WordPress 将插件的行为分为两种，分别是 Action（动作）和 Filters（过滤器）。

什么是 Hook

WordPress 的插件机制是基于 Hook 机制实现的，简单来说，就是在 WordPress 的核心代码中，加入了大量让 Hook 挂载的位置，当程序执行到这里时，就会顺着 Hook 上挂载的钩子走一遍，然后再回来继续执行后面的代码；如果没有挂载任何钩子，则不执行操作。

我们写插件，就是写一些自定义的函数，然后利用 WordPress 提供的对 Hook 操作的函数，将函数挂载到对应的位置上去。



什么是 Action

Action 会在 WordPress 内核运行到一定的点或事件时会调用的，被调用后，插件可以执行一些具体的操作。

什么是 Filters

Filters 则是对内容的处理，通过 Filters 可以在这些数据被渲染到页面前或保存到数据库前对其进行修改。

简单的来说，两者的区别主要是下面这样的：

- Action 函数在被调用时，是可以直接调用的，自定义的函数无需提供返回值；
- Filters 函数在被调用时，会被传入一个具体的字符串，函数执行完成后，还需要将这个字符串返回给系统，方便系统进行渲染。

重要的函数

刚刚只是说明了什么是 Action、什么是 Filters 具体功能的实现，则是通过下面这四个重要的函数来实现的。

do_action

do_action 是 WordPress 插件机制非常重要的一环，当程序运行到这个函数时，就会将挂载在这个 Hook 上的所有函数执行一遍。

这个函数有两个参数，第一个参数是 Hook 的名称，第二个参数则是具体的参数。

```

general-template.php ×

13 * specialised header will be included.
14 *
15 * For the parameter, if the file is called "header-spe
16 * "special".
17 *
18 * @since 1.5.0
19 *
20 * @param string $name The name of the specialised head
21 */
22 function get_header( $name = null ) {
23     /**
24      * Fires before the header template file is loaded.
25      *
26      * @since 2.1.0
27      * @since 2.8.0 $name parameter added.
28      *
29      * @param string|null $name Name of the specific he
30     */
31     do_action( 'get_header', $name );
32
33     $templates = array();
34     $name = (string) $name;
35     if ( '' !== $name ) {
36         $templates[] = "header-{$name}.php";
37     }
38
39     $templates[] = 'header.php';
40
41     locate_template( $templates, true );
42 }
43

```

比如在开发过程中，可能会用到 `get_header` 这个函数，该函数在执行时，首先会调用 `do_action`，那么，在 `get_header` 上挂载的函数就会执行，执行完成后，再执行后面的函数。

有了这个函数的存在，才有了后续我们开发插件时，各种功能的实现。

add_action

`add_action` 可以将我们自定义的函数加到特定的 Hook 上去，等待执行。一般来说，我们只需要执行如下命令即可。

```
add_action("Hook 名", "函数名")
```

不过这样的使用忽略掉了两个参数，在执行一些特定的操作时，可能就不足了。

我们可以看看这个函数的官方文档。

```
add_action( string $tag, callable $function_to_add, int $priority = 10, int $accepted_args = 1 )
```

这个函数一共有四个参数，其中前两者分别是 Hook 名和对应的函数，而后两者分别是优先级和可以接受的参数个数。

优先级：如果你不设置的话，添加的函数默认优先级是 10，也可以根据你自己的需要设置具体的优先级。优先级的规则是越小越先执行。同时添加了一个优先级为 5 和一个优先级为 10 的函数，会先执行优先级为5的函数，再执行优先级为10的函数。

参数个数：默认情况下参数的个数是1。具体设置为多少，则取决于你使用的 Hook 能够提供多少个参数。如何找寻这个参数的个数？可以直接去看这个 Hook 的说明，查看相关的代码说明。如果这个 Hook 提供了调用的说明，可以直接根据上面的 `add_action` 的函数参数来判断这个 Hook 支持几个参数。具体的参数是什么，可以通过实例代码的注释或者到源代码中查找对应 `do_action` 函数的参数。举个例子，我希望了解 `save_post` 这个 Hook 的参数，可以到它的 [API 页面](#)去查看具体的参数调用。可以看到在示例代码中说明了参数的个数和对应的参数的含义。

安全 | https://codex.wordpress.org/Plugin_API/Action_Reference/save_post

```
// Send email to admin.
wp_mail( 'admin@example.com', $subject, $message );
}
add_action( 'save_post', 'my_project_updated_send_email' );
```

Custom Post Type: 'book'

Suppose you have a 'book' custom post type and you add the book author, publisher and whether or not the book is in print when editing. Here's how you could save this information as metadata:

```
/**
 * Save post metadata when a post is saved.
 *
 * @param int $post_id The post ID.
 * @param post $post The post object.
 * @param bool $update Whether this is an existing post being updated or not.
 */
function save_book_meta( $post_id, $post, $update ) {
    /*
     * In production code, $slug should be set only once in the plugin,
     * preferably as a class property, rather than in each function that needs it.
     */
    $post_type = get_post_type($post_id);

    // If this isn't a 'book' post, don't update it.
    if ( "book" != $post_type ) return;

    // - Update the post's metadata.

    if ( isset( $_POST['book_author'] ) ) {
        update_post_meta( $post_id, 'book_author', sanitize_text_field( $_POST['book_author'] ) );
    }

    if ( isset( $_POST['publisher'] ) ) {
        update_post_meta( $post_id, 'publisher', sanitize_text_field( $_POST['publisher'] ) );
    }

    // Checkboxes are present if checked, absent if not.
    if ( isset( $_POST['inprint'] ) ) {
        update_post_meta( $post_id, 'inprint', TRUE );
    } else {
        update_post_meta( $post_id, 'inprint', FALSE );
    }
}

add_action( 'save_post', 'save_book_meta', 10, 3 );
```

See also [quick_edit_custom_box: Creating Inputs](#).

如果这个函数没有说明参数，也可以查看它的源代码，确认参赛，页面底部有说明这个函数的位置，

Change Log

- Since 2.0
- Moved from `wp-includes/functions-post.php` to `wp-includes/post.php` in 2.1
- Triggered separately by `wp_publish_post` in 2.3 (before this version, `wp_publish_post` called `wp_insert_post`).

Source File

Triggered by `wp_insert_post` and `wp_publish_post` in `wp-includes/post.php`

你可以直接去对应的文件查看这个 Hook 的调用参数。了解到参数后，就可以使用这些参数，来执行一系列操作了。

虽然你可能查到了有三个参数，可以不使用三个参数，比如只使用前两个。但是如果要用第一个和第三个，则还是需要在 `add_action` 中设置参数个数为3。

apply_filters

`apply_filters` 函数和 `do_action` 类似，它会对数据执行一系列操作来进行处理。系统通过如下方法进行调用，

```
$value = apply_filters( 'example_filter', 'filter me', $arg1, $arg2 );
```

可以看到这个函数的结果会被赋值给一个变量，所以它和 `do_action` 不同之处就在于它需要一个返回值。其他大体上相同，不再赘述。

add_filter

这个函数和刚刚说的 `add_action` 基本相同，函数的定义也相同。

```
add_filter( string $tag, callable $function_to_add, int $priority = 10, int $accepted_args = 1 )
```

这个函数的使用不再赘述，可以参考上面 `add_action` 的内容。

Hook 列表

WordPress 提供了大量的 Hook，这里不再一一列举，把官方的列表给大家，在使用中出现了问题，再到读者圈内提问即可。

[Action Hook 列表](#)

[Filter Hook 列表](#)

创建一个插件

插件开发目标

这节课来尝试创建一个插件，实现在文章尾部输出一段简单的文字。

插件开发过程

给插件起名

既然要创建一个插件，那么必然就要给插件起一个名字，在起名时，需要注意以下三点：

- (1) 插件名唯一：我们的插件名称应该是唯一的，这样才不会在加载时出现错误。同时也可以不用担心上传到 WordPress 官方中浏览出错(WordPress 官方给每个插件有一个单页，单页地址和你的插件名称有关)。



Mailgun for WordPress

By Mailgun

- (2) 插件名可识别：插件名称应该尽可能的可读，通过简单的读插件的名称，就能知道大体上这个插件的功能。

- (3) 如果名字已经重复了，可以使用前缀来区分，比如 `gitchat_` 来区分插件和其他人的插件。

创建一个插件

1. 创建插件目录

打开开发环境中的 WordPress，进入 `wp-content/plugins` 目录，创建一个新的目录 `gitchat_copyright` 目录。

为了避免冲突，使用 `gitchat_` 作为插件前缀。

2. 创建插件文件，并初始化代码

进入 `copyright` 目录，创建一个文件 `gitchat_copyright.php`，并在其中添加如下代码。

这里创建的 `gitchat_copyright.php` 是插件的入口文件。有了入口文件，WordPress 才能识别出插件和对应的功能。

如果是 Windows 系统，保存时请确保以 UTF-8 编码保存代码。

```
<?php
/*
Plugin Name: Copyright
Plugin URI: https://gitchat.cn/plugins
Description: 这是一个 copyright 插件，能够在文章尾部添加内容。
Version: 1.0
Author: Gitchat
Author URI: https://gitchat.cn/
*/
```

保存后退出。

这段代码，是 WordPress 标准的插件信息头，通过这段代码，WordPress 就可以自动识别出我们的插件具体信息。具体的说明，可以在 [这里](#) 找到相关信息。

这时，回到 WordPress 后台，可以在插件列表中看到这样的界面。



这就是我们创建的插件。WordPress 会自动读取我们在注释中填写的内容，加载为插件的详细信息，可以把这段信息改为你自己的对应信息。

添加插件代码

1. 分析功能

我们希望实现的功能是在文章的尾部加入一些内容，则需要对 `the_content` 函数进行修改，实现在内容后输出特定的内容。

2. 代码实现

在原有代码后加入如下代码，来实现具体的功能：

```
function copyright_end($content){
    if(is_single()){
        $content .= '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>';
        return $content;
    }
}
add_filter( "the_content", "copyright_end" );
```

添加后，代码如下：

```

1  <?php
2  /*
3  Plugin Name: Copyright
4  Plugin URI: https://gitchat.cn/plugins
5  Description: 这是一个 copyright 插件，能够在文章尾部添加内容。
6  Version: 1.0
7  Author: Gitchat
8  Author URI: https://gitchat.cn/
9  */
10
11 function copyright_end($content){
12     if(is_single()){
13         $content .= '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>';
14         return $content;
15     }
16 }
17 add_filter( "the_content", "copyright_end" );

```

保存文件并退出。

3. 启动测试

回到 WordPress 的后台，进入「插件」—「已安装的插件」，找到刚刚创建的「Copyright」插件，并启用。

启用成功后，回到首页，找一篇文章，点击进去，可以看到如图的样式。

2017年12月14日 由ADMIN

世界，您好！

欢迎使用WordPress。这是您的第一篇文章。编辑或删除它，然后开始写作吧！

这是一个来自 GitChat 达人课的插件

[编辑](#)

这说明我们的插件功能已经实现了。

4. 代码解读

在这段代码中定义了 `copyright_end` 函数，来实现对输出内容的修改。

同时，我在下方调用 `the_content` 过滤器，实现对内容的修改嵌入。

这里特别需要注意的是，在函数中调用了 `is_single` 函数，该函数是用来判断当前页面是否是文章页面。

如果注销这个 if 判断条件，那么在首页可能也会看到这个输出的结果。

文章

2017年12月14日 编辑

世界，您好！

欢迎使用WordPress。这是您的第一篇文章。编辑或删除它，然后开始写作吧！

这是一个来自 GitChat 达人课的插件

搜索...



近期文章

世界，您好！

近期评论

一位WordPress评论者发表在《世界，您好！》

优化插件

目前我们的插件并不好用，因为输出的内容是在代码中写死的，对于开发者来说，无伤大雅，但插件的使用者不可能只是开发者，所以需要对插件进行优化，让输出的内容可以在后台进行修改。

实现初始化方法

既然要允许在后台修改我们要输出的内容，那么就需要将用户设置的输出内容进行设置，允许用户输出内容。最简单的，就是将数据存储在 WordPress 的设置表，和站点设置放在一起。

这就需要我们在插件启用时，对这个数据进行初始化。这里用到了 WordPress 提供的 `register_activation_hook`。

如果你放在普通过程中，会发现你的设置内容会被不停的初始化。

在插件中添加如下代码：

```
/**
 * 初始化设置项
 */
function gitchat_copyright_activate() {
    add_option('gitchat_copyright_code', '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>');
}
register_activation_hook( __FILE__, 'gitchat_copyright_activate' );
```

添加后的代码如下：

```

1  <?php
2  /*
3  Plugin Name: Copyright
4  Plugin URI: https://gitchat.cn/plugins
5  Description: 这是一个 copyright 插件，能够在文章尾部添加内容。
6  Version: 1.0
7  Author: Gitchat
8  Author URI: https://gitchat.cn/
9  */
10
11 /**
12 * 初始化设置项
13 */
14 function gitchat_copyright_activate() {
15     add_option('gitchat_copyright_code', '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>');
16 }
17 register_activation_hook( __FILE__, 'gitchat_copyright_activate' );
18
19 /**
20 * 判断当前是否是文章页面，如果是则添加 copyright 输出。
21 */
22 function copyright_end($content){
23     if(is_single()){
24         $content .= '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>';
25         return $content;
26     }
27 }
28 add_filter( "the_content", "copyright_end" );

```

保存插件，回到后台停用插件，并重新启用。

重新启用成功后，就可以在数据库中找到对应的选项，比如我们查询一下看一下。

```
select * from wp_options where option_name = 'gitchat_copyright_code'
```

如何执行上述操作？

如果你是 macOS 操作系统，可以下载 Sequel Pro。使用 Sequel Pro 链接到数据库，并进行管理。

如果你是 Windows 操作系统，可以使用 phpStudy 自带的 phpMyAdmin 来进行管理，访问 <http://localhost/phpmyadmin> 即可。

```
1 select * from wp_options where option_name = 'gitchat_copyright_code'
```

option_id	option_name	option_value	autoload
216	gitchat_copyright_code	<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>	yes

现在我们内容已经放在数据库了。接下来设置内容的输出，让输出使用数据库中的值，而不是写死在代码中。

将第24行的：

```
$content .= '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>';
```

改为：

```
$content .= get_option('gitchat_copyright_code');
```

保存文件。回到网站首页，打开一个文章页，可以看到刚刚设置的效果依然还在。

接下来修改一下数据库中的值，来测试一下输出的是否是我们存放在数据中的内容。

在数据库中执行如下代码：

```
update wp_options set option_value = "<hr><p>现在输出的是数据库中的内容了</p><hr>" where option_name = 'gitchat_copyright_code'
```

刷新刚刚的文章页面，可以看到输出的内容已经发生了改变。

2017年12月14日 由ADMIN

世界，您好！

欢迎使用WordPress。这是您的第一篇文章。编辑或删除它，然后开始写作吧！

现在输出的是数据库中的内容了

[编辑](#)

实现后台的管理

虽然现在实现了参数的数据存储，但是目前的方式依然非常不方便，需要在数据库管理工具的帮助下才能设置输出内容，极为不方便。

接下来把这一项放在系统后台的设置项目中。

在插件中添加如下代码：

```
/**
 * 菜单项注册
 */
$new_general_setting = new new_general_setting();
```

```

class new_general_setting {
    function new_general_setting( ) {
        add_filter( 'admin_init' , array( &$this , 'register_fields' ) );
    }
    function register_fields() {
        register_setting( 'general' , 'gitchat_copyright_code');
        add_settings_field('fav_color', '<label for="gitchat_copyright_code">Copyright 代码</label>' , array(&$this,
'fields_html') , 'general' );
    }
    function fields_html() {
        $value = get_option( 'gitchat_copyright_code' , '' );
        echo '<textarea name="gitchat_copyright_code" id="gitchat_copyright_code" cols="30" rows="10">'. $value. '<
/textarea>';
    }
}

```

添加后的代码如下：

```

1  <?php
2  /*
3  Plugin Name: Copyright
4  Plugin URI: https://gitchat.cn/plugins
5  Description: 这是一个 copyright 插件，能够在文章尾部添加内容。
6  Version: 1.0
7  Author: Gitchat
8  Author URI: https://gitchat.cn/
9  */
10
11 /**
12  * @since 1.0
13  *
14  * Function gitchat_copyright_activate()
15  *     add_option('gitchat_copyright_code', '<hr><p>这是一个来自 GitChat 达人课的插件</p><hr>');
16  */
17 register_activation_hook( __FILE__ , 'gitchat_copyright_activate' );
18
19 /**
20  * New General Setting
21  */
22 $new_general_setting = new new_general_setting();
23 class new_general_setting {
24     function new_general_setting( ) {
25         add_filter( 'admin_init' , array( &$this , 'register_fields' ) );
26     }
27     function register_fields() {
28         register_setting( 'general' , 'gitchat_copyright_code');
29         add_settings_field('fav_color', '<label for="gitchat_copyright_code">Copyright 代码</label>' ,
'fields_html') , 'general' );
30     }
31     function fields_html() {
32         $value = get_option( 'gitchat_copyright_code' , '' );
33         echo '<textarea name="gitchat_copyright_code" id="gitchat_copyright_code" cols="30" rows="10">'. $value. '<
/textarea>';
34     }
35 }

```

这段代码构建了一个新的类，在类中通过 `register_fields` 方法来注入新的设置项，并通过 `fields_html` 设置了输入框的类型。

回到 WordPress 停用插件并重新启用插件。进入「设置」 - 「常规」：



拉到底部，可以看到这样一项：

Copyright 代码

<hr>现在输出的是数据库中的内容了<hr>

仅对WordPress自带主题有效。

保存更改

这时可以修改输入框里的代码并保存。回到文章页后刷新，会发现你的 copyright 代码已经更新了。

这样就实现了在后台的管理。

实现停用方法

有些时候可能需要记录用户对插件的停用，比如当用户停用了插件后，向服务端发送情况，停掉某些特定服务等等。

这部分，可以使用 WordPress 的 `register_deactivation_hook` 来实现。

这个函数的使用方法和刚刚提到的 `register_activation_hook` 差不多，不再举例说明这里给出一个示例代码，大家根据自己的需要进行设置。

```
function gitchat_copyright_deactivate() {
    // 具体的停用插件的逻辑
}
register_deactivation_hook( __FILE__, 'gitchat_copyright_deactivate' );
```

实现卸载方法

我们在启用插件时，在数据库中加入了一些字段，当卸载这个插件时，最好将添加的数据库字段删除，不然的话，长期以往会导致我们的数据库中存在大量无用的字段。

在插件目录下创建一个 `uninstall.php`，在其中加入如下代码：

```
<?php
// part 1
if(!defined("WP_UNINSTALL_PLUGIN"))
    exit();
// part 2
delete_option("gitchat_copyright_code");
```

上述这段代码可以分为两个部分，第一部分是 WordPress 的卸载插件检测，在卸载时，会设置 `WP_UNINSTALL_PLUGIN`。如果检测不到，就退出，以确保这个程序是被合理的调用，而不是会是被直接访问而调用，导致我们的设置丢失。

第二部分则是从数据库中删除这个数据项。以达到删除无用字段的目的。当然，如果你的插件十分复杂，也可以在这里写具体的卸载逻辑。

结论

至此，我们创建了一个非常简单的插件，这个插件只需要寥寥数行代码，就可以实现在整站的所有内容的尾部加入一段版权代码。

[单击这里下载本课插件。](#)

插件后台的设计与开发

对于简单的插件来说，可能无需设置，安装插件后启用功能即可。但是对于一些功能更复杂的插件，我们就需要为插件添加一个后台页面了，这节课我们来学习如何为插件设计、创建一个后台。

学习目标

通过这节课的学习将会为 gitchat_copyright 插件创建一个后台管理页面。

添加菜单

添加菜单项目

添加一级菜单

既然要设置菜单，我们肯定要有一个入口能够进入到插件的页面去。在上节课尝试使用 WordPress 自己的设置页面作为入口，但是这节课我们要自己创建一个页面，所以要添加自己的菜单项目。

这里要用到 `add_menu_page` 函数，这个函数的定义如下：

```
add_menu_page( string $page_title, string $menu_title, string $capability, string $menu_slug, callable $function = null, string $icon_url = '', int $position = null )
```

我们需要设置这7个参数，各个参数的含义如下：

- `page_title`: 页面的标题，会和 `title` 标签内显示的一样。
- `menu_title`: 在控制面板中显示的菜单名称。
- `capability`: 显示该操作项所需要的最低权限。具体权限可以参考 [官方文档](#)。
- `menu_slug`: 菜单别名，需要是唯一的。
- `Function`: 显示该页面内容时调用的函数。
- `icon_url`: 菜单中图标的 url。这个参数除了粘贴 url，还可以粘贴 WordPress 官方的 helper css，比如 `dashicons-chart-pie`，具体可以在[这里](#)找到。或者是 base64 后的图标 `data:image/svg+xml;base64`。
- `position`: 出现在菜单中的位置。具体的 Position 可以参考 [官方说明](#)，需要注意，我们自己定义的 position 应该和官方的 position 不同，避免冲突。

- 2 – Dashboard
- 4 – Separator
- 5 – Posts
- 10 – Media
- 15 – Links
- 20 – Pages
- 25 – Comments
- 59 – Separator
- 60 – Appearance
- 65 – Plugins
- 70 – Users
- 75 – Tools
- 80 – Settings
- 99 – Separator

接下来自定义一个一级菜单。

在我们的插件尾部添加如下代码：

```
/**
 * 注册菜单项
 */
function gitchat_copyright_custom_menu(){
    add_menu_page(
        'Gitchat 版权插件首页',
        'GitChat 版权插件',
        'manage_options',
        'gitchat_optionpage',
        'gitchat_custom_page',
        'dashicons-admin-generic',
        100
    );
}
add_action( 'admin_menu', 'gitchat_copyright_custom_menu' );

function gitchat_custom_page(){
    ?>
    <h1>GitChat 版权插件测试</h1>
    <?php
}
```

这里我们定义了一个页面 title 是 *Gitchat* 版权插件首页，菜单名为 *GitChat* 版权插件，权限为管理员权限，别名为 *gitchat_optionpage*，图标为 *dashicons-admin-generic*，放在所有菜单底部的插件。

需要注意的是，菜单的添加必须挂载在 *admin_menu* 这个 hook 上才行。



点击这个菜单项，就会进入到我们的页面了，可以看到定义的页面内容。

A screenshot of a WordPress page titled 'GitChat 版权插件测试'. The page content area is currently empty. On the left, the same dark-themed sidebar is visible, showing the custom menu item 'GitChat 版权插件'.

