切换导航 Farseerfc的小窝

文

繁體

<u>简体</u>

English

日本語

- <u>▲ About</u>
- <u> Links</u>
- ≜ About
- <u>Links</u>
- □ Import
- □ Life
- <u>□ Tech</u>

- <u>Import</u>
- <u>Life</u>
- <u>Tech</u>
- Q

搜索

- 搜索
- 9

前言: 新天新地,将一切都更新了 Bootstrap 3 的新设计 Pelican 3.5 的新功能 新的文件夹布局 主题: Material Design 风格的 Bootstrap 3 对 Bootstrap 3 的定制 响应式设备的大小 根据宽度自动分栏和瀑布式布局 正文的样式 一些细微的定制 对 bootstrap-material-design 的定制 将以上两者整合在 pelican-bootstrap3 里 插件: 发挥 Pelican 和 reStructuredText 的优 势 i18n-subsites plantuml render-math youtube 和 youku tipue_search 0 neighbors 和 series 0 bootstrapify 和 twitter_bootstrap_rst_directives

- extract_toc 和 summary
- 结语

2015年2月14日更新

前言: 新天新地,将一切都更新了

[1]

不知不觉间放任这边长草很久了,从上次 折腾主题到现在都快三年了, 而从上次 写了篇告白信 到现在也有快两年了。 这期间曾经把主题配色从Bootstrap 2 默认的 白底黑字改成了让眼睛更舒适的黑底白字,也不过是用 drop-in 的配色方案而已,没有本质上的改进。

洞中一日世上千载,两年里 Bootstrap 已经升上 v3.3,而 Pelican 则已经升到 3.5 了。 早就眼馋 Bootstrap 和 Pelican 中的诸多新功能新设计,不 过无奈于时间有限只能饱饱眼福。

近日想写的东西越积越多,终于下定决心花了前前后 **两个月** 的时间重新设计了一遍 Pelican 的主题,配合一些我觉得有用的插件。于是本博客就变

成你们现在看到的样子了。 (以及本篇博文也用了两个月的时间写完,其间还发了几篇别的短文,算是恢复写博客的尝试吧。)

在迈阿密参加 ICSR 2015 的时候 拍到的街边一家叫 Pelican 的旅馆



Bootstrap 3 的新设计

mobile-first

responsive

 全新的 优先移动设备 响应式 设计。 原本 Bootstrap 2虽然有响应式设计, 不过诸多细 节不能符合我的需求,最终还是得手工 hack

@media 查询去微调。 现在的 优先移动设备

responsive grid system

响应式 栅格系统 则相对显得科学很多了,也终于能在手持 设备上看起来舒服一些。诸位可以尝试改变窗口宽度,或者在不同的手持设备上打开这个 blog,体验一下这个页面在不同显示器大小中的效果。如果仍有问题欢迎 发 lssue 给我。

Navbar

科学的 导航栏。 比 Bootstrap 2 那个科学很 sticky
 多了。无论是 保持 在上端还是跟着浮动, 或者像这边这样 自动隐藏 都很简单。

更多细节参考 Bootstrap 3 主页。

Pelican 3.5 的新功能

- Python 2 和 Python 3 统一代码: 再没有恼 人的 unicode 相关的问题了。这对 blog 系统 来说相当重要啊。 而且还能方便切换 pypy 等 不同的解释器。
- 全新的插件系统:非常多功能强大的插件等 着你。
- 增强了导入系统:嗯总算可以导入我的中文的 wordpress 博客了。(虽然那边长草更久

了……)

hard code

● 站内链接:不用 硬编码 目标页面的链接了,可以直接写源文件的位置然后让 pelican 处 plugin theme 理,这样能简化各种 插件 和 主题 的实现。

更多细节参考 Pelican 文档。

新的文件夹布局

Pelican 的新文件夹布局



之前的博客 仍然留在 github 上,其中的内容完全搬过来了。开始写老博客的时候 Pelican 版本较

早,没有形成好的 文件夹布局,导致生成的文章、使用的模板和撰写的内容全都混在一起,非常难以管理, 于是趁改版之际用了新的文件夹布局方式,并分为 4 个 git repo 分别管理历史。