添加二级菜单

有些时候，一级菜单并不够我们用，可能需要两级菜单。添加二级菜单，我们需要用到 `add_submenu_page` 方法。

```
add_submenu_page( string $parent_slug, string $page_title, string $menu_title, string $capability, string $menu_slug, callable $function = '' )
```

这个函数的定义和 `add_menu_page` 基本相同，唯一不同的是需要传入第一个参数，指定上级菜单，如果不指定，系统就不知道子菜单的上一级菜单是什么。

接下来，我们来添加一个子菜单。

在刚刚定义的 `gitchat_copyright_custom_menu` 函数中添加我们的子菜单函数。

新的代码如下：

```
/**
 * 注册菜单项
 */
function gitchat_copyright_custom_menu(){
    add_menu_page(
        'Gitchat 版权插件首页',
        'GitChat 版权插件',
        'manage_options',
        'gitchat_optionpage',
        'gitchat_custom_page',
        'dashicons-admin-generic',
        100
    );
    add_submenu_page(
        'gitchat_optionpage',
        "关于",
        "关于",
        'manage_options',
        'gitchat_aboutpage',
        'gitchat_about_page'
    );
}
add_action( 'admin_menu', 'gitchat_copyright_custom_menu' );

function gitchat_custom_page(){
    ?>
    <h1>GitChat 版权插件测试</h1>
    <?php
}

function gitchat_about_page(){
    ?>
    <h1>关于 GitChat</h1>
    <?php
}
```

这时刷新一下页面，就可以看到菜单加入了二级菜单。



点击其中的关于，就可以看到新加的页面了。

添加页面

使用 WordPress 自定义的样式

现在已经添加了自定义的菜单页面，接下来为这个页面添加内容。



相比于我们自己去设计样式，最简单的方式，是使用 WordPress 内置的样式，来展现我们的样式。这样体验也和 WordPress 官方的内容具有一致性，能够更好的表现出我们想要的内容。

使用 wrap 包裹我们的内容



在这个截图中，上面的内容使用的都是 `h1`, `dan s` 下方的内容明显要更适合 WordPress 的界面设计风格，这就是因为下面的内容被 `wrap` 类所包裹，展示的内容会按照 WordPress 官方的样式来书写。

```
<div class="wrap">
    <!-- Content -->
</div>
```

使用 WordPress 自带的信息提示的样式

我们在制作 WordPress 插件时，可能会用到一些提示，比如信息保存成功后的提示、信息出错的提示等等。这些 WordPress 官方提供了固定的样式，可以直接使用 WordPress 自己的样式来进行展示。

这方面我们可以直接复制这里的代码来显示：

```
<div id="message" class="updated">
    <p><strong>
        <!-- 保存信息 -->
    </strong></p>
</div>

<div id="message" class="error">
    <p><strong>
        <!-- 保存信息 -->
    </strong></p>
</div>
```

实现的效果如下：



这样的展示会更符合 WordPress 整体的体验。

使用 WordPress 的按钮样式



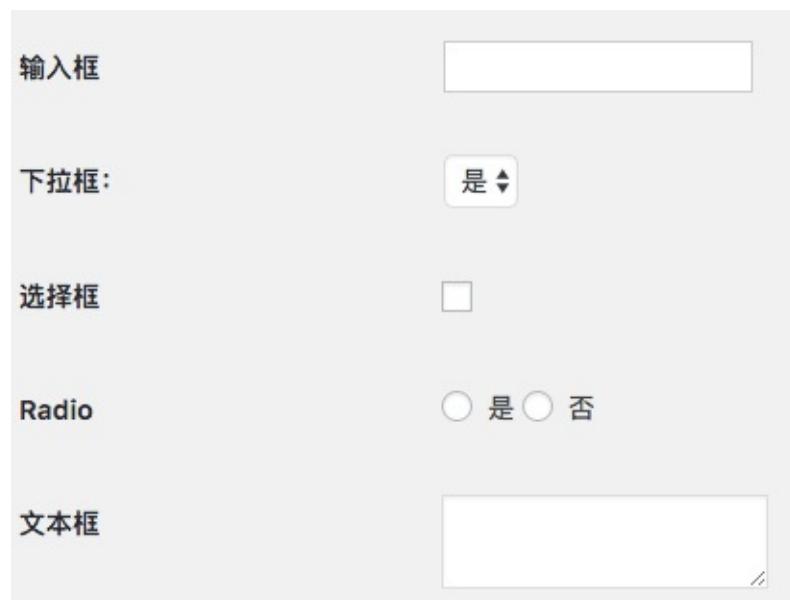
使用浏览器的普通按钮虽然也可以，但体验却差很多，设计也不一致，可以使用 WordPress 的按钮样式，来自定义页面的按钮。具体按钮的代码如下，可以根据自己的需要选择。

```
<input type="submit" name="test" value="普通按钮" />
<input type="submit" name="test" value="标准按钮" class="button" />
<input type="submit" name="test" value="主要按钮" class="button button-primary" />
<input type="submit" name="test" value="副按钮" class="button button-secondary" />
<input type="submit" name="test" value="大按钮" class="button button-large" />
<input type="submit" name="test" value="小按钮" class="button button-small" />
<input type="submit" name="test" value="超大按钮" class="button button-hero" />
```

a 链接也可以通过加入这些样式来美化。

使用 WordPress 自带的表单样式

WordPress 也提供了默认的表单样式，可以直接调用对应的样式来输出对应的样式。



```

<form method="POST" action="">
    <table class="form-table">
        <tr valign="top">
            <th><label for="input-example">输入框</label></th>
            <td><input id="input-example" name="input-example" /></td>
        </tr>
        <tr valign="top">
            <th><label for="select-example">下拉框: </label></th>
            <td>
                <select name="select-example">
                    <option value="1">是</option>
                    <option value="0">否</option>
                </select>
            </td>
        </tr>
        <tr valign="top">
            <th><label for="checkbox-example">选择框</label></th>
            <td><input type="checkbox" name="checkbox-example" /></td>
        </tr>
        <tr valign="top">
            <th><label for="radio-example">Radio </label></th>
            <td>
                <input type="radio" name="radio-example" value="是" /> 是
                <input type="radio" name="radio-example" value="否" /> 否
            </td>
        </tr>
        <tr valign="top">
            <th><label for="textarea">文本框</label></th>
            <td><textarea name="textarea"></textarea></td>
        </tr>
        <tr valign="top">
            <td>
                <input type="submit" name="save" value="保存" class="button-primary" />
                <input type="submit" name="reset" value="重置" class="button-secondary" />
            </td>
        </tr>
    </table>
</form>

```

使用 WordPress 自带的表格样式

序号	达人课名称
1	Angular 初学者快速上手教程
2	快速学习 Spring Boot 技术栈
3	Webpack 达人的成长之路

通过在表格上加入 `widefat striped` 两个类，就可以将我们的表格设置为 WordPress 的样式。

```

<table class="widefat striped">
    <thead>
        <tr>
            <th>序号</th>
            <th>达人课名称</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>1</td>
            <td>Angular 初学者快速上手教程</td>
        </tr>
        <tr>

```

```

        <td>2</td>
        <td>快速学习 Spring Boot 技术栈</td>
    </tr>
    <tr>
        <td>3</td>
        <td>Webpack 达人的成长之路</td>
    </tr>
</tbody>
</table>

```

其他的一些样式，如果我在这里没有提到，你需要用，可以自己通过 Chrome 的开发者工具审查获得，或者到读者圈内提问。

选项页面设置

接下来完善我们的插件管理页面的内容。

修改 `gitchat_custom_page` 函数内容如下：

```

function gitchat_custom_page(){
?>
<div class="wrap">
<?php
if ($_POST['code'] != null && check_admin_referer( 'gitchat_copyright' )){
    update_option( 'gitchat_copyright_code', $_POST['code'] );
    $code = $_POST['code'];
?>
<div id="message" class="updated"><p><strong>信息更新成功! </strong></p>
</div>
<?php
} else{
    $code = get_option('gitchat_copyright_code');
}
?>
<h1>GitChat 版权插件设置</h1>
<form method="POST" action="">
    <table class="form-table">

        <tr valign="top">
            <th><label for="textareaid">版权代码</label></th>
            <td><textarea name="code" col="30" row="10"><?php echo $code;?></textarea></td>
        </tr>
        <tr valign="top">
            <td>
                <input type="submit" name="save" value="保存" class="button-primary" />
                <input type="reset" name="reset" value="重置" class="button-secondary" />
            </td>
        </tr>
    </table>
    <?php
        wp_nonce_field('gitchat_copyright');
    ?>
</form>
</div>
<?php
}

```

这段代码中大部分都是我们在上面提到的表单内容。不再多讲，我来说一说里面的 PHP 代码。

```

<?php
if ($_POST && $_POST['code'] != null && check_admin_referer( 'gitchat_copyright' )){
    update_option( 'gitchat_copyright_code', $_POST['code'] );
    $code = $_POST['code'];
}

```

```
?>
<div id="message" class="updated"><p><strong>信息更新成功! </strong></p>
</div>
<?php
} else{
    $code = get_option('gitchat_copyright_code');
}
?>
```

这段代码主要是两个功能，首先判断是否是 Post 请求，判断 Post 请求中的 code 是否为空，以及这个插件的请求是否合法。如果 code 不为空，则使用 Post 的数据更新设置项，并提示更新成功；如果发来的请求不是 Post，则执行下方的获取代码，并赋值给 code。

在 table 底部我加了一段代码：

```
<?php wp_nonce_field('gitchat_copyright'); ?>
```

这段代码可以帮助我们验证请求。

关于 `check_admin_referer` 和 `wp_nonce_field`

这两个函数是我们的第一次看见，这两个函数是为表单加入验证数据避免被恶意利用。`wp_nonce_field` 函数会在我们的表单中加入两段代码，用于后续的验证。

```
</table>
<input type="hidden" id="_wpnonce" name="_wpnonce" value="407b7e7dbf">
<input type="hidden" name="_wp_http_referer" value="/wp-admin/admin.php?page=gitchat_optionpage">
</form>
```

而 `check_admin_referer` 会验证这两段代码是否匹配，如果不匹配，则会拒绝请求。

总结

这节课我们学习了如何构建插件的设置页面，后续会根据插件的复杂程度，你可以更加细化插件。

本节课理解起来可能不太明白，欢迎到读者圈提问。

[单击这里下载本课插件。](#)

开发一个 WordPress Widget

插件开发目标

这节课来创建一个简单的 WordPress Widget，实现在侧边栏显示达人课的链接。

重新认识 Widget

很多 WordPress 主题都支持自定义的侧边栏小工具，有了自定义的侧边栏小工具，WordPress 的侧边栏也可以变得多姿多彩。在开始开发 Widget 之前，先来重新认识一下 Widget。

一个 Widget 可以分为两个部分：

- 上方的标题部分
- 下方的内容部分

在管理后台，它是这个样子的：



而在前台，它是这个样子的：



每一个 Widget 都至少包含这两部分，其中第二部分又可能会有很多不同的样式。大多数就是对第二部分进行自定义，来达到展示特定内容的目的。

插件开发过程

这节课就来尝试创建一个 WordPress widget 小工具，这个小工具可以通过后台的设置，实现在前台输出达人课的基本信息。

创建插件

首先要创建一个插件，这里创建的插件名称为 `gitchat_widget`，并进行初始化。



初始化小工具

初始化完成插件后，来初始化我们的小工具。

在插件中加入如下代码：

```
class gitchat_widget extends WP_Widget {

    public function __construct() {
        $widget_ops = array(
            'classname' => 'gitchat_widget',
            'description' => '这是一个 GitChat 小工具',
        );
        parent::__construct( 'gitchat_widget', 'GitChat Widget', $widget_ops );
    }

    add_action('widgets_init',
        create_function('', 'return register_widget("gitchat_widget");')
    );
}
```

上述代码就是 WordPress 初始化一个小工具的代码。

这里创建了一个继承了 `WP_Widget` 类的类 `gitchat_widget`，该类的构造函数内构造了一组参数，参数指定了控件的名称、描述、id 等。

此外，通过 `add_action` 实现对 `widgets` 进行初始化的挂载，这里用到了一个函数 `create_function`，该函数可以实现创建一个匿名函数，这个函数没有参数，代码是注册一个 id 为 `gitchat_widget` 的 Widget。

实现前台展示功能

接下来实现前台的展示功能。

首先在 `__construct` 函数后加入一个新的函数 `widget`，这个函数接受两个参数，分别是 `args` 和 `instance`。

此时代码如下：

```

1  <?php
2  /*
3  Plugin Name: GitChat Widget
4  Plugin URI: https://gitchat.cn/plugins
5  Description: 这是一个 Widget 插件
6  Version: 1.0
7  Author: Gitchat
8  Author URI: https://gitchat.cn/
9  */
10
11 class gitchat_widget extends WP_Widget {
12
13     public function __construct() {
14         $widget_ops = array(
15             'classname' => 'gitchat_widget',
16             'description' => '这是一个 GitChat 小工具',
17         );
18         parent::__construct( 'gitchat_widget', 'Gitchat Widget', $widget_ops );
19     }
20
21     public function widget( $args, $instance ) {
22     }
23 }
24 add_action('widgets_init',
25     create_function('', 'return register_widget("gitchat_widget");')
26 );

```

我们将两个参数打印出来看看，具体有什么作用。

在 `widget` 函数中加入代码 `var_export($args);`。

进入 WordPress 后台的小工具中，拖动我们创建的 GitChat Widget 到侧边栏中。



来到前台，我们可以看到这样的输出：

```
array ( 'name' => '侧边栏', 'id' =>
'sidebar-1', 'description' => '将挂件加
入此处来在侧边栏中显示。', 'class' =>
'', 'before_widget' =>
_____
', 'after_widget' =>
'
', 'before_title' =>
'' , 'after_title' =>
', 'widget_id' => 'gitchat_widget-3',
'widget_name' => 'Gitchat Widget', )
```

现在我们知道了这里都有哪些内容了，可以调整一下输出。将 *widget* 函数代码改为这样：

```
public function widget( $args, $instance ) {
    echo $args['before_widget']; // 输出 widget 前面的内容，一般是主题提供的样式。
    // 输出内容
    echo $args['after_widget'];
}
```

添加小工具设置

现在小工具不能设置任何东西，所以接下来要让小工具更加完善，可以添加图片地址和对应的链接，这样就可以在前台显示我们的达人课和跳转链接了。

在 *widget* 函数后新创建一个函数 *form*，该函数接受一个 *instance* 参数，其将会在小工具中为我们渲染一个表单，用于填写具体的信息。同时，还要创建另外一个函数 *update*，该函数接受两个参数，分别是 *new_instance* 和 *old_instance*。

```
public function form( $instance ) {
}
public function update( $new_instance, $old_instance ) {
}
```

接下来一一实现这两个方法。

首先要实现的是 *form* 表单，在这个方法中要输出表单，同时，还需要确保之前填写的内容不会丢失，这样方便修改，而无需每次都重新完全填写所有内容，所以将 *form* 方法改写成下面这样。

```
public function form( $instance ) {
    $title = ! empty( $instance['title'] ) ? $instance['title'] : "默认标题";
```

```

$image_url = !empty($instance['image_url']) ? $instance['image_url'] : "http://images.gitbook.cn/FjhhSzg105uzXTpqZb3N3JIAKokE";
$link = !empty($instance['link']) ? $instance['link'] : "http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad";
?>
<p>
<label for=<?php echo esc_attr( $this->get_field_id( 'title' ) ); ?>">标题</label>
<input class="widefat" id=<?php echo esc_attr( $this->get_field_id( 'title' ) ); ?>" name=<?php echo esc_attr( $this->get_field_name( 'title' ) ); ?>" type="text" value=<?php echo esc_attr( $title ); ?>">
</p>
<p>
<label for=<?php echo esc_attr( $this->get_field_id( 'image_url' ) ); ?>">图片地址</label>
<input class="widefat" id=<?php echo esc_attr( $this->get_field_id( 'image_url' ) ); ?>" name=<?php echo esc_attr( $this->get_field_name( 'image_url' ) ); ?>" type="text" value=<?php echo esc_attr( $image_url ); ?>">
?>">
</p>
<p>
<label for=<?php echo esc_attr( $this->get_field_id( 'link' ) ); ?>">链接</label>
<input class="widefat" id=<?php echo esc_attr( $this->get_field_id( 'link' ) ); ?>" name=<?php echo esc_attr( $this->get_field_name( 'link' ) ); ?>" type="text" value=<?php echo esc_attr( $link ); ?>">
</p>
<?php
}

```

其中，代码的前三行是设置从数据库中获取到原先的数据，如果数据不存在，则将其设置为默认的数据。当然也可以额将后面的默认数据置空。

```

$title = !empty($instance['title']) ? $instance['title'] : "默认标题";
$image_url = !empty($instance['image_url']) ? $instance['image_url'] : "http://images.gitbook.cn/FjhhSzg105uzXTpqZb3N3JIAKokE";
$link = !empty($instance['link']) ? $instance['link'] : "http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad";

```

下面的数据则是输出一个个表单，这里我取其中一个来说明

```

<p>
<label for=<?php echo esc_attr( $this->get_field_id( 'title' ) ); ?>">标题</label>
<input class="widefat" id=<?php echo esc_attr( $this->get_field_id( 'title' ) ); ?>" name=<?php echo esc_attr( $this->get_field_name( 'title' ) ); ?>" type="text" value=<?php echo esc_attr( $title ); ?>">
</p>

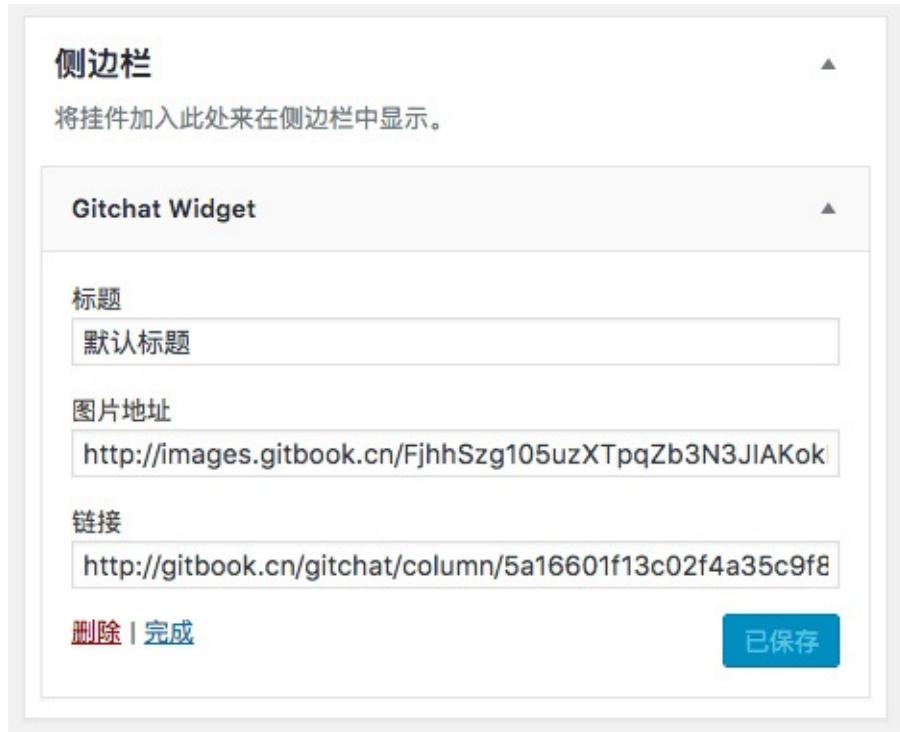
```

首先，输出一个 *label*，来帮助我们分辩到底设置的是哪个选项。然后，定义一个输入框，设置它的名字为 *field_name*，然后再设置其内容为我们刚刚获取到的数据库中的值。

esc_attr 是安全输出函数，对输出内容进行转义，确保恶意代码不会被执行。

保存文件，回到小工具页面，刷新。

再次拖动一个 GitChat Widget 到侧边栏，这时会发现我们的小工具中有了三个设置项。



将三个设置项进行修改，设置为你自己的值。

完善前台展示

现在后台的设置已经完善了，接下来完善前台的展示，刚刚只是展示了框架，并没有展示具体的内容。

刚刚设置的内容，会通过 `widget` 函数的 `instance` 参数传递进入函数。

```
public function widget( $args, $instance ) {
    echo $args['before_widget'];
    echo $args['after_widget'];
}
```

参数

我们可以通过 `$instance['属性名']` 来调用它。这里使用 html 的 `figure` 标签来实现结果的输出。

在上面的代码中加入如下代码：

```
echo '<figure>
    <a href="'. $instance['link'] . '">
        </a>
        <figcaption>'. $instance['title'] . '</figcaption>
    </figure>';
```

效果如下：

```
public function widget( $args, $instance ) {  
    echo $args['before_widget'];  
  
    echo '<figure>  
        <a href="'. $instance['link'] .'">  
              
            <figcaption>'. $instance['title'] . ' </figcaption>  
    </figure>';  
  
    echo $args['after_widget'];  
}
```

这样就简单的实现了前台内容的输出。

回到前台，刷新，可以看到：



点击图片，则可以跳转到 GitChat 的达人课界面。

再次验证功能

回到后台，将我们的小工具的设置改为其他达人课的信息，可以看到内容已经发生了改变。



Angular 初学者快速上手教程

这样，就验证了我们的小工具开发成功。

开发中需要注意的内容

在开发过程中需要注意，并不是每个小工具都有输入和输出，在涉及到用户的输入和输出时，一定要注意在其中加入过滤函数，常用的过滤函数你可以在[这里](#)找到。通过对用户输入的过滤，能够避免由用户的错误输入导致的恶意代码的嵌入。

记住，不要相信用户的输入！

[单击这里下载本课插件。](#)

开发一个短代码插件

课程目标

这节课来开发一个短代码插件，实现自定义短代码，并在此基础上，对 WordPress 自带的编辑器进行增强。

什么是短代码

短代码就是形如 `[git]` 这样的代码，WordPress 自2.5版本开始引入短代码机制，通过短代码，一个完全不懂技术的编辑人员也可以发布样式丰富的短代码。

一个短代码应该是由 `[]` 所包围，同时可以接受一定的参数。当我们使用短代码时，WordPress 在进行内容加载时，就会将短代码通过代码转换为适当的动态内容。虽然短代码可能很短，但是其转换出来的内容可能会非常长。

短代码的使用有很多多种形式，比如：

```
[git]
[git path="213"]
[git path="123321"]Code[/git]
```

这三种都是短代码的调用形式，只是有没有加入参数，和是否需要包含内容。

在插件开发过程中，有些时候需要让用户控制一些内容的显示，这个时候就要用到短代码了，通过短代码，可能用户只需要一行代码，就可以执行你的插件中极为复杂的操作。

开发短代码

在最开始，依然是创建插件，这里创建一个新的插件，插件名为 `gitchat_shortcode`。

ShortCode 这是一个 ShortCode 插件，增加了 `[git]`、`[chat]` 短代码。
[启用](#) | [删除](#) 1.0 版本 | 由 [Gitchat](#) | 访问插件主页

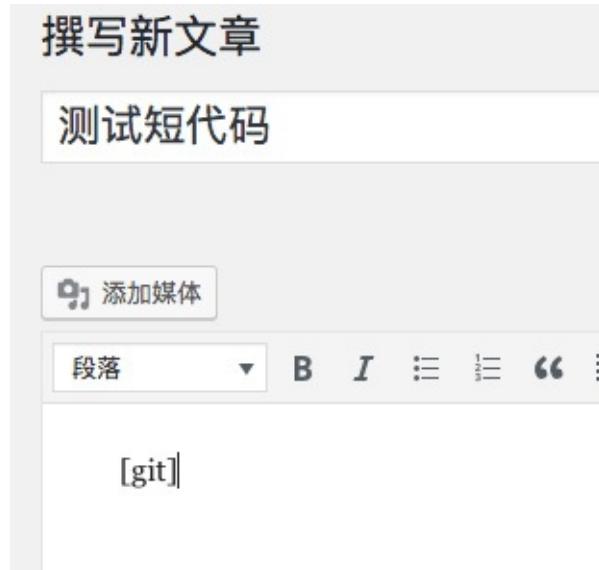
接下来开发这个插件的功能。

注册一个不需要参数的短代码

首先来注册一个不需要参数的短代码。在插件中加入如下代码，就可以注册一个标签为 `git` 的短代码了，这个短代码的功能非常简单，就是输出一个 `Hello World`：

```
function gitchat_git_shortcode() {
    return "Hello World";
}
add_shortcode( 'git', 'gitchat_git_shortcode' );
```

我们到插件后台，启用这个插件，然后进入文章管理中，创建一篇文章来测试短代码。



发布后，查看文章页面，就可以看到短代码输出了。

测试短代码

Hello World

这样就完成了一个最简单的短代码。对于一些纯粹展示型的内容，可以通过这样一个简单的方法实现内容的嵌入。

注册一个加入了参数的短代码

不过，大多数的时候，我们的短代码都需要加入一个或多个参数，来实现短代码的功能。这时就需要对短代码进行优化了。

修改一下短代码函数：

```
function gitchat_git_shortcode( $atts ) {
    $atts = shortcode_atts(
        array(
            'id' => '1',
            'title' => 'hahaha',
        ),
        $atts,
        'git'
    );
}
```

这样，短代码就成功的接收到参数，并进行处理，这里传入了三个参数，第一个参数是我们定义的短代码的参数，可以通过 `[git id="xxx" title="xxxx"]` 来实现调用。第二个参数则是这个短代码函数接受到的参数，是 WordPress 传递给我们的参数，里面包含了输入时的参数。第三个则是参数对应的短代码。

经过处理后，就可以以 `$atts['id']` 的形式在函数内调用具体的参数和值。

比如，我们让短代码输出接收到的参数，将函数改为下面这样：

```
function gitchat_git_shortcode( $atts ) {
    $atts = shortcode_atts(
        array(
            'id' => '1',
            'title' => 'hahaha',
        ),
        $atts,
        'git'
    );
    return $atts['id']."_".$atts['title'];
}
```

回到前台刷新一下页面，我们可以看到，这里我们的短代码将默认设置的值进行了输出。

测试短代码

1_hahaha

2017年12月28日 · admin · 编辑

单击“编辑”按钮，进入后台，修改短代码调用：

测试短代码

固定链接： <https://wordpress.php/?p=9> [更改固定链接](#)

 添加媒体

b *i* [link](#) [b-quote](#) [del](#) [ins](#) [img](#) [ul](#) [ol](#) [li](#) [c](#)

[git id="123" title="你好"]

再次回到前台，查看输出的内容，可以看到，我们设置的内容已经被输出出来了，这就说明内容可以被参数所接受、处理，最后输出了出来：

测试短代码

!123_你好

注册一个接受标签包含内容的短代码

有些时候，我们不止需要参数，毕竟可能会包含非常多的内容，比如代码高亮类的插件。

此时，希望我们的短代码能够获取到两个匹配标签之间的内容。

再修改一下短代码函数：

```
function gitchat_git_shortcode( $atts , $content = null ) {
    $atts = shortcode_atts(
        array(
            'id' => '1',
            'title' => 'hahaha',
        ),
        $atts,
        'git'
    );
}
```

这次，短代码函数接受两个参数，分别是 `atts` 和 `content`，其中 `content` 就是两个标签之间包含的内容。

这里设置了 `content` 的值为 `null`，如果你需要设置默认值，将其改为默认值即可。

将两个参数和内容都输出出来，将代码改为下面这样：

```
function gitchat_git_shortcode( $atts , $content = null ) {
    $atts = shortcode_atts(
        array(
            'id' => '1',
            'title' => 'hahaha',
        ),
        $atts,
        'git'
    );
    return $atts['id']."__".$atts['title']."---".$content;
}
```

现在刷新一下文章页，可以看到，由于短代码没有匹配包含，所以并没有输出 `content`。

测试短代码

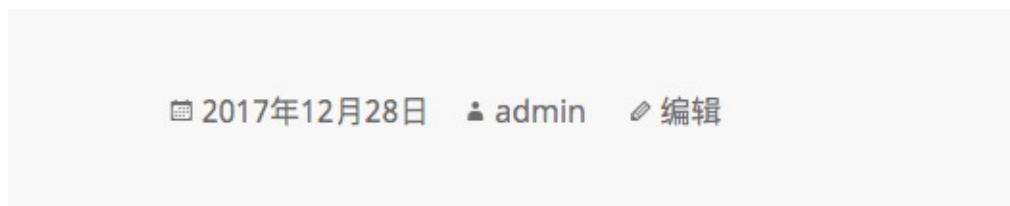
!123_你好--



单击“编辑”按钮，将内容修改为 [git id="!123" title="你好"]这是我的标题[/git]，保存并回到文章页，刷新。

测试短代码

!123_你好—这是我的标题



可以看到，这次输入内容正常的刷新出来了。

至此，完成了三种不同类型的短代码开发，足以应付你的绝大多数场景了。

编辑器增强

我们现在开发好了短代码，但是如果设置了多个短代码或短代码有多个参数，用户在使用时就会很麻烦，因为还要去查询短代码的说明。这里，就来实现编辑器的增强，在编辑器上加入短代码的按钮。

QuickTag

WordPress 的编辑器支持两种模式，分别是 TinyMCE 和 QuickTag 模式（HTML）模式，点击编辑器右上角的文本，就可以添加一个按钮到 QuickTag 中了。



接下来实现在编辑器中加入一个 QuickTag。

在插件中加入如下代码：

```
function gitchat_git_qt() {
}
add_action( 'admin_print_footer_scripts', 'gitchat_git_qt' );
```

这样就在 WordPress 后台加载底部脚本时，加入了我们自己的函数。接下来来写函数体的内容。

将函数改为下面这样：

```
function gitchat_git_qt() {
    if ( wp_script_is( 'quicktags' ) ) { // 判断是否正在加载 quicktags
    ?>
    <script type="text/javascript">
    // 调用 quick tag 的 api 添加按钮
    QTags.addButton( 'git', 'Git', '[git id="13" title="code"]', '[/code]', '', 'QucikTag For GitChat', 1 );
    </script>
    <?php
    }
}
```

我来解释一下上面这段代码。

首先，使用 `if` 语句来判断当前是否正在加载 `quicktag` 相关的脚本，因为如果你加载的过早，会导致 `QTags` 实例调用失败，也就没有办法来加载按钮了。

然后嵌入一段 JavaScript 代码。这个代码很简单，使用 `QTags` 的 `addButton` 方法，来加载一个新按钮，这个函数接受 8 个参数，不过定义时一般来说，可以不传入那么多。

```
QTags.addButton( id, display, arg1, arg2, access_key, title, priority, instance );
```

第一个参数是按钮的 ID，需要是唯一的；第二个参数则是这个按钮上显示的文字，比如设置为 `Git`，显示的时候是



，可以设置成为你自己的（比如使用 `dashicons`）；第三个参数则是第一次按下这个按钮时，会输出的内容；第四个参数是第二次按下这个按钮会输出的内容；第五个参数是 `access_key`，目前已经被弃用了，我们就不用管了；第六个参数是标题，当鼠标指针悬浮在这个按钮上时，将会展示标题文字，可以将其设置为短代码的说明或按钮的说明；第七个参数是优先级，这个参数会决定短代码的位置，比如将其设置为 1-9 时，这个按钮会放在第一位；设置为 11-19 时，会放在第二个位置，如果你希望放在最后，则需要设置一个非常大的值才行；最后一个参数是对按钮进行设置，可以给短代码加入 `ariaLabel` 和 `ariaLabelClose`，基本用不到，可以不填。

为什么第三个和第四个参数要这样拆？

这是因为，如果选中文字再按下按钮时，编辑器会自动将选中的内容放在参数三和参数四之间，类似这样：



如果拆分有问题，生成的代码可能就有问题了，将短代码的参数放在参数三是更为合适的。

保存插件，进入新建文章页面，将编辑器切换为 QuickTag 编辑器，就能看到添加的按钮了：



TinyMCE 增强

QuickTag 非常方便，但是大多数人更习惯使用 TinyMCE，所以我们还要针对 TinyMCE 编辑器进行增强。

Tiny 编辑器的增强就比较麻烦了。

首先要为这个按钮创建一个 JS 文件，在插件根目录下创建一个 `tinymce.js`，在其中加入如下代码：

```
// tinymce.js
(function() {
    tinymce.create('tinymce.plugins.gitchat', {
        init : function(ed, url) {
            ed.addButton('gitchat', {
                title : 'GitChat Button',
                image : url + '/icon.png',
                onclick : function() {
                    ed.selection.setContent('[git id="1" title="haha"]' + ed.selection.getContent() + '[/git]')
                }
            });
        },
        createControl : function(n, cm) {
            return null;
        },
    });
    tinymce.PluginManager.add('gitchat', tinymce.plugins.gitchat);
})();
```

接下来将我们的按钮注册到 TinyMCE 中；重新打开我们的插件主文件，在文件尾部加入如下代码：

```

function register_button( $buttons ) {
    array_push( $buttons, "|", "gitchat" ); // 将按钮加到列表尾部
    return $buttons;
}
function register_plugin( $plugin_array ) {
    $plugin_array['gitchat'] = plugin_dir_url( __FILE__ ) . '/tinymce.js'; // 新增我们自己开发的插件
    return $plugin_array;
}
function gitchat_tinymce_enhance() {
    // 判断用户是否有权限编辑文章/页面，无权则不加载
    if ( ! current_user_can('edit_posts') && ! current_user_can('edit_pages') ) {
        return;
    }
    // 如果未开启富文本编辑，则不渲染。
    if ( get_user_option('rich_editing') == 'true' ) {
        add_filter( 'mce_external_plugins', 'register_plugin' );
        add_filter( 'mce_buttons', 'register_button' );
    }
}
add_action('admin_init', 'gitchat_tinymce_enhance');

```

在 `tinymce.js` 中，我们调用 `tinymce.create()` 创建了一个新的插件，然后使用 `init` 函数定义了一个新增的按钮和其对应的 `onClick` 方法，最后使用 `PluginManager` 来将插件注入到系统中去。

需要准备一个 png 文件作为图标。

接下来在插件主文件中，先定义了 `register_button`，将我们的按钮放入编辑器的列表中。然后定义了 `register_plugin` 将插件页注入到 TinyMCE 中去，最后定义了 `gitchat_tinymce_enhance` 并将其挂载在 `admin_init` 上。在这个函数中，我们对具体的权限和是否进行富文本编辑进行了判断，根据判断的结果，决定是否加入我们的插件。

编辑完成后，保存文件，回到后台，进入编辑器，可以看到我们的按钮，已经添加进去了。



选择我们要处理的文字，点击按钮，就可以把相关内容包含在其中了。如果不选择任何东西，则直接将短代码输入到编辑器中。

至此，我们也完成了对 TinyMCE 编辑器的增强。

[请单击这里下载本课插件。](#)

提交你的插件到 WordPress 的官方仓库

开发完了插件，我们肯定希望插件能够给更多的人去使用，让更多的人成为我们的用户，一个很好的渠道便是上架到 WordPress 官方仓库。

上架到官方仓库后，插件会更容易被其他人所认可：「这个插件能够在官方上架，肯定还不错」。这节课，我就来教大家如何上架一个插件。

整理插件

即使开发好了一个插件，也并不意味着能够将它上传到 WordPress 当中去。WordPress 虽然不如 App Store 审核那般严格、复杂，但是仍然会对代码进行 Review，通过 Review 后，插件才能够上传到 WordPress 官方插件仓库。

除了代码，还需要有一些其他的东西需要遵守，这里一一道来。

1. 使用 GPL v2 作为你的插件开源协议

WordPress 是一个基于 GPL v2 的开源程序，作为其衍生程序的插件，你的程序的开源协议应当于 GPL v2 匹配。具体的列表可以参考 [这里](#)。

不过，如果为了省事，可以直接使用 GPL v2 作为插件的开源协议。

2. 编写 README.txt

因为你的插件要提交到 WordPress 的官方仓库，而官方仓库内的信息都是基于 README.txt 生成的：

Description

Akismet checks your comments and contact form submissions against our global database of spam to prevent your site from publishing malicious content. You can review the comment spam it catches on your blog's "Comments" admin screen.

Major features in Akismet include:

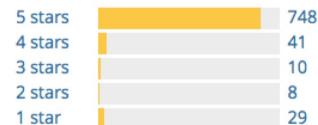
- Automatically checks all comments and filters out the ones that look like spam.
- Each comment has a status history, so you can easily see which comments were caught or cleared by Akismet and which were spammed or unspammed by a moderator.
- URLs are shown in the comment body to reveal hidden or misleading links.
- Moderators can see the number of approved comments for each user.
- A discard feature that outright blocks the worst spam, saving you disk space and speeding up your site.

PS: You'll need an [Akismet.com API key](#) to use it. Keys are free for personal blogs; paid subscriptions are available for businesses and commercial sites.

Version:	4.0.2
Last updated:	1 week ago
Active installations:	5+ million
Requires WordPress Version:	4.0
Tested up to:	4.9.1
Languages:	See all 57
Tags:	akismet anti-spam antispam comments spam

[Advanced View](#)

基于 `readme.txt` 识别出来的



FAQ

Installation Instructions

Contributors & Developers

"Akismet Anti-Spam" is open source software. The following people have contributed to this plugin.

所以需要写好 `readme.txt`。

而其 `readme.txt` 有其自己的格式，需要按照它的格式来书写。示例文件可以在[这里](#)下载。

```
== 插件名称 ==
Contributors: WordPress.org 的用户名
Tags: 插件的标签, 比如 spam
Donate link: 捐款链接地址
Requires at least: 最低 WordPress 版本
Tested up to: 最高测试的 WordPress 的版本
Requires PHP: 所需 php 版本
Stable tag: 稳定版标签, 如 1.0
License: 开源协议
License URI: 协议地址
```

这里是短描述，会展示在列表中

```
== Description ==
这里是长描述，会展示在独立的页面中
```

```
== Installation ==
具体的安装说明
```

```
== Frequently Asked Questions ==
1. 常见问题1
```

Support

Issues resolved in last two months:

7 out of 10

[View support forum](#)

回复内容

2. 常见问题2

回复内容2

`== Screenshots ==`

1. 截图1, 放在 asset 目录中
2. 截图2, 放在 asset 目录中
3. 截图3, 放在 asset 目录中

`== Changelog ==`

`==== 1.0 ===`

更新日志。最新的放在顶部, 旧的在底部

`==== 0.9 ===`

`== Upgrade Notice ==`

更新提醒, 说明为什么需要更新

你可以根据我这里的说明, 加入你自己的内容, 并将其保存在 `readme.txt` 文件中, 放置在插件根目录。

这里需要注意的是, 截图不是放在插件根目录的, 具体可以看后续的内容。

3. 插件开发的功能应该是合法的

这个不用我多说, 如果真的开发的是一个违法插件, 就通过其他渠道发行吧。

4. 未经用户允许, 不得在页面插入外部的链接

这个是说不能随意在页面插入外部的链接, 比如你自己的「Powered By」链接。如果你想插入, 需要征得用户的同意(在插件中做一个设置项目)。

5. 开发者要对自己提供的文件和服务负责

WordPress 官方只会在初次对你的代码进行 Review (在安全部分, 我曾提到过), 也就是说, 除了第一次上传的代码官方会看, 后续的都是不会看的 (毕竟没空)。

插件的管理是通过 SVN 进行的, 可以随意上传你需要的文件, 不过如果这些文件出现了问题, 可能会对你追责的呦~

6. 及时更新你的代码到官方仓库

将你的代码上传到官方仓库可以让用户更加方便的下载到需要的插件, 把最新的代码放在你的开发环境(比如 GitHub 中, 下载起来并不方便)。

7. 保持你的代码可读性

不要使用诸如 `p`、`a`、`b` 之类的单个字符来命名变量或者函数, 你可以尽可能的简化代码, 但是不能以牺牲可读性为代价。

8. 不允许跟踪用户的信息

未经用户允许, 不得追踪用户的信息。

9. 不允许劫持用户的管理面板

避免对用户的管理面板执行过多的干预。

10. 使用 WordPress 提供的库

如果需要在插件中引入一些库，需要使用 WordPress 提供的。具体的列表[详见这里](#)。

11. 不要在 SVN 仓库做日常开发更新

WordPress.org 提供的仓库是用来发布插件的，而不是用来进行日常开发的，日常开发需使用其他版本控制工具。

12. 发布新版本时，更新版本号

发布新的版本时，更新一下你的版本号。

更新了版本号后，WordPress 后台会自动提示有插件更新，很方便。

13. 只允许提交完整的插件

不允许提交一个空白的插件来占「目录名」，你上传的插件必须是可用的插件：



注册一个 WordPress.org 的账户

想要上传插件到 WordPress 官方，首先你要先在 WordPress.org 注册一个账户，这个账户不同于你在 wordpress.org 上注册的账户，两者并不互通，必须重新注册一个账户。

进入 WordPress 官网，点击顶部菜单中的 Plugins，进入到插件市场。在插件市场页面，点击页面右上角的 Register，就能够进入到注册的页面了。



WordPress.Org 使用的也是 WordPress 构建的，所以注册也和 WordPress 相同，输入用户名、邮箱等信息，点击「Create Account」，系统会自动给你的邮箱发送一封注册确认信。

The screenshot shows the WordPress.org account creation interface. At the top is the WordPress logo and the text "WORDPRESS.ORG". Below this is a large text area with the instruction: "Create a WordPress.org account to start contributing to WordPress, get help in the support forums, or rate and review themes and plugins." The main form area has two input fields: "Username" containing "gitchat" and "Email" containing "gitchat@example.com". Both fields have placeholder text below them indicating they are required and specifying allowed characters. There is also a checkbox for subscribing to announcements. A large blue button at the bottom says "Create Account". At the bottom of the page, there is a link "Already have an account? • [WordPress.org](#)".

Create a WordPress.org account to start contributing to WordPress, get help in the support forums, or rate and review themes and plugins.

Username

gitchat

Required. Only lower case letters (a-z) and numbers (0-9) are allowed.

Email

gitchat@example.com

Required. Your password will be emailed here.

Subscribe to WordPress Announcements mailing list (a few messages a year)

Create Account

Already have an account? • [WordPress.org](#)

确认后，回到 WordPress.org 的插件商城，点击右上角的「Login」登录。

提交你的插件

将插件打包成为一个 zip 文件，单击 [这里](#) 打开页面，登录后会看到这样的界面。

Add Your Plugin

Once submitted, your plugin will be manually reviewed for any common errors as well as ensuring it complies with [all the guidelines](#).

Currently there are **702** plugins in the review queue **11** of which are awaiting their initial review.

[Select File](#) [Upload](#)

Maximum allowed file size: 25MB

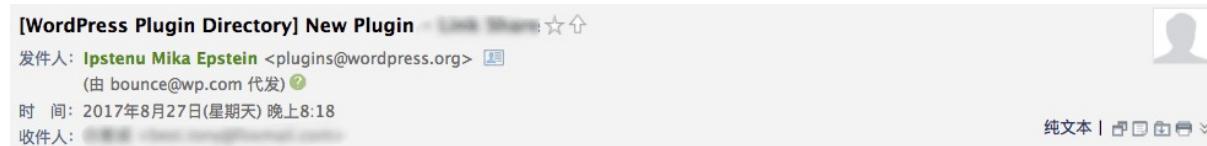
Even if you've submitted a dozen plugins, take the time to refresh your memory with the following information:

- [How to use SVN](#)
- [Plugin Assets \(and how to use them\)](#)
- [Developer FAQ](#)

点击「Select File」，选择刚刚打包的 zip 文件，单击 Upload 按钮，就可以将你的插件添加到审核队列了。

这里我圈出来了两个数字，一个是 review queue，另一个是 initial review queue，需要关注的是 initial review queue。initial review 的多少将会影响插件什么时候会被审核。review queue 的多少会影响插件后续审核的速度（因为大部分情况下，插件都会有这样或那样的问题，无法一次审核通过。）

提交成功后，会收到一封来自 WordPress 邮件，提醒你插件已经成功提交了：



Thank you for uploading [REDACTED] to the WordPress Plugin Directory. We will review your submission as soon as possible and send you a follow up email with the results.

Your plugin has been given the initial slug of [REDACTED] however this is subject to change based on the results of your review.

If there is a problem with this submission, such as an incorrect display name or slug, please reply to this email and let us know. In most cases, we can correct errors as long as the plugin has not yet been approved. Please do not submit your plugin multiple times in an attempt to correct the issue, just email us.

Remember to read the developer guidelines: <https://developer.wordpress.org/plugins/wordpress-org/detailed-plugin-guidelines/>

Please make sure to follow our official blog: <https://make.wordpress.org/plugins/>

--

The WordPress Plugin Directory Team
<https://make.wordpress.org/plugins>

修改插件

在大部分情况下第一次提交都会有一些问题，这时审核团队会给你发邮件，告诉你你的插件有什么问题，然后根据邮件内容去更新代码，并将新的代码以附件的形式回复给审核者。

There are issues with your plugin code.

Please read this ENTIRE email, address all listed issues, and reply to this email with your corrected code attached (or linked). It is required for you to read and reply to these emails, and failure to do so will result in significant delays with your plugin being accepted.

Also please remember in addition to code quality, security and functionality, we require all plugins adhere to our guidelines. If you have not yet, please read them:

* <https://developer.wordpress.org/plugins/wordpress-org/detailed-plugin-guidelines/>

Please sanitize, escape, and validate your POST calls

When you include POST/GET/REQUEST calls in your plugin, it's important to sanitize, validate, and escape them. The goal here is to prevent a user from accidentally sending trash data through the system, as well as protecting them from potential security issues.

再次发送邮件通过审核后，会收到另一封邮件：

Congratulations, your plugin hosting request for [link](#) has been approved.

Within one hour you will have access to your SVN repository with the WordPress.org username and password you used to log in and submit your request. Your username is case sensitive.

<https://plugins.svn.wordpress.org>

Here are some handy links to help you get started.

WordPress Plugin Directory Guidelines:

<https://developer.wordpress.org/plugins/wordpress-org/detailed-plugin-guidelines/>

Using Subversion with the WordPress Plugin Directory:

<https://developer.wordpress.org/plugins/wordpress-org/how-to-use-subversion/>

FAQ about the WordPress Plugin Directory:

<https://developer.wordpress.org/plugins/wordpress-org/plugin-developer-faq/>

WordPress Plugin Directory readme.txt standard:

<https://wordpress.org/plugins/developers/#readme>

A readme.txt validator:

<https://wordpress.org/plugins/developers/readme-validator/>

Plugin Assets (header images etc):

<https://developer.wordpress.org/plugins/wordpress-org/plugin-assets/>

If you have issues or questions, please reply to this email and let us know.

在这邮件中，会得到一个 SVN 仓库的地址，使用 SVN 管理软件将这个仓库克隆到本地。

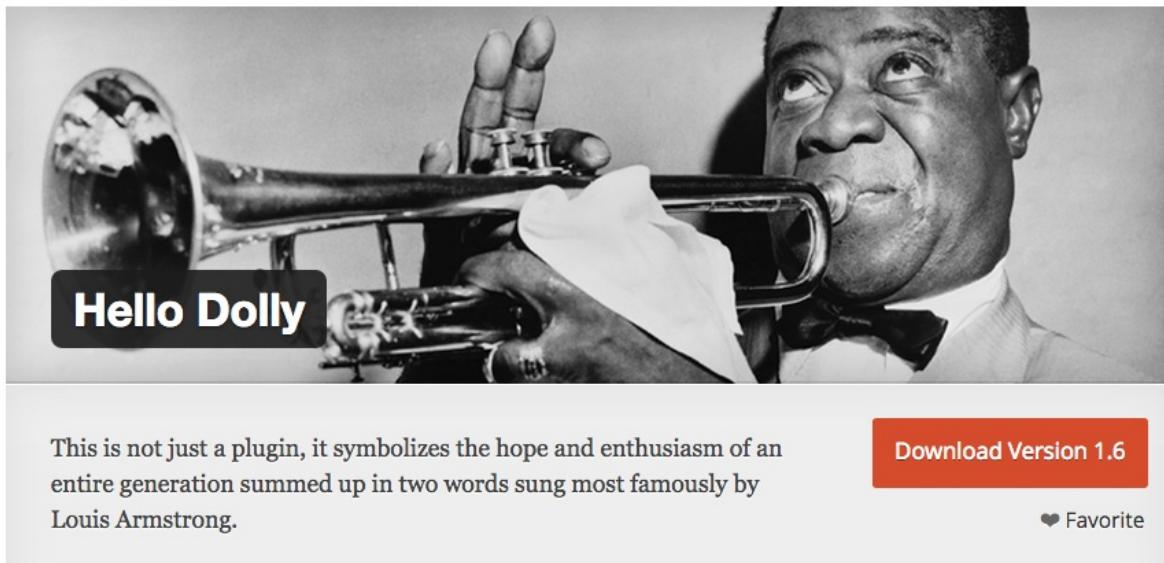
如果不会使用 SVN，建议查看[这篇教程](#)。

上传插件

将仓库克隆到本地后，会发现插件放在 trunk 目录下，其他目录而 assets、branches 、tags 目录下都没有文件。

需要准备一些图片，来作为插件的 Banner 等。

Banner



This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words sung most famously by Louis Armstrong.

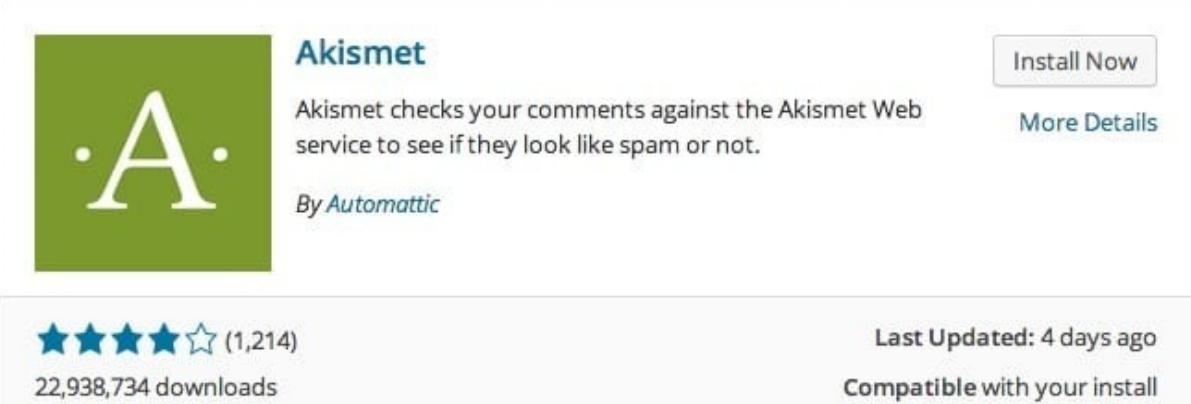
[Download Version 1.6](#)

[♥ Favorite](#)

你需要准备一张标准大小的 Banner (772x250)，格式为 png/jpg。一张高清大小的 Banner(1544x500)，格式为 png/jpg。

Banner 要放在 assets 目录下，并且Banner 命名为 `banner-772x250.png`、`banner-1544x500.png`。

icon



Akismet

Akismet checks your comments against the Akismet Web service to see if they look like spam or not.