首先是存放 总的博客内容的 repo , 其布局是如图那样的。这样将生成的静态网站和生成网站用的配置啦内容啦分开之后,顿时清晰了很多。

然后这个内容 repo 中的三个符号链接分别指向三个子 repo(没用 git submodule 管理纯粹是因为偷懒)。 theme 指向 pelican-bootstrap3 ,是我修改过的 pelican 主题。 plugins 指向 pelican-plugins ,由于 plugins 的质量有些参差不齐,其中不少 plugin 都按我的需要做了些许修改,一些是功能改进,另一些则是修bug(比如不少plugin只支持 python 2)。 最后 output 指向farseerfc.github.io 也就是发布的静态网站啦。

接下来从 主题 和 插件 两个方面介绍一下改版的细节。

主题: Material Design 风格的

Bootstrap 3

上篇 博文 就总结了我为了这个博客寻找了一堆 CSS框架,并且最终决定用 bootstrap-material-design, DandyDev/pelican-bootstrap3 和 Bootstrap3 这三个项目结合的方式实现这个模板的主题。 这三个项目都或多或少经过了我的修改,修改后的项目以 pelican-bootstrap3 为基础放在 这里,包括Bootstrap3 样式和 Material 样式。

对 Bootstrap 3 的定制

由于架构完善,修改 Bootstrap 3 感觉非常简单。 另一方面我在 Web 前端技术上的技能点也不多, 所以修改的地方非常有限,只能按我自己的需求定 制而已。

响应式设备的大小

修改了 Bootstrap 3 响应式设备的大小

```
1 @screen-xs: 320px;
2 @screen-sm: 598px; /* 768px; */
3 @screen-md: 952px; /* 992px; */
4 @screen-lg: 1350px; /* 1200px; */
```

```
5 @screen-xl: 2030px;
6 @container-sm: 582px; /* 750px; */
7 @container-md: 930px; /* 970px; */
8 @container-lg: 1320px; /* 1170px; */
9 @container-xl: 1990px;
```

首先把 Bootstrap 3 默认适配的几个 响应式设备的大小 改成了我需要的大小。 xs 和 sm 的大小分别按照我的手机屏幕 **竖屏** 和 **横屏** 时候的浏览器页面宽度来算, md 是想兼容 Nexus 7 横屏 960 的宽度以及 一个常见上网本 1024 的宽度。 lg 的大小则按照常见的笔记本 1366 宽的屏幕来适配。

这里 Bootstrap 3 支持的设备大小的一个问题是,它最多考虑到 1200 像素宽的显示器,而更宽的 比如 1600、 2048 甚至 2560 像素宽的显示器现在也并不少见,其结果就是页面中左右两侧 有很大的空间被浪费掉了。作为深受这一问题困扰的用户之一,我用 这里介绍的方法 给 bootstrap 增加了一类

bigger than bigger

「比大更大」的 xī 响应式设备尺寸,宽度设为支持 2048 像素宽的显示器,具体的修改反映在 variables.less 文件里。

根据宽度自动分栏和瀑布式布局

接下来目标是让主页的文章列表像 Google+ 主页那样根据显示器宽度自动调整分栏,使得宽度不同的显示器上每个分栏的宽度接近。想要达到的效果是,根据上面定义的屏幕宽度尺寸:



						-V		
	导航栏				│ 导航栏 			
文 	ζ	文	侧		文	文章	文	侧
	重	文章	边		章	章	文章	边
$\ \ $ 1		3	栏		1	3	5	栏
								1
	ζ	文	侧		文章	文章	文	侧
	重	文章	边		章	章	文章	边
2		4	栏		2	4	6	栏
			2					2
footer				footer				
底栏				底栏				