By [Automattic](#)

★ ★ ★ ★ ☆ (1,214)

22,938,734 downloads

Last Updated: 4 days ago

Compatible with your install

图标也需要准备2个 png 的，或者一个 SVN 格式的也可以。

如果要使用 png 格式的，需要准备一个 128x128 的和一个 256x256 的。如果是 svg 格式的，只需准备一个。

icon 要放在 assets 目录下，并且icon 命名为 `icon-128x128.png`、`icon-256x256.png`，如果你使用 SVG 格式的，文件命名为 `icon.svg`。

screenshots

截图文件也同样要放在 assets 目录下，并将其命名为 `screenshot-1.png`、`screenshot-2.png`。

使用时，你要在 `readme.txt` 中调用。调用方式如下：

这是照片说明：一行一张图片，后面跟着说明。

文件存放示意图

文件存放的位置应该和我的示意图一致：

```
● test tree -L 2
.
├── assets
│   ├── banner-1544x500.png
│   ├── banner-772x250.png
│   ├── icon-128x128.png
│   ├── icon-256x256.png
│   ├── screenshot-1.png
│   ├── screenshot-2.png
│   └── screenshot-3.png
├── branches
└── tags
    ├── 0.0.4
    ├── 0.0.5
    └── 0.0.6
└── trunk
    ├── languages
    └── readme.txt
```

使用 SVN 上传

文件放置正确后，执行如下命令，即可将插件更新到官方仓库了。

```
svn add .
svn commit -m 'upload assets'
```

上传成功后，就可以在插件页面查看到插件下具体信息了。

自此就完成了插件的上传了，后续就可以把插件分析给别人了，或者让别人在 WordPress 后台搜索进行安装。

为你的主题、插件实现国际化

这节课来学习如何为你的主题和插件实现多语言。

为什么要为插件/主题实现国际化

我们的主题、插件可以通过加入国际化，实现全球范围的用户支持，即使是海外的用户，也可以直接使用你的主题/插件。所以我建议你在制作主题/插件时，就按照多语言的规范来制作。

核心代码

```
function i10n(){
    $current_locale = get_locale();
    if(!empty($current_locale)){
        $mo_file = dirname(__FILE__).'/languages/'.$current_locale.".mo";
        if (@file_exists($mo_file)&& is_readable($mo_file))
            load_textdomain('your-plugins',$mo_file);
    }
}
add_action('init','i10n');
```

在插件主文件/主题的 functions.php 中加入上述代码，就可以为主题/插件接入多语言功能。

这段代码会自动到插件的根目录下的 `/languages` 目录下查找对应语言的语言包，比如 `zh_CN.mo`。

存在语言包文件的话，则会自动加载语言包，对界面进行汉化。

调整主题的输出

上面实现了语言包的引入，但是我们现在还没有语言包，首先要制作语言包，这里为大家介绍两个函数：

```
// 返回值，不输出
__( 'New', 'your-plugins' )
// 直接输出
_e( 'New', 'your-plugins' )
```

这两个函数可以用于在主题、插件中的输出，当使用 `__("xx", "xxy")` 时，主题、插件会自动到 `xxy` 的语言包中查找 `xx` 字段，从而实现多语言输出的支持。

```
<div class="blog-masthead">
    <div class="container">
        <nav class="blog-nav">
            <a class="blog-nav-item active" href="/"><?php _e( 'Home', 'gitchat' ); ?></a>
            <?php wp_list_pages( '&title_li=' ); ?>
        </nav>
    </div>
</div>
```