一开始纯粹用 Bootstrap3 的响应式栅格实现这个分栏布局,结果发现效果不太理想, 因为文章列表和侧边栏的高度是变化的,会导致栅格间留下大片空白。后来改用 这里示范的纯CSS瀑布式布局 实现文章和侧边栏的布局,具体的实现代码在waterfall.less ,总算达到了想要的布局了。

正文的样式

最最重要的是文章正文的样式。这里我想要达到的效果是,在大屏幕上用更大的字号,让读者看起来更舒适,同时在小屏幕上用比较小的字号,最终保证基本上「一行」的文字数接近。这个修改主要针对 . jumbot ron ,用了不太科学的方式代码太长就不贴全了。

一些细微的定制

把主题配色改成了现在这样的淡紫色 @brand-primary: darken(#6B5594, 6.5%); ,配合我的头像风格,这个修改只需要一行。接着删掉了 .btn 的white-space: nowrap; 让按钮的文字可以换行,这也只是一行修改。

2015年1月29日更新

另外我也不太喜欢 Bootstrap 3 默认在手机上的 collapsed navbar 折叠导航栏 ,折叠之后的操作不够直观方便而且依赖 javascript 所以有 bug …… 于是我把它关掉了,具体方式是在 variables.less 把 @grid-float-breakpoint 和 @grid-float-breakpoint-max 都设为 0就可以了。

对 bootstrap-material-design 的定制

这里定制的地方不多。原样式中一个不太科学的做法是所有 .btn 都强制加上了阴影 效果,这在已经有阴影的环境里用的话非常碍眼,像是 Win9x 风格的厚重睫毛膏。既然可以单独 给每个样式加阴影,于是就把 .btn 强制的阴影去掉了,只保留鼠标悬停之后强调的阴影。

其它定制的细节么就是统一配色风格,修补漏洞错 误,微调响应式效果而已,这里不细说。

将以上两者整合在 pelican-bootstrap3 里

Pelican 实现显示源代码按钮

显示源代码按钮借用了 Pelican 配置中自带的 OUTPUT SOURCES 选项将源文件复制到输出文件夹:

- 1 OUTPUT SOURCES = True
- 2 OUTPUT_SOURCES_EXTENSION = '.rst'

然后在 Makefile 里用 pygmentize 把所有源代码文件着色:

最后在按钮按下的时候用 jQuery 载入源代码:

1 <a onclick="\$.get('{{SITEURL}}}/{{article.s</pre>

虽然难看的 hack 比较多,但是能用!

虽说 pelican-bootstrap3 是我 fork 出来的,不过由于我修改的地方实在太多,代码看来基本上接近重写了一份。好在之前有给 pelican 写 bootstrap 2 主题的经验,这次修改算得上驾轻就熟。 可以对比一下上游作者的博客 和这里的样子体会一下感觉。具体修改过的地方包括:

- 1. 套用 bootstrap-material-design 的各个元素样式。
- 2. 在文章列表模板应用上面提到的 Bootstrap 3 的栅格布局和瀑布式布局。
- 3. 翻译到多个语言,这里在后面的 i18n-subsite 插件里详述。
- 4. 套用后面会介绍到的各种插件。
- 5. 统一侧边栏的样式到一个模板里。
- 6. 添加 Atom 订阅按钮和 breadcrumb 条。
- 7. 对正文中出现的插图,添加点击放大的功能, 通过 Bootstrap 的 modal 实现。

- 8. 上面提到的用 这个bootstrap插件 让导航栏自动隐藏。
- 9. 显示源代码按钮 ,也就是每篇文章信息栏中



插件: 发挥 Pelican 和 reStructuredText 的优势

先列举一下我目前用到的所有插件:

```
PLUGINS =
               ["i18n subsites",
                "plantuml",
                "youku",
                "youtube",
                 'tipue search',
                 'neighbors',
                 'series',
                 'bootstrapify',
                 'twitter bootstrap rst direc<mark>ti</mark>
                 "render math",
10
                 'extract toc',
11
                 'summary']
12
```