因此，要将我们的主题/插件中所有涉及到文字输出的内容，改为使用这个函数进行输出！

这也是我为什么要推荐你在开发主题时就按照规范来，如果一开始没有养成良好的习惯，最终修改时，就会面临巨量的工作，可能一下子就打消了做国际化的念头。

这一步一定要做，一方面，我们需要使用这两个函数实现根据不同语言输出不同的内容，另一方面，还需要借助这两个函数来快速生成语言包。

此外，还需要修改我们的 `style.css` 文件，在其中加入语言设置项：

```
/*
Theme Name: GitChat
Theme URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Author: 白宦成
Author URI: http://gitbook.cn/gitchat/column/5a16601f13c02f4a35c9f8ad
Description: GitChat WordPress 演示达人课
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Tags: blue
Text Domain: gitchat
Domain Path: /languages

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

前者指明我们的翻译文件的文本域（你可以理解为唯一的标识）和语言文件夹所在目录，这里我们使用 `languages`，修改完成后，保存这个文件，并创建对应的语言文件夹。

如何快速生成语言包

这里需要用到一个软件——Poedit，我们会使用这个软件来生成语言包，并生成不同语言的翻译文件。

可以在[他们的官网下载](#)。

Poedit 提供了 Windows/macOS 的版本，还提供了源代码，即使使用的是 Linux，也可以使用。

安装完成后，打开 Poedit，点击其中的「翻译 WordPress 的主题或插件」：

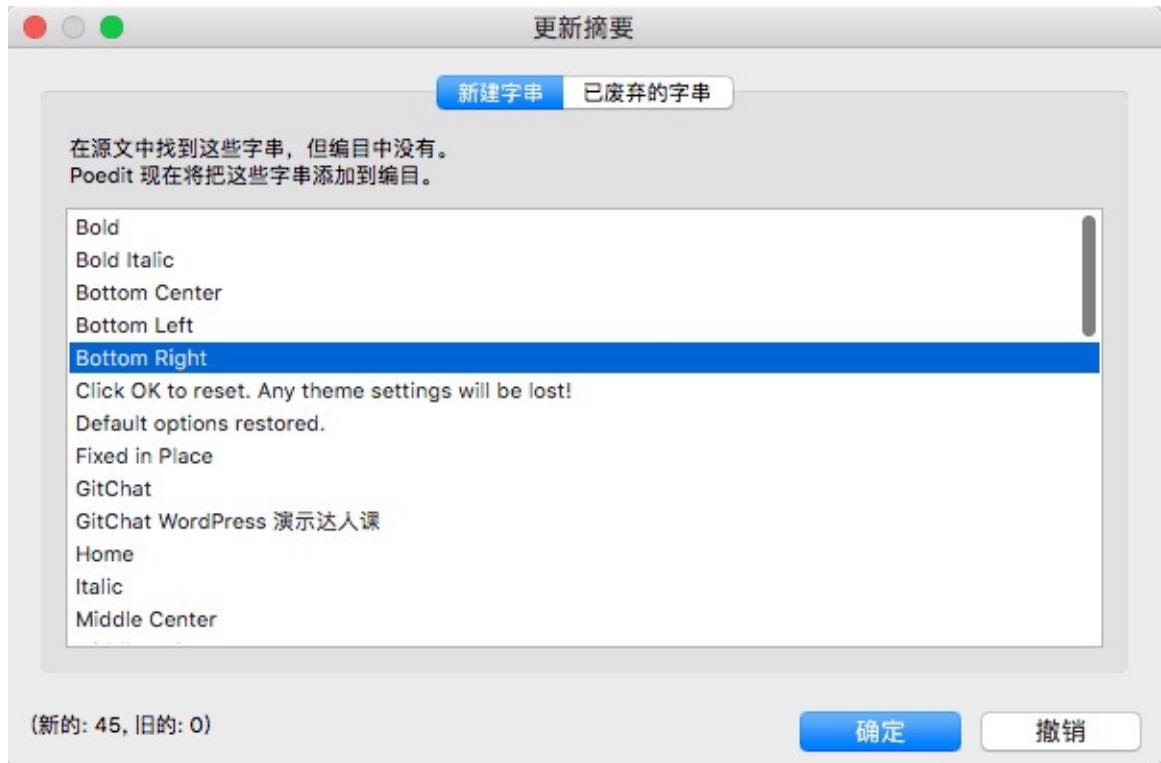


然后在弹出的窗口中选择你的主题文件夹，或者拖动文件夹到这里来：

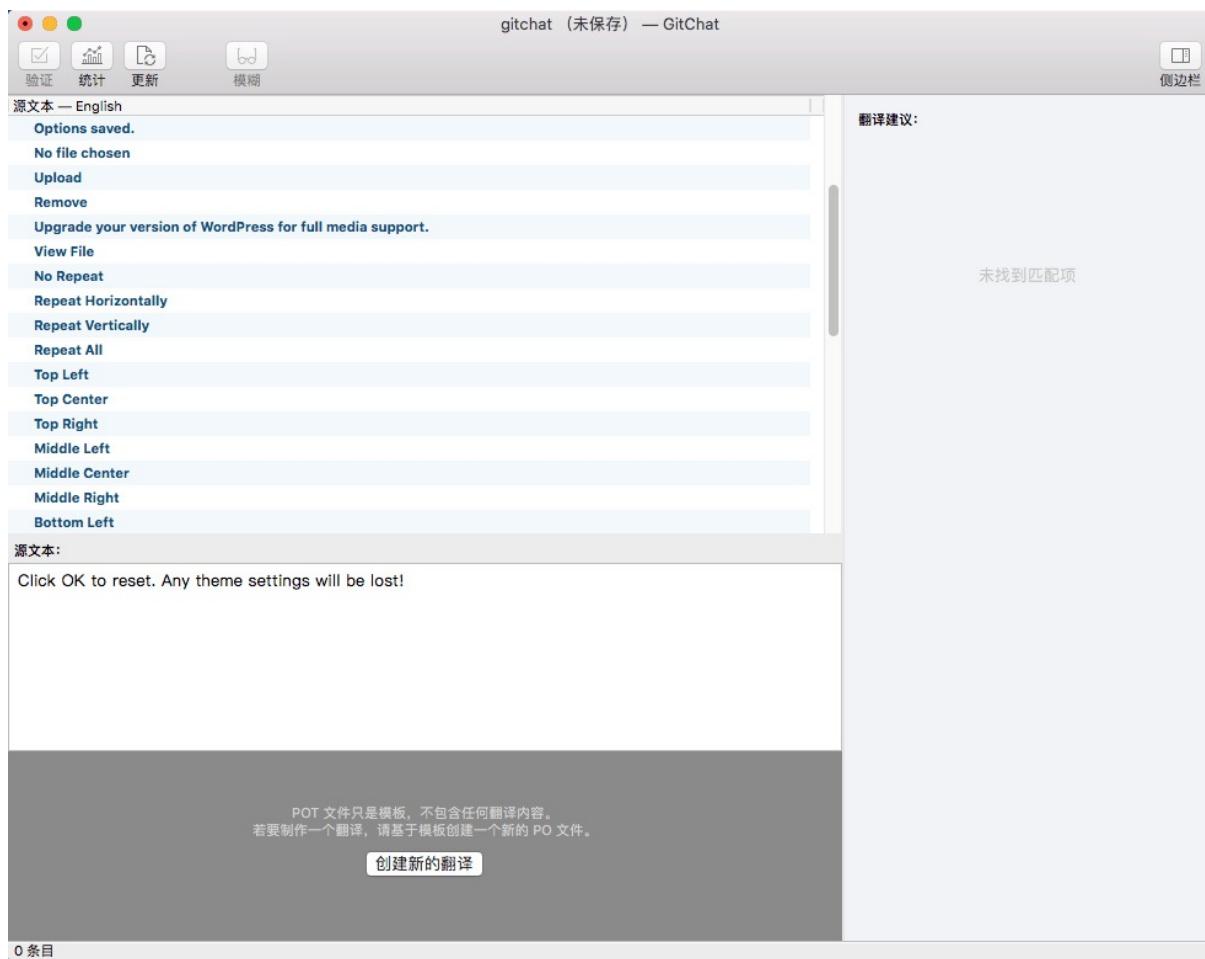


这里选择「创建 Pot」，这样可以创建一个用于翻译的模板，后续再根据这个模板创建我们自己语言的翻译。

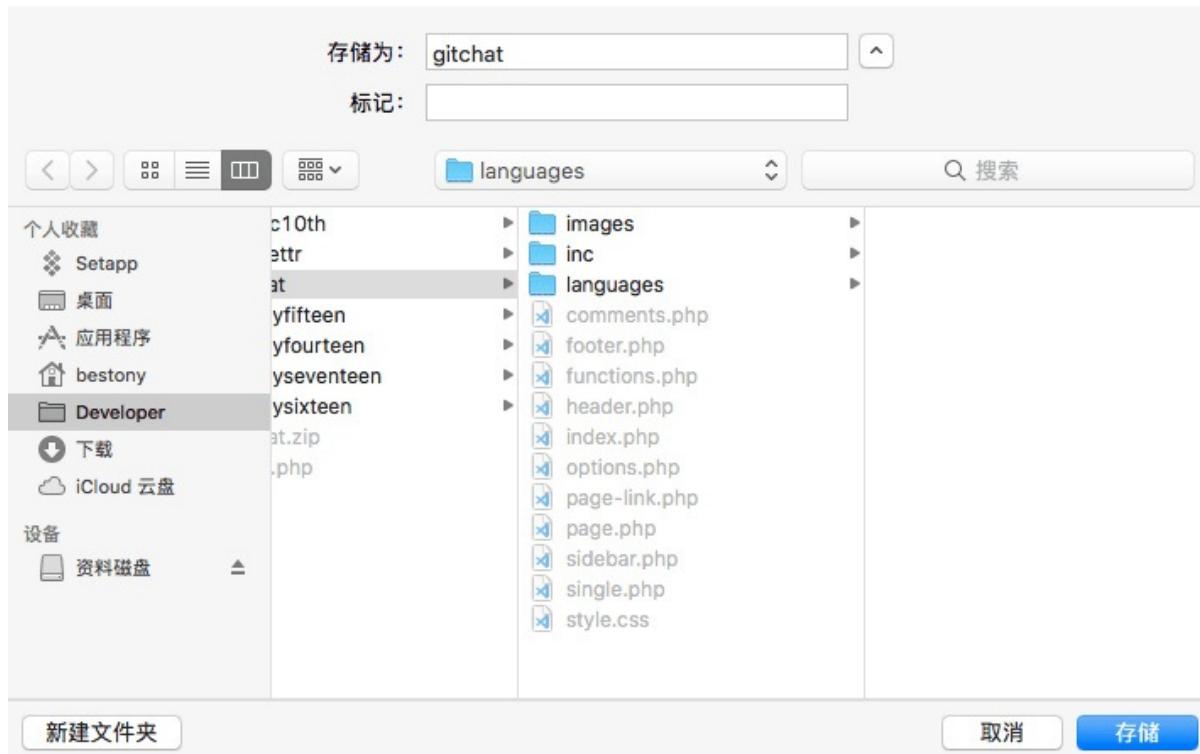
Poedit 会读取我们的主题文件，并生成要翻译的字符串列表：



单击“确定”按钮，就会看到我们所有的主题：

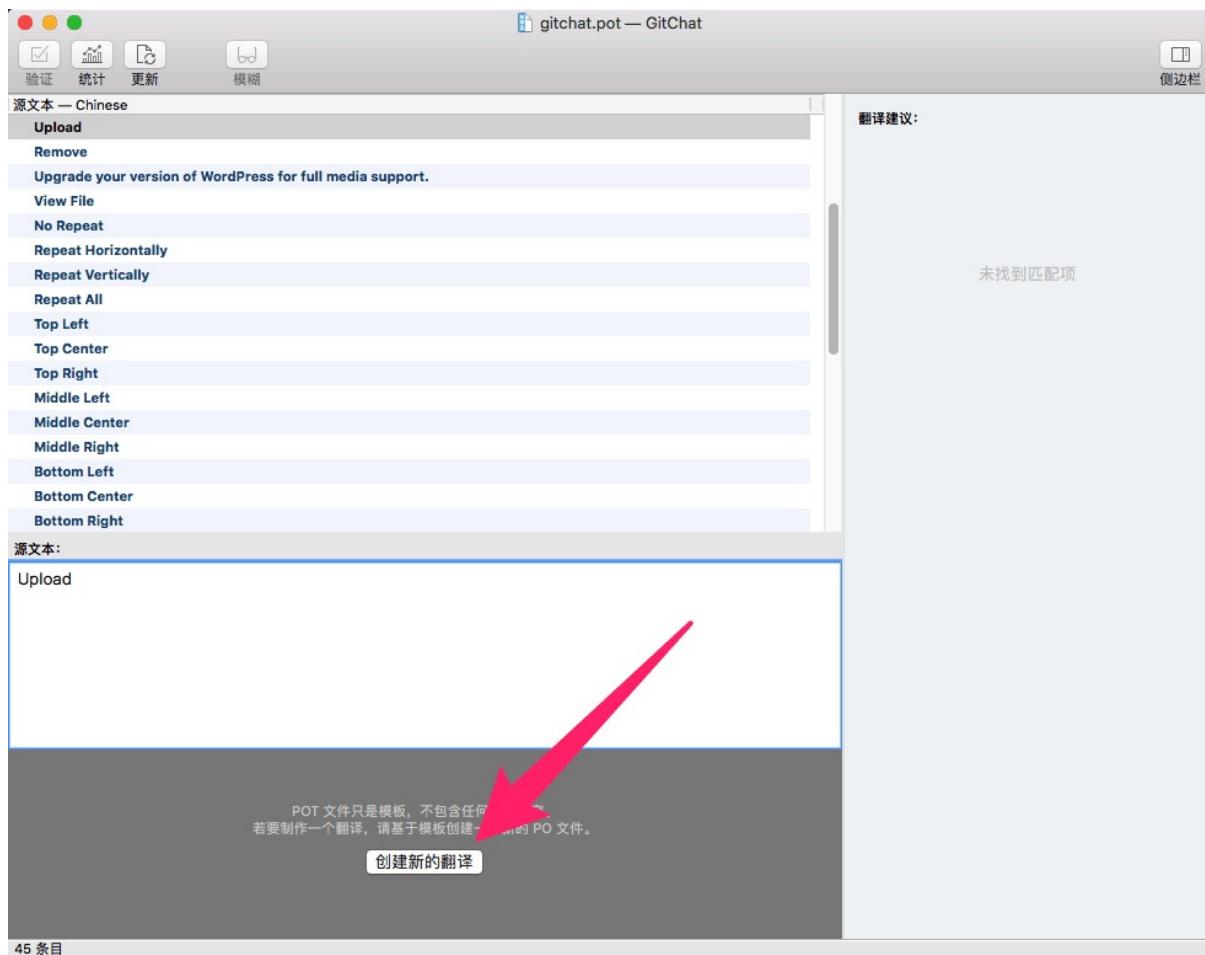


确认无误后，单击文件“保存”按钮，将这个 Pot 文件保存下来，后续语言翻译的志愿者就可以借助这个模板文件，来帮助你更新翻译了。



翻译主题

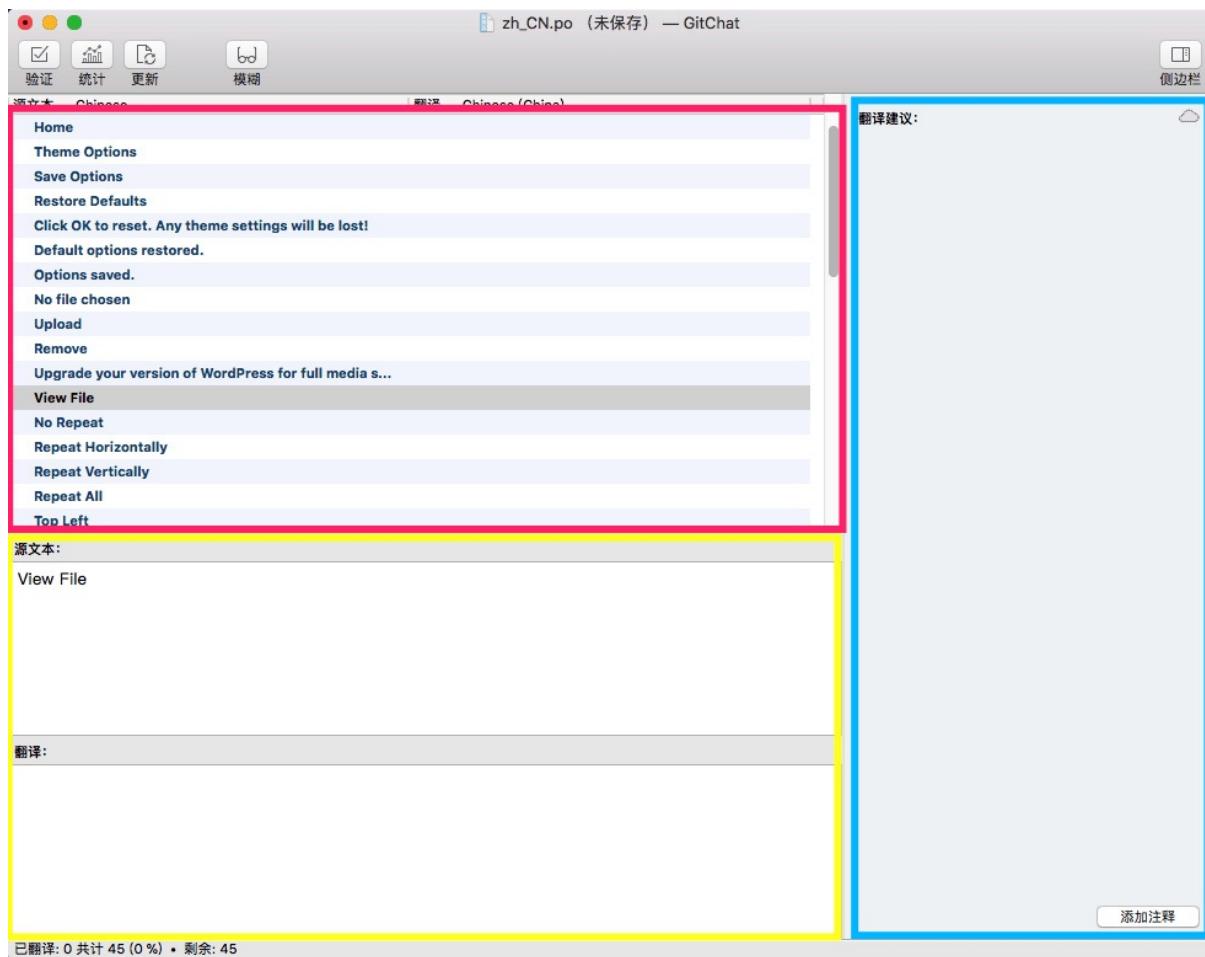
生成了语言包后，接下来翻译主题，点击界面上的创建新的翻译。



在新的窗口中选择我们要翻译的项目：



单击“确定”按钮，就可以进入到翻译的界面了。



红色区域是条目区域，点击我们要翻译的条目，相关信息就会在下方黄色的编辑区域显示出来。

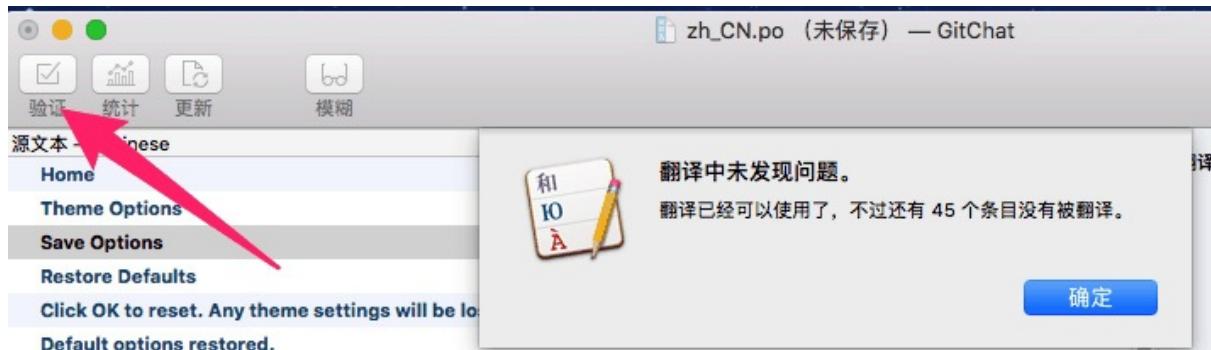
右侧的是建议区域，在这个区域，软件将会查询云端数据库和翻译平台，对当前内容进行翻译，给你提供参考。



如果曾经翻译过这个字段，这里还会显示你之前的翻译，供你选择。

验证翻译

当翻译完成后，点击上方工具栏中的验证按钮，Poedit 将会检查你的翻译是否出现了错误。



会提示当前翻译是否可以使用、还有多少个条目没有被翻译。

查看统计

单击上方的“统计”按钮，可以查看当前语言包的状况。



比如有多少个文本需要翻译、多少个文本需要检查等等。

更新翻译

单击上方的“更新”按钮，Poedit 会自动扫描主题目录下的代码文件，检查是否有新的需要翻译的字段加入进来。

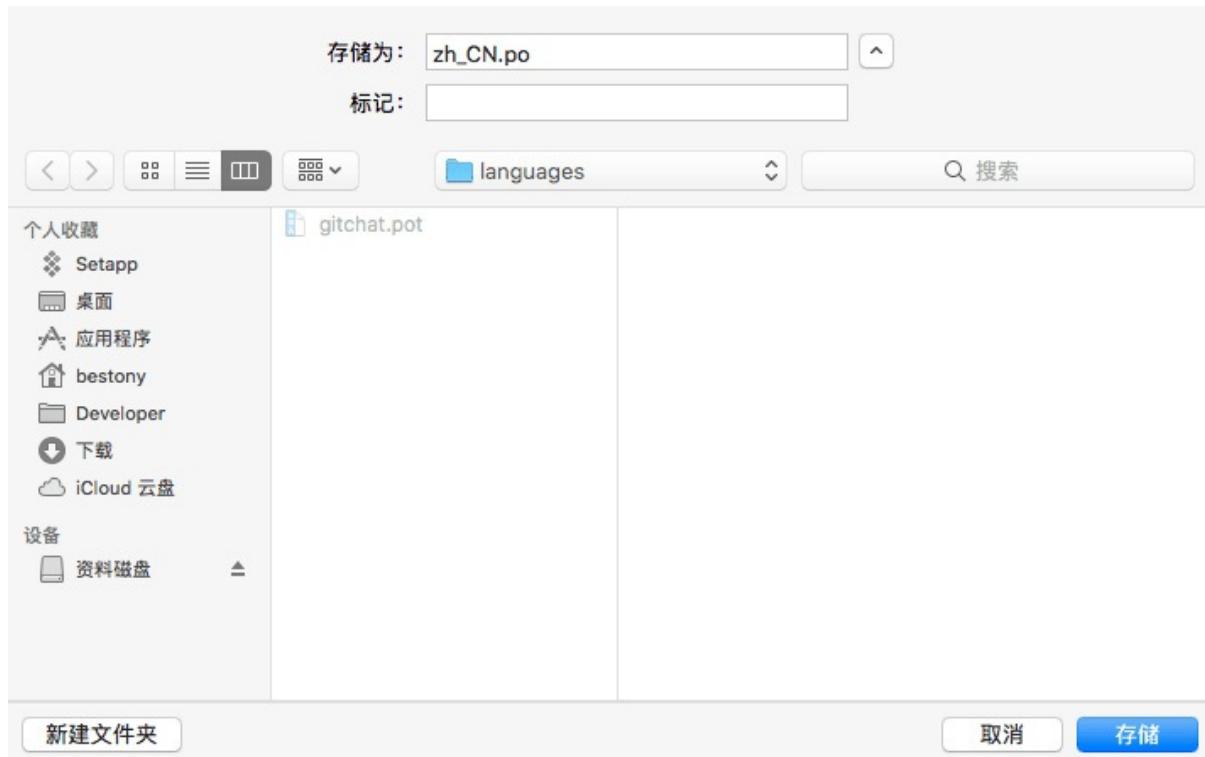


并在弹框中告诉你哪些是新的、哪些是旧的。

一般来说，可以直接单击“确定”按钮，但是如果代码没有发生任何改变，更新翻译后却提醒有了变化，就要仔细检查是否真的发生了代码变化。

保存文件

当翻译完成后，可以单击「文件」|「保存」命令，来保存你的翻译文件，这个文件需要保存到你的 language 文件夹下，并且以语言为文件名，就像这样：



保存后，会自动在这个文件夹下生成两个文件，其中 `zh_CN.po` 文件是翻译的源码，可供我们修改使用 `zh_CN.mo` 则是 WordPress 用于多语言识别的二进制文件。



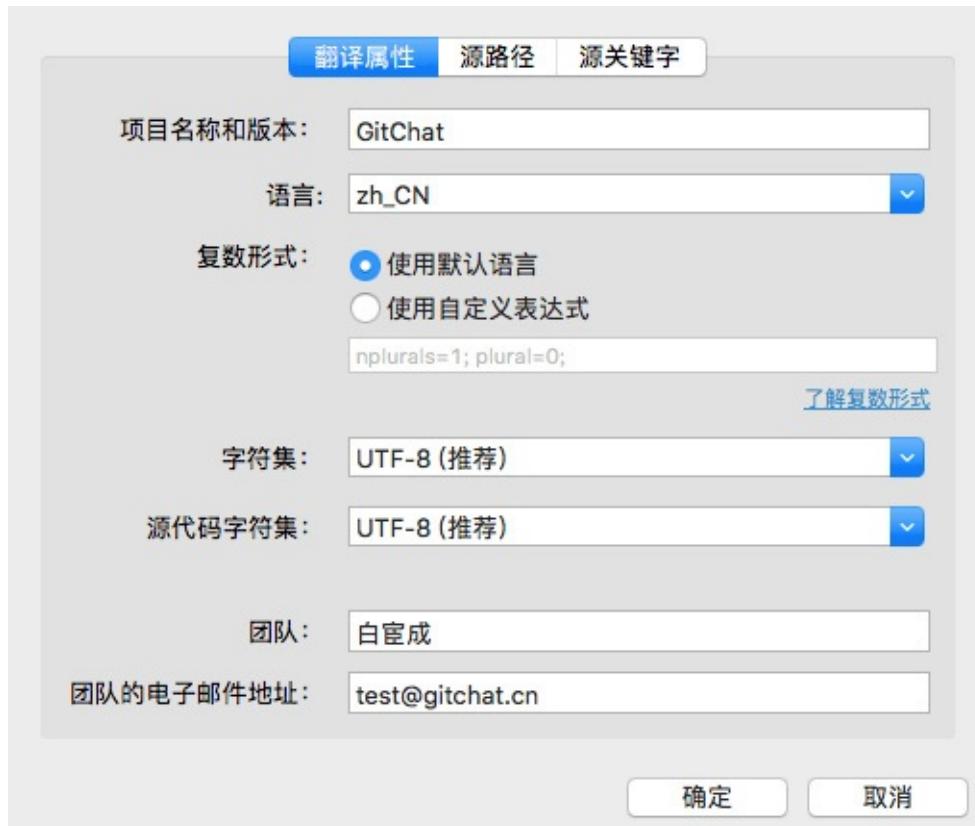
至此，我们完成了基本的翻译工作。

Poedit 的高级用法

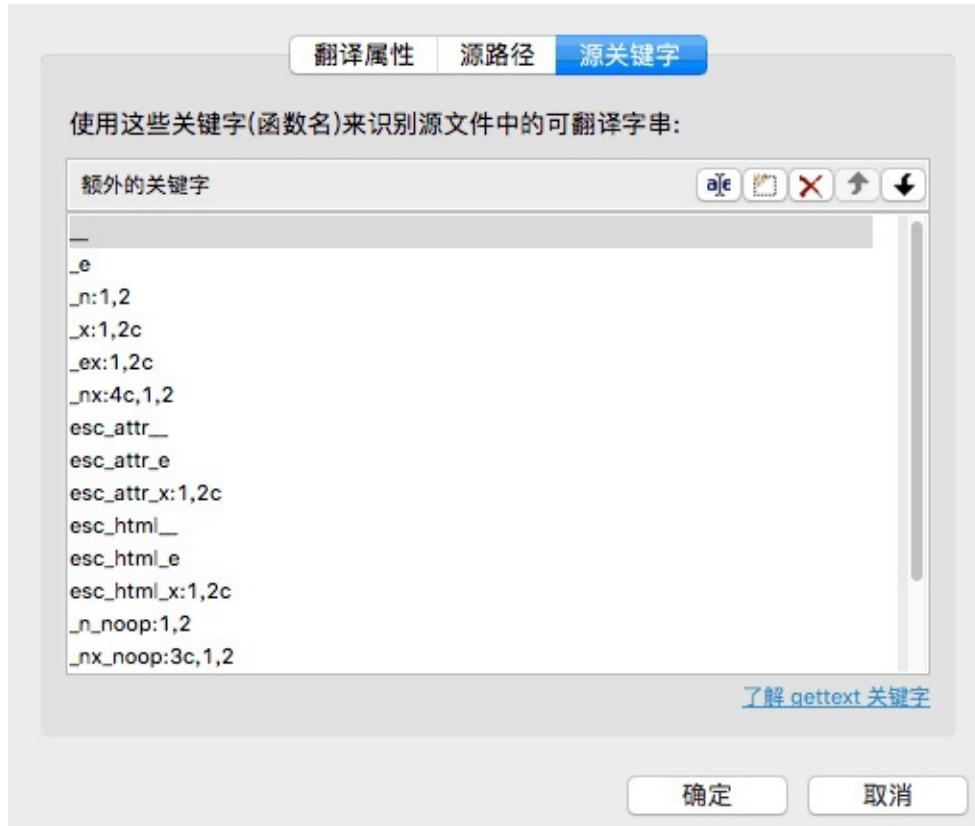
翻译文件属性设置

在使用 Poedit 时，应该设置一下你的自己的译者信息。

单击「编目」|「属性」命令，可以看到属性设置，可以修改信息为自己的：



在「源关键字」一项中，可以看到所有 Poedit 会识别的 WordPress 的多语言输出函数：



可以到 WordPress 的官网文档中查询这些函数的用法，使用这些函数来完成你的开发。

自动填充翻译

我之所以喜欢使用 Poedit 来做翻译，很大程度上是因为 Poedit 的记忆库功能，Poedit 的记忆库功能可以学习我以往的翻译内容，来填充记忆库，当我再进行翻译时，会自动从记忆库匹配完全一样的词条出来，这让翻译有了累计效应，你翻译和处理的主题、插件越多，翻译、处理主题就越快，因为很多内容都是之前曾经翻译过的，只需要再判断是否符合当前这个场景即可。

单击菜单栏中的「编目」—「从 TM 中填补缺少的翻译」命令，可以勾选第一项，这样就只匹配完全一致的词条，可以提升准确率，同时减少工作：



如果你已经翻译过多次，而且每次匹配的效果都不错，那么可以勾选第二项。如果没有勾选，匹配的项目会被标记为模糊，还需要手动确认是否匹配。

总结

经过这节课我们学习了如何汉化一个主题，插件的操作基本是一致的，将核心代码添加到插件的核心文件即可。

使用 WPML 插件构建一个多语言站点

这节课来学习如何使用 WPML 来创建一个 多语言站点。

购买插件

WPML 是付费插件，可以到[官网](#)购买。

如果不想购买，可以去看第29课 Polylang 的内容，在下一课中，将使用免费（主要功能）的 PolyLang 来创建多语言站点。

初始化配置

安装并启用 WPML 后，会有一个初始化的过程，首先我们初始化一下插件，单击「配置 WPML」按钮，进入配置的页面。

您需要配置WPML，然后才能开始翻译。

配置WPML 入门指南

将语言设置为中文：

1. 内容语言 2. 翻译语言 3. 语言切换器 4. 注册

当前内容的语言

在添加其他语言之前，请选择已有内容的撰写语言：

Chinese (Simplified) ▾

下一步

然后选择我们要支持的语言，这里依然选择英文：

网站语言

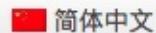
为您的网站选择要启用的语言（您以后也可以添加或删除语言）。

- | | | |
|--|--|--|
| <input type="checkbox"/>  Malay | <input type="checkbox"/>  世界语 | <input type="checkbox"/>  丹麦语 |
| <input type="checkbox"/>  乌克兰语 | <input type="checkbox"/>  乌兹别克语 | <input type="checkbox"/>  乌尔都语 |
| <input type="checkbox"/>  亚美尼亚语 | <input type="checkbox"/>  依地语 | <input type="checkbox"/>  俄语 |
| <input type="checkbox"/>  保加利亚语 | <input type="checkbox"/>  克罗地亚语 | <input type="checkbox"/>  冰岛语 |
| <input type="checkbox"/>  加泰罗尼亚语 | <input type="checkbox"/>  匈牙利语 | <input type="checkbox"/>  印地语 |
| <input type="checkbox"/>  印度尼西亚语 | <input type="checkbox"/>  土耳其语 | <input type="checkbox"/>  威尔士语 |
| <input type="checkbox"/>  尼泊尔语 | <input type="checkbox"/>  巴斯克语 | <input type="checkbox"/>  希伯来语 |
| <input type="checkbox"/>  希腊语 | <input type="checkbox"/>  库尔德语 | <input type="checkbox"/>  德语 |
| <input type="checkbox"/>  意大利语 | <input type="checkbox"/>  拉丁语 | <input type="checkbox"/>  拉脱维亚语 |
| <input type="checkbox"/>  挪威布克莫尔语 | <input type="checkbox"/>  捷克语 | <input type="checkbox"/>  斯拉夫语 |
| <input type="checkbox"/>  斯洛文尼亞语 | <input type="checkbox"/>  旁遮普语 | <input type="checkbox"/>  日语 |
| <input type="checkbox"/>  法语 | <input type="checkbox"/>  波兰语 | <input type="checkbox"/>  波斯尼亚语 |
| <input type="checkbox"/>  波斯语 | <input type="checkbox"/>  泰米尔语 | <input type="checkbox"/>  泰语 |
| <input type="checkbox"/>  爱尔兰语 | <input type="checkbox"/>  爱沙尼亚语 | <input type="checkbox"/>  瑞典语 |
| <input type="checkbox"/>  盖丘亚语 | <input type="checkbox"/>  祖鲁语 | <input type="checkbox"/>  立陶宛语 |
| <input checked="" type="checkbox"/>  简体中文 | <input type="checkbox"/>  索马里语 | <input type="checkbox"/>  繁体中文 |
| <input type="checkbox"/>  罗马尼亚语 | <input type="checkbox"/>  芬兰语 | <input checked="" type="checkbox"/>  英语 |
| <input type="checkbox"/>  荷兰语 | <input type="checkbox"/>  葡萄牙语 (巴西) | <input type="checkbox"/>  葡萄牙语 (葡萄牙) |
| <input type="checkbox"/>  蒙古语 | <input type="checkbox"/>  西班牙语 | <input type="checkbox"/>  赛尔维亚语 |
| <input type="checkbox"/>  越南语 | <input type="checkbox"/>  阿尔巴尼亚语 | <input type="checkbox"/>  阿拉伯语 |
| <input type="checkbox"/>  韩语 | <input type="checkbox"/>  马其顿语 | <input type="checkbox"/>  马耳他语 |

接下来配置语言切换器，在语言切换器这里，可以通过拖拽来调整我们语言展示顺序以及当某篇文章没有对应的内容时的跳转机制。

语言顺序

拖放语言更改其顺序



如何处理没有翻译的语言

跳过语言

缺少译文则链接至语言首页

下面这些选项，则根据你自己的个人需要来设置，可以将语言转换器添加到小工具和页脚，来让访客更加容易切换语言。

语言切换器菜单

+ 添加新的语言切换器到菜单

语言切换器小部件

+ 添加新的语言切换器到小部件区域

页脚语言切换器 ?

在页脚显示语言切换器

文章翻译的链接 ?

在文章上方或下方显示链接，并以其他语言提供

设置完成后，就是输入你在 WPML 官网购买的授权进行激活。

设置插件

WPML 提供了非常多的设置项，可以根据自己的需要来进行设置。

首先是根据你的需要，可以设置默认的语言以及增加新的语言、删除不再使用的语言。

网站语言

为这个网站启用了以下语言：

简体中文 (默认)

英语

[更改默认语言](#) [添加/删除语言](#)

[Edit Languages](#)

接下来是 URL 格式，可以根据自己的需要，选择不同的语言格式，比如子目录格式、不同域名的格式、参数格式等等。推荐大家使用目录，需要配置的内容较少，而且 URL 也比较美观。

语言URL格式

选择如何确定访客以哪种语言查看内容

不同的语言在不同目录中 (<http://icaci2018.php/> - 简体中文, <http://icaci2018.php/en/> - 英语)

对默认语言使用目录

各语言使用不同的域

语言名称作为参数添加 (<http://icaci2018.php?lang=en> - 英语)

[保存](#)

语言切换器部分大体上和我们初始化的内容是一致的，不过这里有几个多出来的选项，我说明一下：

语言切换器选项

您网站中所有的语言切换器都受此部分中的设置的影响。

语言顺序 ?
拖放语言更改其顺序

英语 简体中文

如何处理没有翻译的语言 ?
 跳过语言
 缺少译文则链接至语言首页

保留URL参数

额外的CSS ?

向后兼容 ?
 跳过向后兼容性

这里保留 URL 参数可以让语言切换器在切换语言时，带上一定的参数，比如 `version`、`time` 等信息，在这里设置参数，如 `version`、`time`，然后当通过链接进入时，链接中的版本信息和时间信息就被保留下来了。

额外的 CSS 可以为语言选择器添加一些 CSS 样式，使其适配我们网站的风格。

向后兼容选项是否开启取决于你的主题是否需要，你的主题是最近更新过的，可以不勾选这个选项了。

这四个选项我们之前说过，不再说明：

语言切换器菜单 ? + 添加新的语言切换器到菜单

语言切换器小部件 ? + 添加新的语言切换器到小部件区域

页脚语言切换器 ? 在页脚显示语言切换器

文章翻译的链接 ? 在文章上方或下方显示链接，并以其他语言提供

自定义语言切换器这里允许我们根据自己的需要设置语言切换器的样式，如果有特殊的切换需要，可以在这里调整。



要显示的博客文章则是设置你的站点在不同语言下的显示策略，如果某篇文章不存在对应版本的翻译，该如何选择？

隐藏语言则可以帮助你将某些语言隐藏起来，方便调试，调试完成后，再重新开放出来。

使主题多语化可以为你的主题加入多语言的支持，一般都是勾选的。

浏览器语言重定向则根据你自己的需要选择，是否需要根据用户的浏览器语言自动切换到对应的语言中。可以根据自己的需要设置，不过建议禁用掉，因为如果随意跳转，可能会让你的访客产生迷惑。

SEO 选项则是在页面的头部加入其他语言的链接，勾选上即可。

浏览器语言重定向

WPML能够根据浏览器语言自动重定向访客。
此功能使用Javascript。请确保您的网站没有JS错误。

禁用浏览器语言重定向
 仅当译文存在时才基于浏览器语言重定向访客
 总是基于浏览器语言重定向访客（如果缺少译文，则重定向至首页）

记住访客的语言偏好 个小时（请输入24或其倍数）。

SEO选项

在头部显示其他语言。

hreflang 链接的位置

用于AJAX操作的语言过滤

存储语言cookie，以便支持AJAX的语言过滤 [?](#)

用于 Ajax 操作的语言过滤：这一项取决于你的主题是否使用了 Ajax，如果使用了，则可以开启这个选项。如果没有使用，可以不开启这个选项。

翻译目录

在 WPML 中，对目录的处理要在插件中进行，单击「WPML」—「分类翻译」命令，就可以进入到目录的翻译页面，选择你要翻译「目录」或「标签」。

可以看到已经创建的目录：



显示为一个地球图标，则说明这个目录的主语言为当前所对应的语言，而后面的加号，则表示需要为这个目录创建一个对应的目录。后面显示的铅笔的，则说明已经为这个目录创建了对应目录，接下来可以点击铅笔对这个目录进行修改。

点击加号新增对应的目录，可以很方便的设置对应的内容。点击中间的按钮，可以将左侧的内容复制到右侧。设置完成后，单击“保存”按钮，即可创建对应语言的目录了。



设置菜单

单击「外观」—「菜单」命令，可以看到不同的语言：

The screenshot shows the WPML menu editor interface. On the left, there's a sidebar with categories: 'Page' (selected), 'Recent', 'View all', 'Search', 'Example page', 'Select All' (highlighted in blue), and 'Add to menu'. Below these are 'Article', 'Custom Link', 'Category', 'Tag', and 'Form'. On the right, the main panel has 'Menu Name' set to 'Main Menu', 'Language' set to 'Simplified Chinese', and a 'Menu Structure' section with the instruction 'Add items from the left sidebar'. A 'Menu Settings' section includes 'Automatically add pages' and 'Display location'. At the bottom right is a red 'Delete menu' button.

切换不同的语言，可以看到不同语言下的主菜单，然后勾选菜单的显示位置即可。

对于有多个菜单的语言，可以在上方的选择器中切换对应的菜单。

This screenshot shows the WPML menu editor with a different configuration. It has a top bar for selecting the menu ('Main Menu - English') and a note to 'Select or Create New Menu'. Below is a sidebar with 'Page' selected. The main panel shows 'Menu Name' set to 'Main Menu - English'.

翻译文章

对于我们来说，最主要的还是创建文章和页面，这里我以文章举例。

打开「文章」—「写文章」页面，就可以开始创建文章了，点击页面右侧的语言链接，可以切换我们正在写的文章是属于哪一门语言的。



或者，可以点击菜单栏中的语言选项，来切换当前正在操作的语言。



切换了语言就会发现，不同语言下的目录是不同的，需要根据不同语言，创建不同的目录。



扩充你的 WPML 能力

WPML 提供了多种插件，可以安装这些拓展插件来提升 WPML 的能力，实现更好的管理。

插件列表[详见这里](#)。

总结

至此，我们就完成了使用 WPML 来创建一个多语言站点的操作，如果你希望使用一个免费版本的插件来构建多语言站点，下节课的 Polylang 不会让你失望。

使用 Polylang 插件建设一个多语言站点

这节课来学习使用 Polylang 插件创建一个多语言站点。

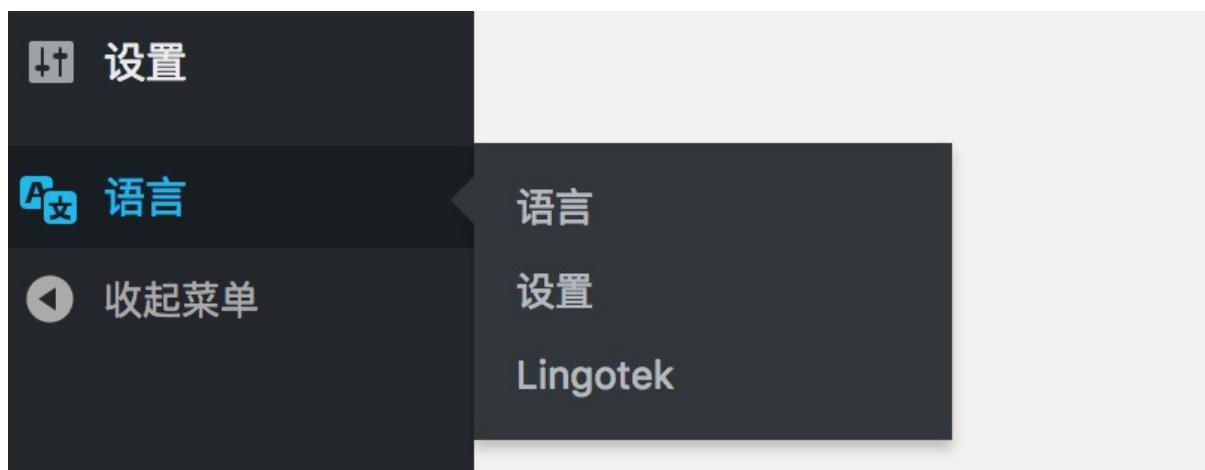
安装插件

进入 WordPress 后台，单击「插件」—「安装插件」命令，在搜索框内输入关键词 *polylang*，然后进行搜索，找到插件后进行安装，并启用插件。



初始化插件

启用成功后，后台会多出一个「语言」菜单项：



单击「语言」—「语言」命令，来添加一个语言，我们最常用的便是使用英文，这里也添加一个英文。

显示选项 ▾

语言

添加新语言

选择语言

English - en_US

你可以列表中选择一个语言或者直接在下面编辑它。

全名	区域	代码	★ 顺序	国旗	文章
找不到条目。					

关于 Polylang

Polylang 提供了一个全面的 [帮助文档](#) (英文而已)，它包括使用 Polylang 创建你的多语言网站和如何日常使用它，以及帮助程序员如何使 Polylang 适应他们的主题和插件。支持和额外功能仅 [Polylang Pro](#) 用户可用。

Polylang 发布授权与 WordPress 相同，[GPL](#)。

全名
English
这个名字是用来在网站上显示的 (例如: English)

区域
en_US
WordPress 语言区域 (例如: en_US) 安装 .mo 文件。

代码
en
语言代码 - 最好是2个字母的ISO 639-1 (例如 en)

文字方向
 从左到右
 从右到左
选择语言的文字方向

国旗
United States
为语言选择一个旗帜标志。

最后有个顺序，可以根据你的需要进行调整，序号越小，排的越靠前。

这里我们添加两种语言，分别是英文和中文，其中中文作为默认语言。

全名	区域	代码	★	顺序	国旗	文章
English	en_US	en		0	🇺🇸	1
中文 (中国) 编辑 删除	zh_CN	zh	★	0	🇨🇳	1
全名	区域	代码	★	顺序	国旗	文章

2个项目

2个项目

接下来要到前台为用户提供一个切换语言的工具，这里可以使用 Polylang 自带的 widget 来实现。

单击「外观」|「小工具」命令，进入到小工具页面，拖拽其中的「语言切换」小工具，到我们的边栏中去。

语言切换

显示一个语言切换

添加时，会提示你具体的样式，可以根据站点风格选择对应的设置项，不过，如果选择了显示下拉菜单，其他几项就会消失，仅保留隐藏没有翻译的语言和强制链接到主页。我这里选择显示语言名称和国旗。



然后回到首页，就可以看到我们的小工具了。

选择一门语言

English
 中文(中国)

翻译界面

现在已经创建好了语言，接下来就需要手动为文字提供翻译了。

单击「语言」|「字符串翻译」命令，就会进入到翻译的界面。在这里，可以将 WordPress 站点本身的字段翻译成各种语言，从而使得站点的界面能够匹配不同语言的内容。

	副标题	WordPress	English 中文(中国)	Another WordPress Site 另一个WordPress站点
<input type="checkbox"/> 又一个WordPress站点				
<input type="checkbox"/> Y年n月j日	日期格式	WordPress	English 中文(中国)	Y Years n month j days Y年n月j日
<input type="checkbox"/> ag:i	时间格式	WordPress	English 中文(中国)	ag:i ag:i
<input type="checkbox"/> Select A Language	小工具标题	Widget	English 中文(中国)	Select A Language 选择一门语言
<input type="checkbox"/> 字符串	名称	分组	翻译	
批量操作 <input type="button" value="应用"/>				
<input type="checkbox"/> 清除未使用的字符串翻译 使用它从数据库中删除未使用的字符串，例如一个插件被卸载以后。				

翻译文章和页面

虽然现在翻译完了界面，但是真正的重头戏文章还没有翻译，接下来以文章为例，来说明这部分主要内容的翻译。

Polylang 将站点内容根据语言的不同，划分了两个不同的体系，比如我这里是中文和英文，我的文章、目录、标签，两种语言各自有一套，互不干扰，所以你在使用时也要注意，两个语言版本的目录、标签的建立。

目录和标签的建立非常简单，单击「文章」|「分类目录」命令，就可以进入到目录的页面，在这个页面，进行目录的创建。

添加新分类目录

名称

这将是它在站点上显示的名字。

别名

“别名”是在URL中使用的别称，它可以令URL更美观。通常使用小写，只能包含字母，数字和连字符(-)。

父级分类目录

无

分类目录和标签不同，它可以有层级关系。您可以有一个“音乐”分类目录，在这个目录下可以有叫做“流行”和“古典”的子目录。

图像描述

描述只会在一部分主题中显示。

语言

中文 (中国)

设置语言

翻译

添加新分类目录

目录创建的其他部分和之前都一样，这样要注意的是「语言」和「翻译」，一般情况下，我们只设置默认语言，而不在创建时添加，只是为了方便我们将不同的目录串联起来。

选择「语言」下拉框，可以设置我们目前创建的这个目录以哪个语言为主，翻译则暂时留空。

名称	图像描述	别名	美国	中国	总数
GitChat	—	gitchat			0
测试	—	测试			0
Service	—	service			0
Test	—	test			0

在这里可以看到我创建了一个 GitChat 目录，而测试和 Test 目录则是之前创建的目录，这两个目录是同一个目录在不同语言下的展示。

在红色框中的部分，显示一个对勾的，则说明当前的这个目录的主要语言是哪个，比如 GitChat 目录的主要语言是中文，测试目录的主要语言也是中文，Test 目录的主要语言则是英文。

显示图标为一个铅笔，则说明这个语言已经创建了，单击这个钱包，就可以修改对应语言的目录。

显示图标为一个加号，则说明我们还没有为这个语言创建对应的目录，点击加号，来创建目录，这个时候只需要填写目录的名称、别名和图像描述即可，PolyLang 会自动把两个目录关联起来，非常的方便。

添加新分类目录

名称



这将是它在站点上显示的名字。

别名

“别名”是在URL中使用的别称，它可以令URL更美观。通常使用小写，只能包含字母，数字和连字符(-)。

父级分类目录

无 

分类目录和标签不同，它可以有层级关系。您可以有一个“音乐”分类目录，在这个目录下可以有叫做“流行”和“古典”的子目录。

图像描述

描述只会在一部分主题中显示。

语言

 English 

设置语言

翻译

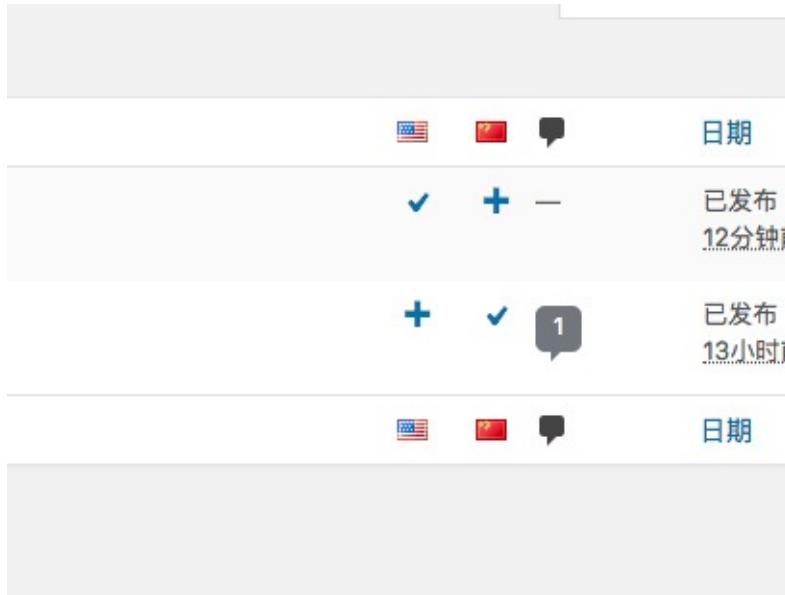
 GitChat

添加新分类目录

标签的实现方法类似，这里就不再赘述了。

接下来我们来翻译文章，单击「文章」|「所有文章」命令，就可以进入文章列表。

在文章列表中可以看到和目录页面一样的按钮，这个按钮提示了这篇文章的主语言是什么、是否已经有了对应语言的翻译。

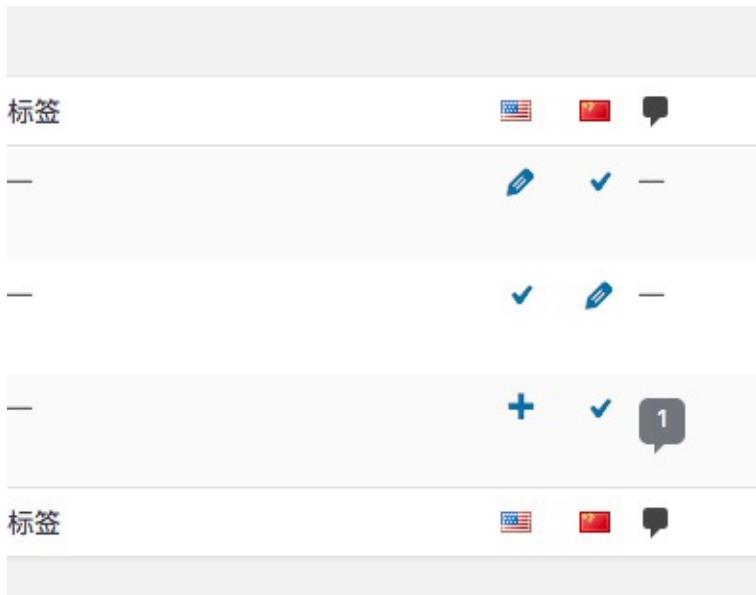


和目录操作一样，点击加号，就可以为某一篇文章创建特定语言的文章。

在添加文章时，可能会注意到，这里的目录只有以当前语言为默认语言的目录，我们为其他语言测试的目录是不存在的，也从另一个方面印证了之前说的内容。



输入文章内容，单击“发布”按钮，即可发布一条新的文章。回到文章列表，就可以看到新的文章了。



同时，这两篇文章的对应的图标也从加号切换为乐铅笔。

这样就完成了对已有内容的处理。如果想要发布新的文章，过程大体类似，先创建你的主要语言的文章，再创建其他语言的文章即可。

处理菜单项

由于是双语站点，所以如果有菜单项，菜单项也应该是对应不同的。

进入到菜单项中，就可以看到菜单位置变成了两个，每个语言各有一份。

- 主菜单 English
- 主菜单 中文 (中国)

根据你的语言，设置对应的语言的菜单，然后将他们放在对应的菜单位置上即可。

插件设置项

插件默认为我们提供了一些设置项，之后来具体说一说，PolyLang 为我们提供了这些设置项。

模块	Description
URL修改 设置	决定您的链接地址看起来什么样。
检测浏览器语言 禁用	当主页被访问时，根据浏览器偏好设置语言
媒体 禁用	为多媒体启用语言和翻译
自定义文章类型和分类法 已禁用	为自定义文章类型和分类法激活语言和翻译管理。
同步 设置	同步选项允许在一篇文章或页面的翻译之间保持相同的元数据内容的值（或分类法和 taxonomies and page parent）of meta content between the translations of a post.
WPMU 兼容性 已激活	WPMU 兼容模式
共享另名 已禁用	允许通过语言为文章和分类项共享相同的链接别名
您需要 Polylang Pro 版本才能启用此功能。 现在升级。	
翻译别名 已禁用	允许翻译自定义文章类型和分类法链接中的别名。
您需要 Polylang Pro 版本才能启用此功能。 现在升级。	
工具 设置	当卸载 Polylang 插件时是否删除所有数据。
许可密钥 已禁用	管理 Polylang Pro 或插件的许可证。
模块	Description

在 URL 修改中，我们可以设置文章/页面的链接规则：

URL修改

语言根据内容来设置
文章、页面、分类和标签的URL不会被修改。
例如： `http://icaci2018.php/en/my-post/`

语言根据链接的目录名来设置
例如： `http://icaci2018.php/en/my-post/`

语言根据链接的二级域名来设置
例如： `http://en.icaci2018.php/my-post/`

语言根据不同的域名来设置

为默认语言隐藏 URL 语言信息
 在链接中移除 /language/
例如： `http://icaci2018.php/en/`

在链接中保留 /language/
例如： `http://icaci2018.php/language/en/`

[取消](#) [保存更改](#)

一般来说选择第二个和或者第三个。如果你在同一台虚拟主机上绑定了多个域名，则使用第三个选项，将所有你用到的域名绑定在这里；如果你的虚拟主机只允许绑定一个域名，则选择第二个选项，第四个选项和第三个选项操作的要求是一致的，需要你的主机能够绑定多个域名。不过我们大部分时候都不会使用多个域名，所以就不再说明，不过如果你使用，其设置也是非常简单的，填写对应域名即可。

语言根据不同的域名来设置

English

中文 (中国)

<http://.../en/>

取消

保存更改

之所以推荐大家使用后面几种而不是第一种，主要是因为如果链接内容中有了语言信息，用户就可以将站点保存在书签中，而无需每次都手动切换至某个特定的语言。

对于右侧的设置项，上面的隐藏默认语言的 url 信息，是指对于默认语言是否也显示后面的语言后缀。而下面的选项，则是判断是否在链接中显示 language，一般来说，我们尽可能不显示，以减少 URL 的长度。

为默认语言隐藏 URL 语言信息

在链接中移除 /language/

例如： <http://.../en/>

在链接中保留 /language/

例如： <http://.../language/en/>

检测浏览器语言

禁用

当主页被访问时，根据浏览器偏好设置语言

媒体

禁用

为多媒体启用语言和翻译

下方的检测浏览器语言和媒体默认开启，我也推荐大家开启，这样站点的体验会更好，用户访问时会自动切换到对应的语言。

在同步这里，可以设置我们的文章需不需要同步一些状态（Meta Data），由于在不同的语言下，可能有不同的发布策略，这里推荐不选。不过，如果发布策略是不同语言文章的各项属性相同，同期发布，则可以在这里勾选要同步的项目。

同步

分类法
 文章格式

自定义字段
 父级页面

评论状态
 页面模板

Ping 状态
 页面顺序

置顶文章
 特色图像

 发布日期

取消

保存更改

WPML 兼容性提供了 PolyLang 对 WPML 的兼容，这里就不多说了。

WPML 兼容性

已激活

WPML 兼容模式

对于工具选项来说，如果你不是在也不打算用这款插件了，这里就不要勾选，这样卸载了插件，数据还在，后续还可以安装上插件，继续使用。

不过如果你的翻译内容太烂，想重建一份翻译，就可以先勾选这里选项，再删除插件。

工具

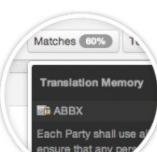
当在插件页面点击“删除”按钮时删除所有 Polylang 数据。

取消

保存更改

Lingotek

Lingotek 整合了一些翻译服务，可以借助这些翻译服务，实现内容的自动化翻译，而无需自己人工翻译。

<p>秃头鹰</p>  <p>秃头鹰，健壮的鸟喙，带浅色的小巧嘴，是世界上最重的猛禽。它们是独居的，通常在森林中筑巢。它们是唯一一个能飞得比其他猛禽更高的鸟。它们有“三趾脚”，前脚有三根脚趾，后脚有两根脚趾。它们的视力很好，翅膀很大，不需要扇动翅膀就能飞翔。</p> <p>当秃头鹰捕获一些老鼠，它们会将它们带回巢穴，在 40 英尺 (12 米) 的高度上吃掉它们。它们的视力非常好，可以在 1 英里的距离内看到老鼠。它们的翅膀非常有力，可以飞到 10,000 英尺的高度。它们的羽毛非常柔软，以至于它们在飞行时几乎感觉不到风的存在。它们的爪子非常锋利，足以撕裂任何东西。它们的视力非常好，以至于它们可以在 1 英里的距离内看到老鼠。它们的翅膀非常有力，可以飞到 10,000 英尺的高度。它们的羽毛非常柔软，以至于它们在飞行时几乎感觉不到风的存在。它们的爪子非常锋利，足以撕裂任何东西。</p> <p>了解更多...</p>	 <p>翻译管理系统</p> <p>您需要连接到一个专业的翻译管理系统么？</p> <p>激活</p> <ul style="list-style-type: none"> • 为你的网站获得上限为100 000字的免费机器翻译。 • 机器翻译是低预算情况下的理想选择。接近即时翻译，结果也还算可以接受 <p>了解更多...</p>	 <p>专业的翻译我的站点</p> <p>您需要专业的翻译您的站点么？</p> <p>激活 请求翻译</p> <ul style="list-style-type: none"> • 开始专业翻译竞标程序。 • 激活账号以便于Lingotek可以获取你的站点上需要翻译的精确字数以及需要翻译的语言。 • 一旦点击请求翻译报价，一个认证过的翻译管理者将会联系您，并给您一个无合约的报价。 <p>了解更多...</p>	 <p>需要额外服务？</p> <p>您需要翻译站点方面的帮助么？</p> <p>激活 请求服务</p> <ul style="list-style-type: none"> • 开始额外服务的进程。 • 您需要某人来运行您的本地化项目？ • 您需要定制工作流么？ • 您有已经存在的翻译记忆以供使用么？ • 您需要创建术语集方面的帮助么？ <p>了解更多...</p>
---	--	---	--

这里可以根据你自己的需要来开启对应的服务。

单击“激活”按钮后，会为你安装一个新的插件，在这个插件中链接你的 Lingotek 服务，就可以实现机器翻译了。

总结

经过这节课，我们学习了如何使用 PolyLang 来创建一个双语站点，对于需要做外贸网站、多语言平台的你来说，或许有所帮助。

至此，达人课的主要内容就更新完整了，如果在看达人课时有什么问题，欢迎到读者圈来和我一起讨论，后续我也将根据读者圈的提问情况，更新一些新的课程出来。

最后，感谢购买，祝学习愉快~

WordPress 数据库操作 WPDB 指南

在开发 WordPress 插件时，难免会遇到一些需要对数据库进行操作的功能。对于 WordPress 自带文章、页面、评论、用户等信息，可以通过 `query_posts` 来完成数据查询功能，但是，在一些特殊的场景中，我们可能需要自定义自己的数据结构。在这个时候，我们就需要考虑添加一个新的表，来存储我们自定义的数据。这时，就需要了解 WordPress 的 WPDB，并尝试借助 WPDB 来完成数据的查询。

在使用 WPDB 时，需要掌握一定的 SQL 基础，所以，如果还不会使用 SQL 对数据库进行增删查改，那么可能需要先学习一下 SQL 基础内容。

初识 WPDB

WPDB 是 WordPress 提供的操作数据库的类，借助这个类，我们可以轻松的实现对 WordPress 数据库的操作，从而完成自定义的数据更新的需求。

WPDB 在使用时，无需自行实例化，直接使用 WordPress 提供的 `$wpdb` 全局变量即可。下面给出一个调用 `$wpdb` 的例子。