嗯其实不算多。接下来逐一介绍一下这些各具特色 的插件。

i18n-subsites

internationalization subsite

这个插件的目的是创建 国际化 子站。

之前介绍 Pelican 配置的时候就提到过, 原本的 Pelican 就支持一篇文章用多种语言书写,有 Lang 属性注明这篇文章使用的语言,以及 slug 属性注明 多语言的翻译之间的关联,换句话说同一篇文章的 多个语言 版本应该有相同的 slug 和不同的 lang 。 然后原本 Pelican 里对多语言的 实现方式是,首先 有一个 主语言 是模板和大部分文章采用的语言,文 章列表中会优先列出 用 **主语言** 撰写的文章,然后 从 主语言 的文章链接到别的翻译版本。 很多博客 系统和CMS对多语言的支持都是这样的,这种处理 方式的缺点也显而易见:作为 主语言 的语言必须足 够通用,才能让进来的人找到合适的翻译版本,所 以诵常 主语言 都是英语。

而这个插件做的事情描述起来很简单:将文章按语言属性分到多个子站,每个子站独立放在各自的文件夹。 比如主站是 https://farseerfc.github.io/的话,那么英语的子站就可以是https://farseerfc.github.io/en/。 然后分别对多个

子站生成静态页面。具体的实现方式是对 pelican 的页面生成步骤做了拆分:

- 1. pelican 按正常情况读入文章,生成元信息。
- 2. i18n-subsites 针对每个语言,覆盖掉 pelican 的一些选项设置比如路径和 URL , 分别调用 pelican 的页面生成器按模板生成文章。
- 3. 对共用的静态内容比如模板的 js 和 css 文件, 只在主站中生成,子站中的相应链接全部链回 主站。

虽然描述起来简单,但是这个插件可以说最大化利用了 Pelican 的插件系统,实现细节相对比较 复杂,大概是我用的这些插件里面最复杂的了。不夸张的说 Pelican 3.4 支持的新插件 API 和 站内链接功能基本上就是为了配合这个插件的。至于具体它会覆盖哪些 Pelican 的配置,请参阅它的README.md文件。

按内容拆分多语言子站的做法只解决了问题的一半,还留下另一半的问题,也即对模板的翻译。 对这个问题, i18n-subsites 提供了两套方案供选择:

1. 用覆盖配置路径的方式让每个子站套用不同的

模板。这配置起来简单,但是对模板维护起来 有点困难。

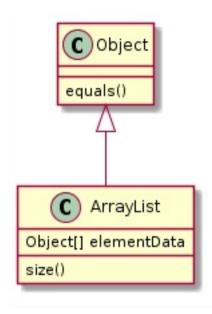
2. 用 jinja2 的 i18n 插件,配合 Python 的 gettext 库实现内容翻译。这个方案 配置起来 比较复杂 ,但是配置好之后用起来就很方便 了。 只是要记得每次修改了模板都要更新翻 译,处理 *.po 和 *.mo 文件等等琐碎事宜。

这里我用 jinja2 的 i18n 插件的方式实现了模板的翻译,各个语言的翻译在这里 , 然后用 这里的 SCons 脚本 根据内容是否变化自动更新 po 和 mo 文件。

配置好这一套方案之后,还要注意在模板和文章中处理好链接。用 Pelican 3.4 之后推荐的 新的文章间链接的写法以及将 SITEURL 设置为实际 URL 并且关闭 RELATIVE_URLS 之后,应该就不会出没什么问题了(可能还要考虑使用的模板和插件的兼容性,大部分都是写死了 URL 的问题)。

plantuml

嵌入 PlantUML 的示例



PlantUML 是一个Java实现的, 用接近文字描述的语言绘制 UML 图或者 GUI 界面图的工具,非常适合嵌入在 Markdown、 reStructuredText、AsciiDoc 等这种轻量级标记语言里。 然后么这个plantuml 插件就是定义了一个新的