```
global $wpdb;
$res = $wpdb->get_results( "your sql" );
```

执行查询

如果想要使用 WPDB 进行 SQL 语句的查询，可以使用 `$wpdb` 的 `query` 方法。

```
global $wpdb;
$wpdb->query('query');
```

这个方法将会返回所执行的 SQL 语句影响的行数。在使用时你需要注意，其返回值可能是 0 或者 `false`（表示语句有问题），你在进行结果的判断时，推荐使用 `==` 来进行对比。

获取一行数据

在执行数据库时，大部分时候需要查询单个行的数据，比如查看某个元素的单个条目的详情。这个时候，就可以使用 `get_row` 方法来获取一行数据。

```
global $wpdb;
$wpdb->get_row('query', output_type, row_offset);
```

在使用时需要注意，`get_row` 有三个参数：

- 第一个参数为需要执行的 SQL 语句。
- 第二个参数为需要输出的结果的类型，我们可以将其设置为 OBJECT，表示返回值是一个对象；也可以将其设置为 `ARRAY_A`，表示将返回值转换为 Key-Value 形式的数组，读者可以通过 `$array['myKey']` 的形式来获取结果；或者设置为 `ARRAY_N`，表示返回值为一个排序的数组，需要通过 `$array[1]` 来提取返回值的值。
- 第三个参数则是设置我们的数据是否要跳过多少行，默认为 0，即取第一个数据，如果设置了跳过的数据，则查询的数据会是跳过所设置条数后的数据的值。

获取通用查询结果

除了获取单个数据，还有很多场景下需要获取某一个类型的所有数据，如列表。在这个时候，可以使用 `get_results`。

```
global $wpdb;
$wpdb->get_results('query', output_type);
```

`get_results` 也可以应用于获取单行数据，不过我们有 `get_row`，因此，在获取单行数据时可以不用 `get_results`。

在使用 `get_results` 时，除了需要设置要执行的 SQL 语句外，还需要注意设置返回值的类型。和 `get_row` 一样，支持 `OBJECT`、`ARRAY_A`、`ARRAY_N`，`get_results` 还提供了 `OBJECT_K` 的类型。

当你设置为 `OBJECT_K` 时，返回值会被放在一个数组内，数组的 Key 将会是所查询数据中的第一列的值（一般来说，是我们查询时的 ID，比如文章的 ID 为 5，那这时该对象的 Key 就是 5）。

插入数据

除了查询数据以外，插入新的数据对于我们来说，也是非常重要的。WordPress 同样提供了插入数据的方法。

```
global $wpdb;
$wpdb->insert( $table, $data, $format );
```

执行 `insert` 方法时，需要设置表名（表名需要带上前缀，前缀可以通过 `$wpdb->prefix` 来获取到），并设置要添加的数据，这里传入的数据 `$data` 应该是一个数组类型的数据，并将其 Key 设置为数据库对应的字段名。`$format` 是插入数据的类型，默认为 `array`，不需要做修改，`string` 也是一个可选项，不过，我们以数组的形式插入数据更方便。

在执行完成 `insert` 方法后，可以获得插入后数据在表内的序号，我们可以通过 `$wpdb->insert_id` 来获取到这个 ID。

更新数据

在执行业务逻辑时，我们还会遇到需要更新数据的场景，此时，我们可以执行 `update` 方法来更新数据。

```
$wpdb->update( $table, $data, $where, $format = null, $where_format = null );
```

在使用 `update` 时，需要设置数据库表名（需要加入前缀）。`$data` 则需要传入我们要更改的数据，传入的值为数组类型，比如 `array('column1' => 'value1', 'column2' => 123)`；`$where` 也类似，需要传入我们要查询的条件 `array('column1' => 'value1', 'column2' => 123)`，`$format` 则是我们输入的 `data` 的类型，默认为 `array`，一般不需要修改；`$where_format` 是我们查询的 `where` 的类型，默认为 `array`，一般也不需要修改。

删除数据

WordPress 并未提供单独的删除数据的接口，所以，当需要删除数据时，可以使用 `get_results` 或 `query` 方法来执行 `DELETE` 语句，删除数据。

WordPress SQL 查询语句安全检查

WordPress 插件想要上架到官方的市场中，一个不可避免的问题就是通过官方的安全检查。事实上，WordPress 的插件审核团队也主要审核你的插件的安全问题，并不会太在意你的插件具体功能是什么。因此，在开发时，应该对你的 SQL 语句进行安全检查，从而，确保 SQL 语句足够安全，不至于出现安全问题（特别是你的查询依赖于用户的输入）。

在这种情况下，可以使用 WordPress 官方提供的安全审查函数 `prepare`，该函数可以将用户的输入进行安全转码，从而确保进入数据库的语句都是安全的。

```
$sql = $wpdb->prepare( 'query' [, value_parameter, value_parameter ... ] );
```

在使用时，我们需要使用 `%s` 这样的数据来替换 SQL 语句中的值，然后在第二个参数中的数组中传入对应的值，进而进行 SQL 的处理，确保 SQL 语句的安全，就像这样。

```
$wpdb->prepare( "
INSERT INTO $wpdb->postmeta
( post_id, meta_key, meta_value )
VALUES ( %d, %s, %s )",
10, $metakey, $metavalue )
```

这里支持的占位符包括 `%s` (文字、字符串) 、 `%d` (数字) 。

一些常用的变量

在进行数据查询时，可能会用到下面的这些变量，我列举出来，方便读者使用时查询。

- `$wpdb->prefix`：获取表前缀，这个非常重要，不要写死为 `wp_`，因为有的用户是自定义的。
- `$wpdb->num_rows`：获取最近查询的行数，这个比较有用，可以通过这个命令，获取到上一次查询的行数，然后显示在列表里。
- `$wpdb->insert_id`：获取最近插入数据的 ID，用处不是很多，可以通过这个值，来查看是否成功插入。
- `$wpdb->last_result`：获取最后一次执行查询的结果，可以免于查询，直接获取上次查询的结果，减少性能的损耗。
- `$wpdb->last_query`：获取到已经执行过的上一次查询，可以用于展示上一次查询的语句。

参考阅读

本篇内容仅介绍了一些常用的方法和变量，实际上，WPDB 还支持更多的方法和变量，如果上述的描述无法满足你的要求，可以前往官方的介绍页面，查看更多的 WPDB 方法。

WordPress 的官方介绍，[详见这里](#)。

动手开发插件：Custom Author 插件开发实战

今天分享一个插件的完整开发流程，帮助你来理解我的开发工作流，让你更好的开发出自己的 WordPress 插件。

起因

咱们 GitChat 的丹华老师找到我，向我咨询了两个问题。



丹华老师是咱们 GitChat 达人课《区块链全景课》的撰稿人，课程写的非常好，如果你对区块链也有兴趣，不妨去看看。

其中一个问题，我告诉他 WordPress 官方的解决方案，已解决掉了。但是第二个问题没解决，问题如下：

如何在发布文章时指定特定的作者？

遇到这个问题后，我先向丹华老师咨询了一个问题：

1. 你使用的作者是否会重复？

得到了丹华老师的否定的回答后，我就知道，我可能需要帮他找一款同类型插件来解决他的问题。如果使用的作者能够重复，那么其实可以通过 WordPress 内置的用户系统来实现，只需要在发布时选择作者就行了。不过因为很少会重复，手动的收入就很有必要了。

为什么要先沟通问题？

遇见问题时，程序员的天性总是希望通过造轮子来解决问题，但是，我们应该尽可能的控制自己，不要通过造轮子来解决问题。WordPress 的插件已经足够多了！完全可以通过沟通，来引导用户确定自己的需求，或对需求进行转化。尝试着用一些非技术的手段来解决问题！技术应该是核武器，不要轻易动用。

通过一段搜索后，我确定没有同类型的插件，便考虑写一个新的插件。

定名

在开始写插件的时候，难免要为插件起名，一个好的插件名可以帮助你引入更多的用户。由于我的插件非常简单，仅仅需要提供发布文章时的自定义作者，因此我初步拟定插件名为 Custom Author。

同名的插件你发布到 WordPress 时，访问的路径会是 `xxx-2`，非常不好看，所以我又对插件名称进行了确认，确认了该名称尚未被使用。

在为插件起名时，如何确定 WordPress 中没有同名的插件？

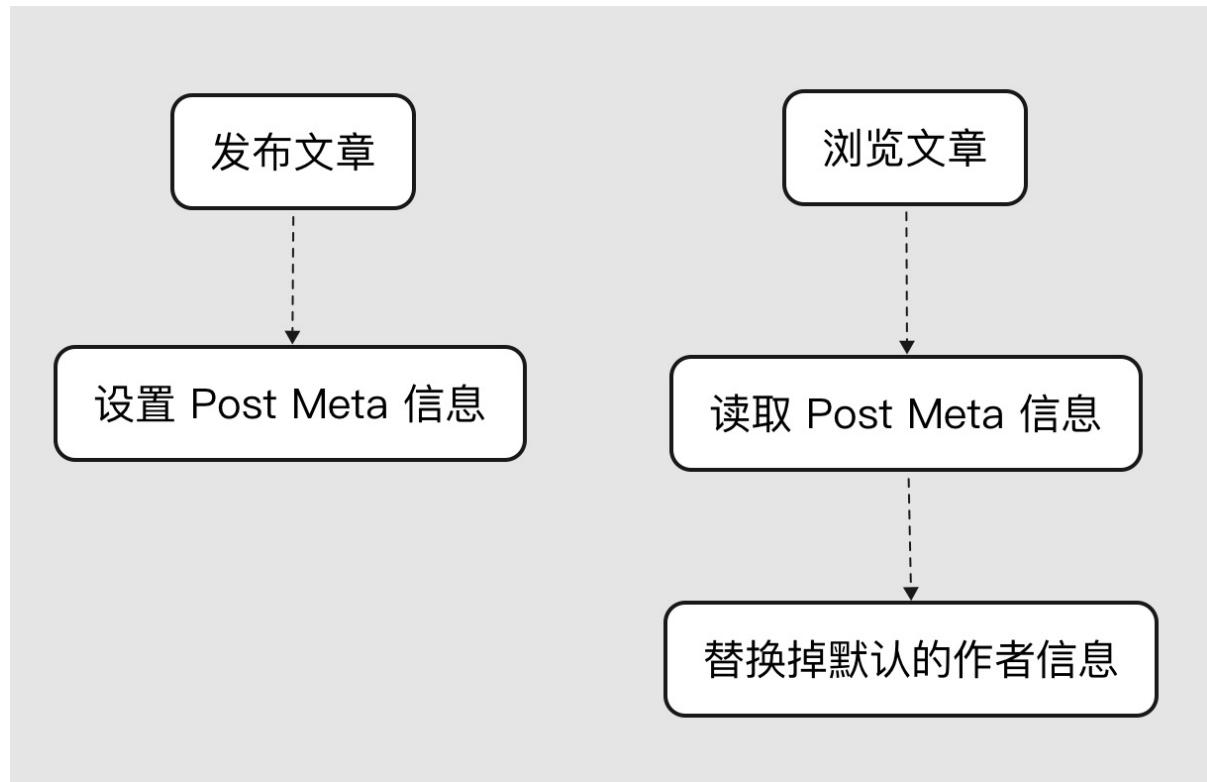
访问网址 <https://wordpress.org/plugins/插件名称>：

- 如果你看到的是搜索结果，就说明该名称尚未被使用，页面的路径也会被自动导流到 <https://wordpress.org/plugins/search/custom-author/>。
- 如果你看到的是插件的详情页面，则说明该名称已经被使用了。

思路

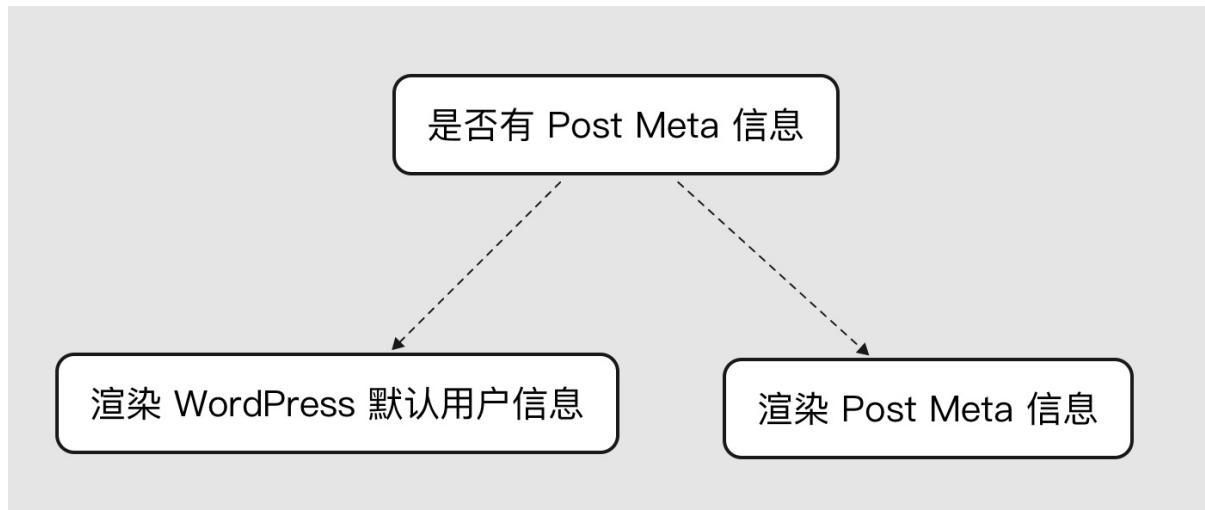
由于我需要为文章手动指定作者信息，而 WordPress 本身的用户系统中用户的创建需要输入较多信息，且容易造成用户的污染，因此，我考虑使用 Post Meta 信息来进行用户信息的设置。

同时，在浏览文章时，将对应的 Post Meta 信息读取，替换掉默认的作者信息。



此外，在替换作者信息时，也需要考虑一些边缘情况。需要考虑，如果某篇文章没有设置 Post Meta 信息的话，就需要让其显示默认的作者信息，或者设置一个默认作者信息（这个可以通过插件设置来实现，不过当前这个插件其实没有必要做这个功能，使用 WordPress 默认的用户信息也可以）。

这里经过考虑，我决定在浏览文章时，加入一层判定，如果设置了 Post Meta 信息，就读取 Post Meta 信息；未设置该信息时，就显示默认的用户信息。



在用户操作时，对用户最为友好的便是通过界面的手动输入来设置作者信息，因此，我考虑在界面中加入一个输入框，来作为作者信息输入。此处考虑到插件的特定用户是丹华老师，因此，我将这个输入框放在发布框内，方便他操作。



实现

在完成了插件功能的考虑后，接下来就进入到插件的开发阶段了。在刚刚，我们将插件的整个工作流程分为了两部分，前台的显示和替换以及后台的 Post Meta 设置。

后台功能

在这里，首先添加了两个 Action，用来在 WordPress 上的发布框中添加内容的输出，并添加了数据处理函数。

```
add_action('post_submitbox_misc_actions', 'cus_author_createCustomField');
add_action('save_post', 'cus_author_saveCustomField');
```

然后分别设置他们的处理函数 `cus_author_createCustomField` 和 `cus_author_saveCustomField`。

在 `cus_author_createCustomField` 中，我使用了如下的代码，来判断当前编辑的内容是否是 Post，也就是文章类型，从而可以设置仅对文章类型的页面输出插件的内容。

```
$post_id = get_the_ID();
if (get_post_type($post_id) != 'post') {
    return;
}
```

随后，提取文章的 Post Meta：

```
$value = get_post_meta($post_id, '_custom_author_name', true);
```

这里我设置第三个参数为 `true`，表明我希望返回值是单个对象，而不是一个数组。

并添加 nonce 安全处理，借助 nonce 安全处理，可以避免我们设置的接口被恶意请求。WordPress 在处理时，会进行安全校验。

```
wp_nonce_field('custom_author_nonce', 'custom_author_nonce');
```

以及使用 HTML 输出界面，此部分不再粘贴代码，相关代码在[详见这里](#)。

在 `cus_author_saveCustomField` 中，添加了如下一系列函数来进行校验：

```
if (defined('DOING_AUTOSAVE') && DOING_AUTOSAVE) {
    return;
}
```

上述代码可以避免自动保存功能触发的更新。

```
if (
    !isset($_POST['custom_author_nonce']) ||
    !wp_verify_nonce($_POST['custom_author_nonce'], 'custom_author_nonce')
) {
    return;
}
```

上述代码会检测是否提交了 nonce 信息，以及 nonce 信息是否是正确的。如果不正确，就返回，从而确保用户的请求一定是 WordPress 后台发起的，而不是用户通过其他渠道恶意发起的。

```
if (!current_user_can('edit_post', $post_id)) {
    return;
}
```

以及，判断用户是否有对应的权限编辑特定的文章，如果没有权限，就返回。

在完成了权限校验后，便是对用户输入进行保存：

```
if (isset($_POST['_custom_author_name'])) {
    update_post_meta($post_id, '_custom_author_name', sanitize_text_field($_POST['_custom_author_name']));
} else {
    /**
     * 不存在就删除
     */
    delete_post_meta($post_id, '_custom_author_name');
}
```

在这段代码中，需要注意的是，我使用了 `sanitize_text_field` 来对用户输入进行校验。你不应该相信用户的输入，而是尽可能的做好安全校验。

后台部分完整代码[详见这里](#)。

前台部分

在完成了后台部分的内容后，我们来实现前台的内容展示。在完成这部分功能时，不需要再使用 `add_action`，而是要使用 `add_filter`。

```
add_filter('the_author', 'cus_author_the_author');
function cus_author_the_author($author){
    $custom_author = get_post_meta(get_the_ID(), '_custom_author_name');
    if ($custom_author) {
        return $custom_author[0];
    } else {
        return $author;
    }
}
```

前台内容的展示逻辑非常简单，你只需要判断文章是否设置了自定义的作者名称。如果设置了自定义作者信息，就使用手动设置的值；如果没有设置，就使用函数传入的默认的作者名称。

提交审核

在完成了插件的开发后，我便筹划将插件提交到官方。毕竟，通过官方审核的插件能够让更多人信任，同时，使用时体验也更好，只需要在 WordPress 后台搜索即可。

Add Your Plugin

Once submitted, your plugin will be manually reviewed for any common errors as well as ensuring it complies with [all the guidelines](#).

Currently there are 25 plugins awaiting review.

Maximum allowed file size: 10MB

Even if you've submitted a dozen plugins, take the time to refresh your memory with the following information:

- [How to use SVN](#)
- [Plugin Assets \(and how to use them\)](#)
- [Developer FAQ](#)

前往 [Wordpress 网站](#) 提交插件审核，等待一晚上的时间（由于存在时差，插件的审核总是在我们的半夜进行），就会收到审核结果了。如果审核没有通过，只需要根据审核邮件的内容进行修改。

There are issues with your plugin code.

Please read this ENTIRE email, address all listed issues, and reply to this email with your corrected code attached (or linked). It is required for you to read and reply to these emails, and failure to do so will result in significant delays with your plugin being accepted.

Remember in addition to code quality, security and functionality, we require all plugins adhere to our guidelines. If you have not yet, please read them:

* <https://developer.wordpress.org/plugins/wordpress-org/detailed-plugin-guidelines/>

You will not be able to submit another plugin while this one is being reviewed, so please read the email carefully. We know it can be long, but you must follow the directions at the end as not doing so will result in your review being delayed.

修改完成后，将修改后的插件作为附件回复给审核者，再等一个晚上就会收到审核的意见。

Congratulations, your plugin hosting request for Custom Author has been approved.

Within one hour you will have access to your SVN repository with the WordPress.org username and password you used to log in and submit your request. Your username is case sensitive.

<https://plugins.svn.wordpress.org/custom-author>

Here are some handy links to help you get started.

审核通过后，就会收到 WordPress 官方的邮件，提醒你使用 SVN 管理插件，接下来就是准备插件的一些展示文件了，这部分内容可以参考一下咱们之前的达人课 [第19课：提交你的插件到 WordPress 的官方仓库](#)。

轻松玩转 WP：如何使用 WordPress 的邮件发文功能

在之前的读者圈内，有读者问我，如何通过邮件来发布 WordPress 文章，之前就说要补充这部分的内容，一直没顾上，今天把这个内容补充上来。

本篇内容主要分为两个部分，分别是如何利用 WordPress 自带的「通过电子邮件发布」功能（不推荐使用，仅作了解），并介绍了一个功能更强的发布插件「Postie」（推荐使用）。

使用「通过电子邮件发布」功能发布文章

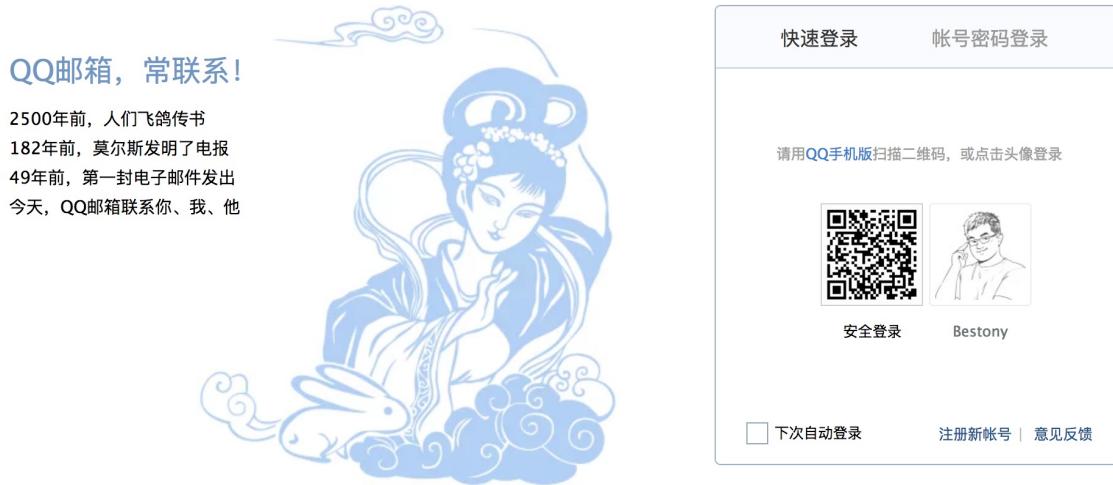
WordPress 本身提供了通过电子邮件发布的功能，如果你只是希望实现用电子邮件发布文章的能力，那么这个功能已经满足你的要求了。

不过，同样因为是 WordPress 官方提供的功能，在功能的设计上并没有我们所希望的那么完善，而且，根据官方的说明，这个功能也将在未来的某个时间点，从 WordPress 中移除掉（The built-in WordPress functionality is deprecated and will be removed in an upcoming release.）。好在目前这个功能并没有被去掉，而你不希望被众多插件拖慢的话，这个自带的功能还是很不错的。

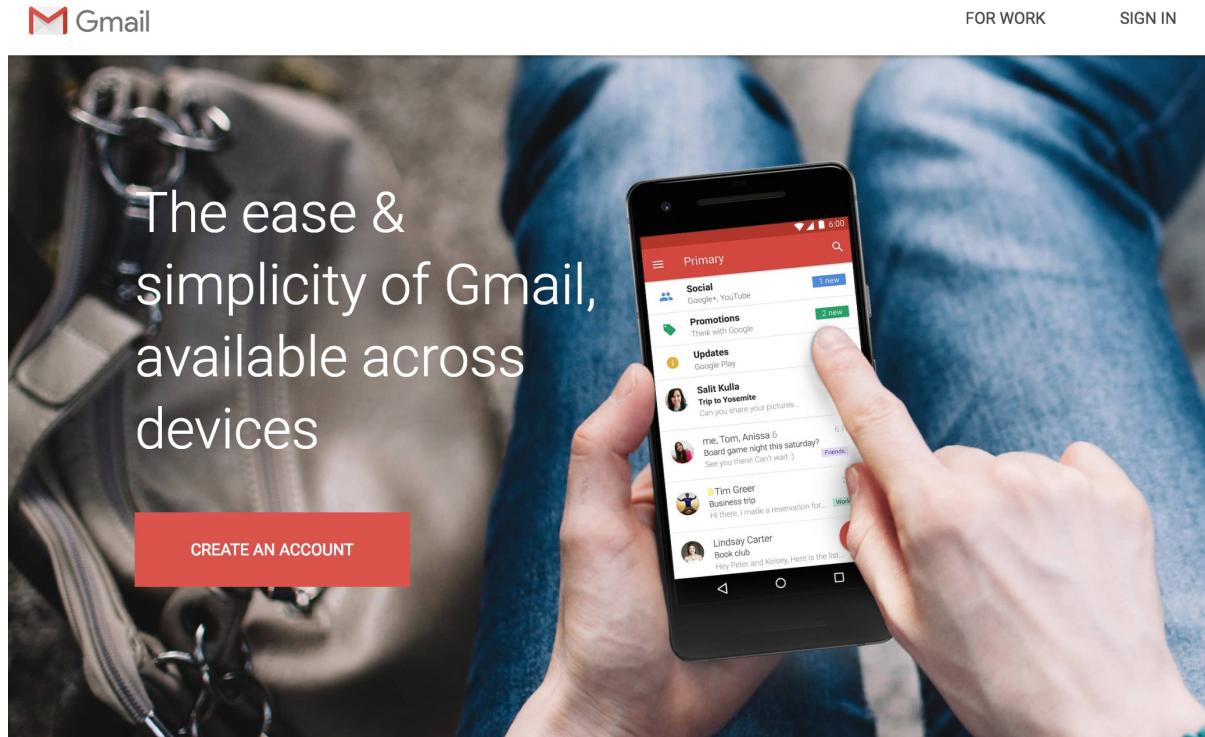
话不多说，接下来，我们开始配置 WordPress 自带的通过电子邮件发送的功能。

1. 准备邮箱

无论是使用自带的功能，还是使用第三方插件提供的功能，都需要准备一个邮箱来接收具体的文章，这就要求你准备好一个邮箱，用来接受邮件。



一般来说，我们可以使用第三方的邮箱来接受邮件，比如 163 邮箱、Gmail、QQ 邮箱等，这里你需要注意的是，建议使用一个全新的邮箱，并借助于邮箱系统自带的过滤器，来确保无关的邮件不会进入你的收件箱，从而确保你的博客内容不会收到干扰。此外，你可以使用自建的域名邮箱来作为接收文章的邮箱。



在选择邮箱时，需要考虑到国内特殊的网络环境，如果你的 WordPress 部署在国内的主机上，最好不要使用 Gmail 等在国内无法正常访问的邮箱，使用国内服务商提供的邮箱可以确保你的邮箱能够正常的登录和获取邮件。

在完成了邮箱的注册等操作后，注意要到邮箱的设置界面去设置开启 POP 服务，如果未开启该服务，则无法让 WordPress 登录你的邮箱来获取邮件和邮件内容了。

POP3/IMAP/SMTP/Exchange/CardDAV/CalDAV服务

开启服务：	POP3/SMTP服务 (如何使用 Foxmail 等软件收发邮件？)	已开启 关闭
	IMAP/SMTP服务 (什么是 IMAP，它又是如何设置？)	已开启 关闭
	Exchange服务 (什么是Exchange，它又是如何设置？)	已关闭 开启
	CardDAV/CalDAV服务 (什么是CardDAV/CalDAV，它又是如何设置？)	已关闭 开启
	(POP3/IMAP/SMTP/CardDAV/CalDAV服务均支持SSL连接。如何设置？)	

生成授权码

温馨提示：在第三方登录QQ邮箱，可能存在邮件泄露风险，甚至危害Apple ID安全，建议使用QQ邮箱手机版登录。

继续获取授权码登录第三方客户端邮箱 [?](#) **生成授权码**

此外，还需要注意的是，国内的一些邮箱采用了独立的授权码机制（比如 QQ 邮箱），在这种情况下，你需要将独立的授权码填写在你的 WordPress 后台的设置界面。

2. 设置 WordPress 后台的发布功能

在完成了邮箱的准备后，就可以进入 WordPress 后台，在 WordPress 后台的「设置」——「撰写设置」页面，找到「通过电子邮件发布」，并将邮箱服务提供商给你的邮件服务器等信息，填写在这里。这里需要注意的是，我们需要在这里指定默认的邮件发布分类目录。



设置完成后，单击“保存”按钮，接下来就可以试着使用邮件来发布内容了。

3. 测试内容的发布

我们打开自己的私人邮箱，在私人邮箱中，编写一封发向我们的收件邮箱的邮件，用来测试我们的测试内容的发布。



发布内容后，打开浏览器，[输入网址](#)，将 yourdomain 替换为你自己的域名），来触发 WordPress 去拉取邮件。



作者为no-reply@oschina.net

作者为no-reply@oschina.net

作者: 1

文章标题: 开源中国一周精彩回顾 ?? Angular

任务完成。信息1已删除。

作者为fang040705@111abj.com

作者: 1

文章标题: Newsletter notice to 894849635

任务完成。信息2已删除。

作者为privacy-noreply@policies.google.com

作者: 1

文章标题: 隐私权政策和隐私控制设置方面的诸多

任务完成。信息3已删除。

访问完成后，稍等2分钟，然后回到站点的首页、刷新，就可以看到我们通过邮件发布的内容了，再根据我们的需要编辑内容，发布即可。

□ 标题	作者	分类目录	标签	日期
□ my new email — 待发布	admin	未分类	—	— 最后修改 1分钟前
□ WordPress Test Email — 待发布	admin	未分类	—	— 最后修改 1分钟前
□ 邮件已被客户端成功 — 待发布	admin	未分类	—	— 最后修改 3分钟前
□ WordPress Test Email — 待发布	admin	未分类	—	— 最后修改 7分钟前

4. 设置自动拉取内容

在使用中我们遇见了一个问题，那就是内容无法自动拉取，需要访问特定的域名来触发，如何简化掉这一个流程呢？接下来提供两个思路。

通过用户访问来自动拉取内容

可以在网页的底部添加一些代码，从而当页面被请求加载时，能够自动去拉取文件，而不需要你手动去拉取，这样就降低了操作的难度，同时还可以借助访客来确保能拿到最新的文章。

通过 Crontab 来自动拉取内容

本质上来说，想要触发拉取文章，就是需要让 `wp-mail.php` 被请求，我们可以使用 Crontab 来实现自动的内容拉取。

我们可以在 Crontab 中使用 `wget` 命令来确保 `wp-mail.php` 文件被请求。

```
wget -N http://yourdomain/wp-mail.php
```

将其中的 `yourdomain` 替换为自己的域名，然后将其加入 Crontab 中来实现定期的自动访问。比如，你需要每 5 分钟能自动拉取内容，就可以添加如下内容（WordPress 默认限制每 5 分钟检查一次，如果需要更高的频次，则需要在 `wp-config.php` 文件中设置 `WP_MAIL_INTERVAL` 的值）。

```
*/5 * * * * wget -N http://yourdomain/wp-mail.php
```

这样就无需借助于用户的访问来进行内容的拉取，对于一些小的站点来说，能够更加清楚的确保新内容的获取。

如果你刷新的频次太高，就会看到这样的提示：



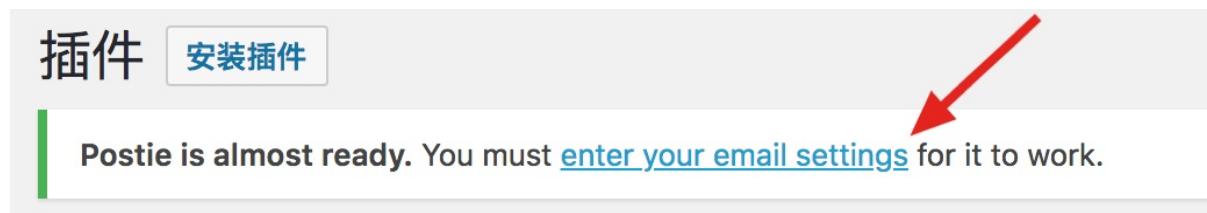
此外，还可以 使用各种云监控提供的定时访问功能来触发内容的拉取，这个和 crontab 的用法基本一致，可以参考各个平台的说明。

使用「Postie」插件发布文章

WordPress 自带的邮件发布功能不完善，无法设置自定义的邮件标题等，因此，推荐你使用 Postie 来实现邮件发布文章。

配置插件

在 WordPress 后台的插件管理界面搜索并按照 Postie 插件，安装完成后，点击提示，进入设置页面。



在设置页面中的「Mailserver」部分设置收件邮箱的地址（邮箱注册和配置的部分参考使用「通过电子邮件发布」功能发布文章中的第一小节，准备邮箱部分。）

Connection

Sockets 

Sockets is preferred, but doesn't work with some hosts.

Mail Protocol

POP3 

Port

110

Standard Ports:
POP3: 110
IMAP: 143
IMAP-SSL: 993
POP3-SSL: 995

Mail Server

pop.163.com

Mail Userid



Mail Password

.....

配置完成后，点击页面底部的「保存更改」，保存配置，然后点击页面右上角的「Test Config」。

在弹出的页面中搜索 `DISABLE_WP_CRON`，看看这一项的值是不是 Off，如果不是 Off，就要看看是不是在 `wp-config.php` 中关闭了插件运行所需的 `WP_Cron`。

然后拖动到底部，找到「Connect to Mail Host」一栏，可以看到「Successful POP3 connection on port 110」，则说明你的配置是正确的。

Connect to Mail Host

Postie connection: sockets

Postie protocol: pop3

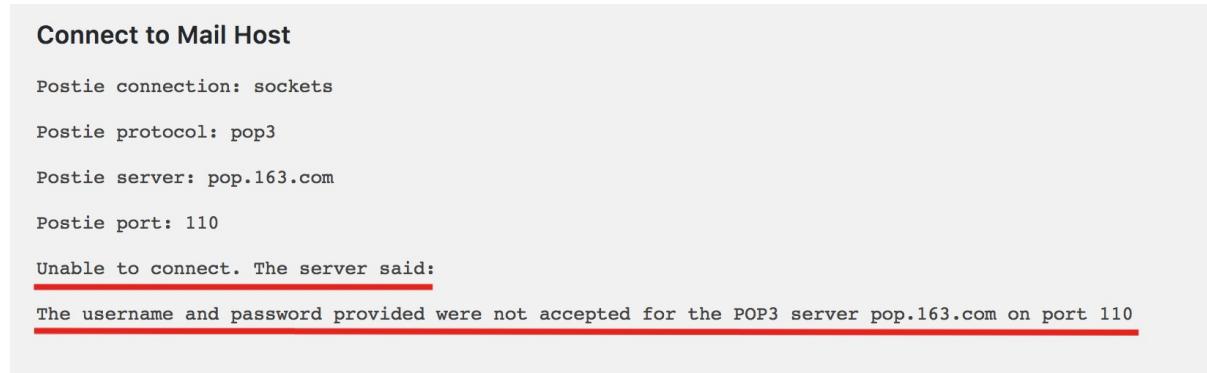
Postie server: pop.163.com

Postie port: 110

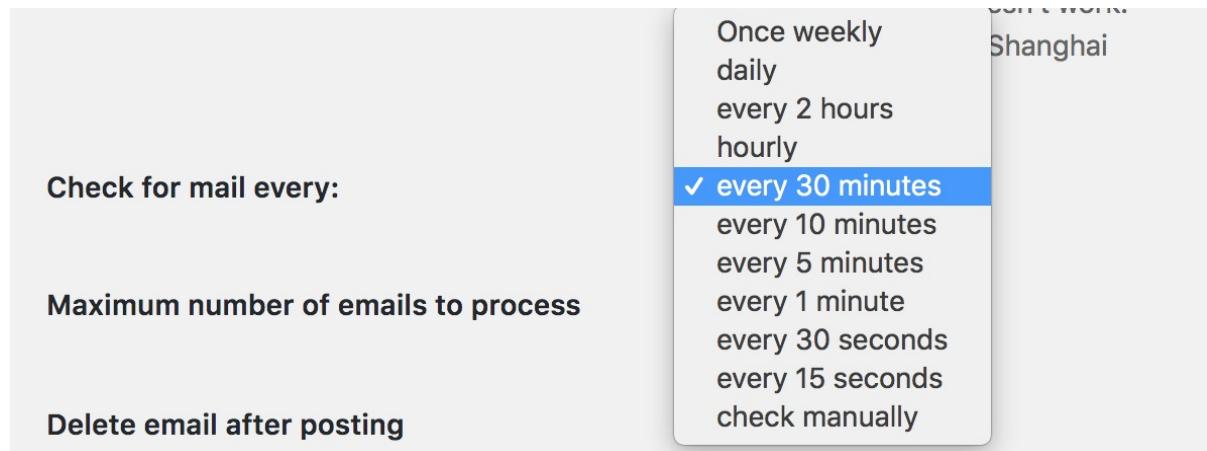
Successful POP3 connection on port 110

of waiting messages: 0

如果设置错误，则会提醒你报错，比如下面这样的，便是账号密码不匹配。



除了邮箱账号密码的设置以外，还有一个很重要的设置便是自动检查频次的设置，在下方的「Check for mail every」部分可以设置检查邮箱的频次，默认是 30 分钟一次，可以根据你的需要，调整检查的频次。



此外，建议你设置一下「Message」中的 **Preferred Text Type**，这个项目用于指定默认的邮件类型，默认值是 Text，但是国内的邮件系统大多都是默认 HTML，所以可以也设置为“默认 HTML”。

Message 页面下方的配置都是发文章时的一些默认配置，都非常简单，就不再赘述，如果你有需要，可以单独在读者圈进行提问。

配置完成后，保存配置。

发送邮件

在邮箱中新建一封邮件，并发送到我们的邮箱中，等待一个检查周期，就可以看到已经发布了邮件的内容。

全部 (7) 已发布 (5) 草稿 (1) 待审 (1) 回收站 (17)					搜索文章			
批量操作		应用	全部日期		所有分类目录	筛选	7个项目	
<input type="checkbox"/> 标题		作者	分类目录	标签	评论	日期		
<input type="checkbox"/>	test My WordPress	admin	未分类	—	—	已发布 1分钟前		

我们可以看到，WordPress 自动的识别出我们通过邮件设置的格式等。

A screenshot of a mobile keyboard interface. At the top, there is a toolbar with various icons: a double arrow for text direction, a magnifying glass, a text input field containing the text 'zhes舍 OK', a downward arrow, bold ('B'), italic ('I'), a horizontal line with three dots above it, a fraction icon (1 over 2 over 3), a double quotes icon, a horizontal line with three dots below it, a horizontal line with three dots to its left, a link icon, a grid icon, and a square icon.

自定义目录和标签

在通过邮件发布文章时，可以设置自定义的目录和 tag，这是 Postie 相比于原生功能的优越之处，而且，使用起来非常简单。

如果你需要设置自定义的目录，可以在邮件的标题中指定，使用 [] 或 - - 来包裹你的目录名即可（需要目录名存在），比如 [category1] 、 - category1 - 。

如果你需要设置自定义的标签，就在文章的正文中放置 `tags:`，然后放入 `tag`（不要求存在），不同的 `tag` 之间使用英文逗号隔开，比如：`tags: cats, funny`。

The screenshot shows the WordPress editor interface. At the top left, it says "主题 [category1] 测试自定义目录". Below the toolbar, there's a "正文" section containing the text "tags: cats, funny". The toolbar includes icons for adding attachments, large attachments, online documents, and other editing functions. A horizontal dashed line separates the editor from the post content.

tags: cats, funny

测试内容

白宦成

发布后的效果如下：

<input type="checkbox"/> 标题	作者	分类目录	标签		日期
<input type="checkbox"/> 测试自定义目录	admin	category1	cats、funny	—	已发布 1分钟前

至此，我们完成了使用 Email 来发布 WordPress 文章的内容，虽然 WordPress 官方提供了通过邮件发布的功能，但是我更加推荐你使用功能完善的「Postie」来完成通过 Email 发布文章的需求。

答疑 20171130

如何修改网站地址

源问题：老师 我在ftp上传好后登陆我备案好的网站 然后自动跳转到了数据库的地址 配置好后登陆我的网站还是在数据库的地址 我想配置好后地址栏里显示我的域名我这该怎样解决呢

你是需要修改网站域名。接下来，我跟你说明一下修改域名的操作。

1. 登陆 WordPress 的后台，进入到设置-常规中



2. 将**WordPress地址 (URL)** 和**站点地址 (URL)** 替换为你新的、备案后的域名。

The screenshot shows the '常规选项' (General Options) page in the WordPress settings. It includes fields for '站点标题' (Site Title) set to 'WordPress Dev', '副标题' (Tagline) set to '又一个WordPress站点', and descriptions for both. Below these are fields for 'WordPress地址 (URL)' and '站点地址 (URL)', both currently set to 'http://wordpress.dev'. A note below the address fields says, '如果您想让您的站点主页与WordPress安装目录不同, 请在此输入地址。' (If you want your site's homepage to be different from the WordPress installation directory, enter the address here.)

3. 保存并退出。刷新后，你再次访问你的网站就不会在登录时出现跳转到原有域名的问题了。

是否应该为图片开启 Gzip?

源问题： gzip压缩图片疑问，看到网上有些建议不要压缩图片，不知道如何才好。

你可以只压缩文本文件(html/css/js)

- 为什么要开启Gzip：因为我们需要通过压缩，来减少服务器传送的代码。
- 为什么不开启Gzip：压缩需要占用CPU计算时间，解压缩同样需要占用CPU计算时间。对于小文件(html/css/js，多为数K到数十K)来说，CPU的占用可以忽略，但是对于图片来说，需要花费大量的CPU计算时间来完成压缩和解压缩的工作，会导致用户体验变差。

有没有什么国外的主机商推荐？

源问题：想問一下,我剛付費,付費前見到用的 hosting 供應商。如我是不想放在這供應商,不知大師可有建議其他國外或香港的供應商？

国外主机我用的不多，不过这里推荐给你 [HawkHost](#)。HawkHost 成立于2004年，提供虚拟主机、VPS、reseller 服务。

The screenshot shows the HawkHost website homepage. At the top, there's a navigation bar with links for ALL PRODUCTS, HOSTING, DOMAINS, VPS, RESELLER, CONTACT, and SUPPORT AND CLIENT AREA. The main headline reads "Awesome Prices on World Class Hosting". Below it, a sub-headline says "Get supercharged hosting that scales with your business backed by the best 24/7 support team on the planet." A promotional banner on the right side features the text "SUPER SALE" in large white letters, "25% OFF ALL HOSTING PACKAGES" in green, and "ENDING SOON" in red. At the bottom left, there's a button labeled "Get Started Now!" and another labeled "Learn More". A small note at the bottom states "*Requires 2 year contract - Limited time offer - 30 days money back guarantee". At the very bottom, there's a banner with the Chinese flag and the text "DIRECT PEERING WITH CHINA FOR BLAZING FAST SPEEDS IN ASIA".

他们的服务器采用的是 IBM 的 softlayer 数据中心，不管是对大陆还是对海外，都非常不错。

此外，他们的虚拟主机不做任何限制完全放开让你用，具体可以看官方的 TOS。

为什么要从官网下载 WordPress

源问题：为什么要购买luxn ?不可以从其他网站中下载wordpress吗?

我们无法保证从其他地方下载的 WordPress 是未经修改的、安全的，所以建议大家从官网下载。

小白要怎么学这个课程呢？

源问题：老师你好，我刚看到这个，立马付费加入，但是对于程序，我是绝对小白，是不是需要从开篇开始读起，我看了一下读者圈，说的都不太懂

先看导读，然后跟着第一课开始看，一直看到第七课。第八课开始的内容对于完全没有基础的同学来说还是有难度的。你可以在掌握了前七课内容的基础上，再进行后续的学习。

需要将 XAMPP 换成 PHPStudy 吗？

源问题：老师好，我正在上您的wordpress课程，里面提到环境配置可以使用phpstudy，但我windows7电脑原本安装了XAMPP。我了解到phpstudy的优势是可以随意切换php版本，以及同时运行多个php版本，但这个功能在项目中会很重要吗？我需要把xampp换成phpStudy以更好学习您的课程吗？谢谢

不需要。PHPStudy 和 XAMPP 都是 Windows 下的集成环境，你可以根据自己的喜欢选择。有了 XAMPP，你已经可以学习后续的课程了。本节课程不会过多的涉及到 WordPress 之外的东西。

答疑 20171201

如何迁移 WordPress 的数据?

源问题：最後一問,我自己有一個免費版wp,內容是否可轉移至新開的付費網站?

WordPress 本身带有了非常好的数据导入导出工具，你可以直接使用。

导出

在工具 中选择导出,



在导出的页面选择全部内容，并下载导出的文件，会自动为你下载一个wordpress.xml文件。

导出

在您点击下面的按钮后，WordPress会创建一个XML文件，供您保存到计算机中。

我们称这种格式为WordPress eXtended RSS或WXR，它包含了您的全部文章、页面、评论、自定义栏目、分类目录和标签。

保存完下载的文件后，便可以在其它WordPress站点中使用“导入”功能进行内容导入。

选择导出的内容

所有内容

选择此项，则将包含您站点的所有文章、页面、评论、自定义字段、条目信息（分类和标签等）、导航菜单以及自定义文章。

文章

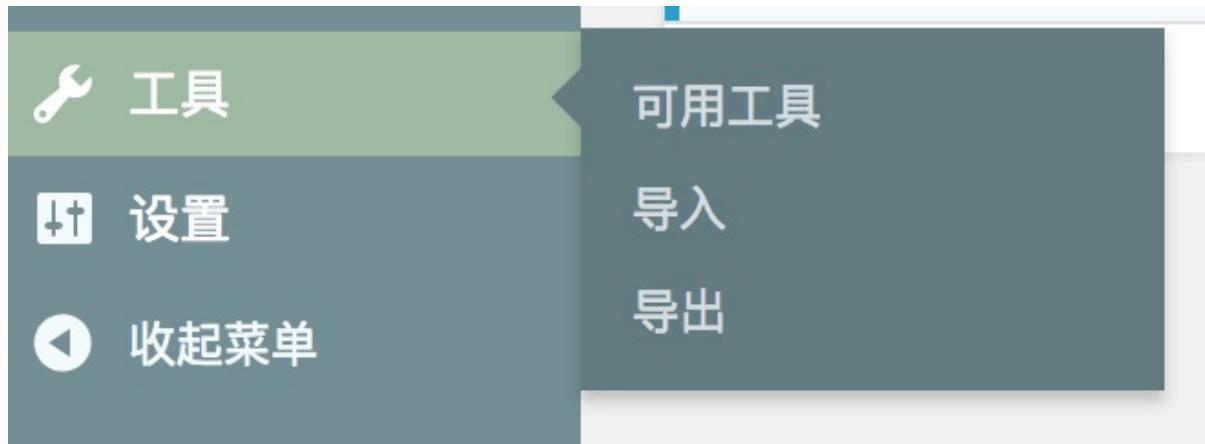
页面

媒体

[下载导出的文件](#)

导入

在工具中选择导入,



安装其中的WordPress 导入工具

导入

若有需要，您可以把其它系统的文章和评论内容导入到这个WordPress站点。请从以下系统中选择一个导入源，开始导入。

Blogger 从Blogger博客导入文章、评论和用户。

[现在安装 | 详细信息](#)

LiveJournal 从LiveJournal通过他们的API导入文章。

[现在安装 | 详细信息](#)

Movable Type和TypePad 从Movable Type或TypePad博客导入文章和评论。

[现在安装 | 详细信息](#)

RSS 从RSS feed导入文章。

[现在安装 | 详细信息](#)

Tumblr 从Tumblr通过他们的API导入文章和媒体。

[现在安装 | 详细信息](#)

WordPress 从WordPress导出文件导入文章、页面、自定义字段、分类和标签。

[现在安装 | 详细信息](#)

分类目录-标签转换器 选择性地将已有的分类目录转换为标签，或将标签转换为分类目录。

[现在安装 | 详细信息](#)

链接表 导入OPML格式的链接。

[现在安装 | 详细信息](#)

安装完成后，运行导入器；选择你要导入的文件，并上传即可。

导入 WordPress

您好！请选择您要导入的 WordPress eXtended RSS (WXR) 文件，我们将把其中的日志、页面、评论、自定义字段、分类及标签导入到您的站点中。

选择一个 WXR (.xml) 文件，然后点击上传进行导入。

从您的计算机上选择一个文件：(最大大小：128 MB) wordpressdev.wo....2017-12-01.xml

答疑 20171206

如何不通过第三方插件实现邮件发送功能

源问题：白老师，请问一下，如何实现不安装第三方插件用户注册成功后发邮件通知和找回密码发邮件通知功能？

这个问题，首先要说明下我们为什么要设置 SMTP。实际上，如果你使用的是诸如 cPanel、DirectAdmin 之类的虚拟主机面板，是不需要设置SMTP的，因为默认是支持通过 PHP 发送邮件的。

那为什么国内的主机大多不支持呢？这是由于如果你没有处理好网站的安全问题的话，很容易出现你的网站被别人拿去作为肉鸡，对外疯狂的发送垃圾邮件，从而导致最终服务器的IP被拉入黑名单，而国内的主机商为了避免这个问题，所以都封禁了主机的25端口（这个端口用于发送邮件）。如果你向服务商申请开启25端口，就可以通过 php 和 sendmail 来发送邮件，就无需安装第三方插件了。

WordPress 如何存储文章数据？

源问题：白老师，想请教一下，wordpree发布完文章后，文章内容是以网页形式保存还是存在数据库，如果以网页形式保留的话我们以后修改页面链接风格的话是不是又多了一个页面文件保存下来？这样系统容易膨胀。

不是保存为文章，WordPress 始终将文章内容保存在数据库的。除非你借助插件生成了静态页面。如果是缓存插件生成的静态页面，则无需担心，插件会根据策略自动删除的。

如何迁移 WordPress 站点？

源问题：白老师，请教一下，我在自己NAS配置好了WordPress网站了，请问该如何迁移到阿里云虚拟主机呢？我在NAS安装WP时的数据库地址、用户名、密码、数据库表的前缀跟阿里云虚拟主机不一致，谢谢！

这个问题分为两种情况：

1. 你迁移后不改域名
2. 你迁移后改为新的域名

如果是前者，操作比较简单，具体操作步骤如下

1. 使用 MySQL 数据管理工具导出数据，比如 phpMyAdmin，或者可以执行命令来导出数据库数据 `mysqldump -u<userName> -p <dbName> >> <fileName>.sql`。替换其中的 `<userName>` 等信息为自己的。比如 `mysqldump -uroot -p wordpress >> wordpress.sql`。
2. 将网站文件打包生成压缩包。如果你的网站文件已经上传在了别人虚拟主机，你可以通过 FTP 将文件下载下来，保存到本地。
3. 在新的虚拟主机中，使用数据库管理工具导入刚刚生成的 SQL 数据文件。

4. 将文件通过 FTP 上传到新的虚拟主机。并按照下方截图，修改 `wp-config.php` 文件中的对应配置为你的新的虚拟主机配置。具体的修改你可以参考：

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');      数据库名

/** MySQL database username */
define('DB_USER', 'root');          数据库用户名

/** MySQL database password */
define('DB_PASSWORD', 'f*****');    数据库密码

/** MySQL hostname */
define('DB_HOST', 'localhost');     数据库主机地址

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');

```

如果是后者，你需要在搬迁前，先修改网站地址。

1. 参考[如何修改网站地址](#)，修改你的网站地址。
2. 使用 MySQL 数据管理工具导出数据，比如 phpMyAdmin，或者可以执行命令来导出数据库数据 `mysqldump -u<userName> -p <dbName> >> <fileName>.sql`。替换其中的 `<userName>` 等信息为自己的。比如 `mysqldump -uroot -p wordpress >> wordpress.sql`。
3. 将网站文件打包生成压缩包。如果你的网站文件已经上传在了别人虚拟主机，你可以通过 FTP 将文件下载下来，保存到本地。
4. 在新的虚拟主机中，使用数据库管理工具导入刚刚生成的 SQL 数据文件。
5. 将文件通过 FTP 上传到新的虚拟主机。并按照下方截图，修改 `wp-config.php` 文件中的对应配置为你的新的虚拟主机配置。具体的修改你可以参考：

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');      数据库名

/** MySQL database username */
define('DB_USER', 'root');          数据库用户名

/** MySQL database password */
define('DB_PASSWORD', 'f*****');    数据库密码

/** MySQL hostname */
define('DB_HOST', 'localhost');     数据库主机地址

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');

```

答疑 20171216

安装完成插件后报错怎么办？

源问题：白老师，我成功安装一个插件后，出现这个提示Warning: sprintf() [function.strftime]: Too few arguments
请问是什么问题？

一般来说，官方的 WordPress 核心不会轻易报错。如果你在安装了一个插件/主题后，出现了报错，那么大概率来说，是这个插件/主题本身的代码不够规范，导致出现了错误。如何判断具体是哪个插件呢？你可以先停用你刚刚安装的插件，看看是否还在报错。如果停用后，不再报错，则说明是这个插件的问题。

在你确定了问题后，就可以到 WordPress 官方的插件页面，去提交问题了。

如何选择源码存放目录？

源问题：源代码应该放到后台哪个位置？

一般来，cPanel、DirectAdmin 这类的面板，都有一个 `public_html` 文件夹，你可以将你的 WordPress 源码放在 `public_html` 中，如果你需要安装在子目录(`xxx.com/blog`)，则可以在 `public_html` 目录下创建新的目录，来放 WordPress 的源码。

WordPress 的主题/插件可以不受版权限制的使用呢？

白老师，请教一下，收费的主题或插件是不是都有申请过专利或其他版权登记之类的？不然我们是不是可以稍作修改就可以不受版权限制来使用呢？

软件著作权的登记是一个费时费力的事情，所以大部分的主题作者不会考虑去做相关的登记。而且，「著作权」本身是在这个主题开发完成后，就天然存在的，登记仅仅是去做一个公证，所以大部分开发者都不会选择去做著作权的登记。

不登记不意味着我们可以无限制的使用，我们拿 Themeforest 来举例。Themeforest 的每个主题都有两个不同的授权：Regular License 和 Extend License。我们大部分时候都是购买 Regular 授权，这个授权按要求，仅能用于「你自己」的「一个」站点，无法用于给客户，如果你要给客户安装，需要另外购买一份 Regular License 或者 Extend License。但是事实上，由于版权意识的淡薄和代码中不加入相关的限制，你自己使用也没有问题。你在客户的站点上使用，开发者也没有办法追责。

Regular License ▾

\$ **49**

Regular License **SELECTED**

\$ **49**

Use, by you or one client, in a single end product which end users **are not** charged for. The total price includes the item price and a buyer fee.

Extended License

\$ **2550**

Use, by you or one client, in a single end product which end users **can be** charged for. The total price includes the item price and a buyer fee.

[View license details](#)

由于国内对版权的淡漠，国内的不少开发者都采用了按域名授权的模式来售卖主题，一份主题的授权只能绑定1~2个域名，如果你需要在更多的域名上使用，需要另外购买。

如果安装 WordPress 时，手抖误操作了，把文件覆盖了怎么办？

源问题：我是准备在绑定新域名安装程序！结果你说把路径设置一下！我也跟者做！结果把我原来站点内容覆盖了！现在如何解决吗？

金无足赤，人无完人。我们每个人都有可能遇见误操作的情况。如果你真的出现了误操作的情况，也不用着急，有办法找回。

这里的找回仅限于「覆盖」

在默认覆盖的情况下，我们的文件夹是不会被删除的，所以你首先要做的是将网站根目录下的 **wp-content** 文件夹下载到本地进行备份，以免丢失附件。然后，到数据库管理工具（如 phpMyAdmin）中导出你的数据，以防不测。

在备份完成后，你可以删除网站根目录下所有的文件，重新上传 WordPress 的源码，进行安装。在安装完成后，上传你的 **wp-content** 目录，并使用数据库管理工具导入你刚刚备份的数据，来恢复站点的访问。

事实上，整个网站目录，只有 WP-Content 目录是和我们密切相关的，其他目录的文件都是 WP 官方提供的，每次的版本更新也只会更新那些文件，而不会更新 WP-Content 目录。所以，经常对 WP-Content 目录进行备份、经常导出自己的数据库备份，还是很有必要的。

如何判断一个主题/插件中是否包含恶意代码？

源问题：白老师，如何才能找出主题或插件里面的恶意代码？是不是可以安装 Sandboxie 软件测试？

Sandboxie 是针对二进制文件的沙箱，并不会针对 WordPress 起作用。想要测试 WordPress 主题或插件，你需要安装一个空白（不安装任何第三方主题/插件）的 WordPress，然后安装你要检测的主题/插件，看你的网站能否正常运转。

具体到操作层面，我的建议是你在空白 WordPress 下使用 git 进行版本控制，安装插件前，进行一次 commit，然后安装插件，运行一段时间后，使用 `git status` 命令查看你的文件的变动情况，如果基本没什么变动（事实上应该是完全没有变动，因为这里要求你不能安装缓存插件），则这个主题/插件大概率上是安全的。

最稳妥的方法还是自己对代码进行一遍 review。

免费主题是否能够来做商业网站？

源问题：白老师，请问免费主题修改后能否用来做单位网站？会不会涉及侵权问题？如何才能知道主题是否安全？

免费主题能不能做单位网站你要看开发者的授权。如果开发者允许，自无不可。

此外，由于 WordPress 本身采用具有传染性的 GPL 协议进行开源，所以理论上所有的 WordPress 插件、主题都要以 GPL 进行开源。如果你真的要使用，要注意一下。

有没有什么日历插件比较推荐？

源问题 我想在网站上放置一个日历，不是文章发表的记录，而是用来发布活动的，可以自定义时间、事件。请问有没有这类插件？

这方面我推荐你使用 The Events Calendar，官方提供插件的中文，很方便。

使用 Filezilla 上传 WordPress 解压包有文件传输失败怎么办？

源问题：使用 Filezilla 上传 WordPress 解压包有文件传输失败怎么办？

你可以重新上传，然后跳过那些已经上传了的文件。也可以上传压缩包，使用阿里云虚拟主机后台的解压工具进行解压。