reStructuredText 指示符 .. uml:: ,把嵌入的内容提取出来调用 plantuml 命令处理 成图像然后再插入到文章中。

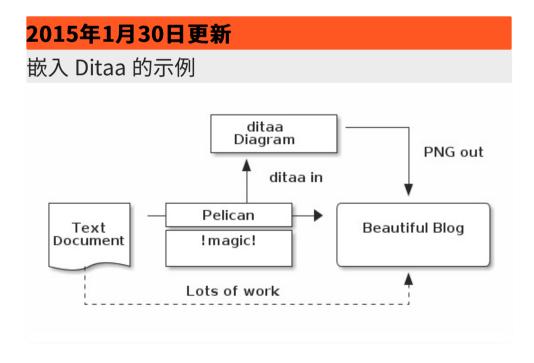
directive

比如示例里的这个 UML 图就是用这样一段简单的文字描述生成的:

```
1 .. uml::
```

```
3    Object <|-- ArrayList
4
5    Object : equals()
6    ArrayList : Object[] elementData
7    ArrayList : size()</pre>
```

实际用起来这个插件实现上稍微有点小问题:首先它只支持 python2,所以我把它改写成了 python 2 和 3 都通用的语法;其次它原本输出的文件夹似乎会被 pelican 删掉,所以把它改了个位置; 然后它输出的 URL 也和 i18n-subsites 插件间有不兼容的问题,也顺带修掉了。 修改之后的代码在这里。



plantuml 是绘制UML的,除此之外还有一个类似的

工具是绘制一般的 流程图 的,叫 ditaa ,和 plantuml 非常像,也比较像 reStructuredText 的表格。 于是我也照猫画虎实现了一个 ditaa 的

指示符 , 用起来类似这样:

```
.. ditaa::
                                  ditaa
                                 Diagram
                                    ditaa in
10
                             Pelican
11
12
             Text
13
          Document|
                           !magic!
14
                {d}|
15
16
17
                          Lots of work
18
```

render-math

嵌入公式的示例

示范行内公式 \(A_\text{c} = (\pi/4) d^2\).

整行公式

\begin{equation*} \alpha{}_t(i) = P(O_1, O_2, \ldots O_t, q_t = S_i \lambda{}) \end{equation*}

这个插件提供在 reStructuredText 中用 LaTeX 语

法插入数学公式的能力,定义了 :math: 行内角色 和

.. math:: 指示符。 实际工作的渲染库当然是大名 鼎鼎的 MathJax ,这个插件 会用 MathJax 的 CDN 载入,所以也没有额外的依赖文件。(只是不知道 是否会被国内墙掉, 如果公式显示不正常请 **务必** 告诉我。)

youtube 和 youku

directive

顾名思义,这两个插件分别实现嵌入 youtube 和 youku 视频。其中 youtube 是原本就有的插件, youku 是我照猫画虎抄的。 之前写了一篇 KDE5 Plasma 之跳动卖萌的活动按钮 用到了这两个插件。

tipue_search

Tipue search 是一个非常有意思也很强大的搜索工具,通过 jQuery 实现静态博客的站内搜索功能。实现方式是,它需要你写一个 json 文件,包含 整个网站的 全部 文章的标题和文字内容,然后在搜索的时候读入这个 json 做搜索(是不是有点耍赖)。虽然听起来会有性能问题,但是应用在小型的静态博客上效果意外很不错,比如本站的所有文章内容 放在一起的 json 也只有 300KiB 左右。

这个插件就是自动在 pelican 输出完全部静态网页 之后,调用 beautifulsoup4 从所有网页中抽取出 纯文本,产生这个 json 给 Tipue 用。

neighbors 和 series

这两个插件比较类似也都比较简单, neighbors 提供一篇文章的前后文章信息, 在主题模板里可以用来制作 **上一篇** 和 **下一篇** 按钮。 series 提供将多篇文章 且类为一个 **系列** 的支持,当然也需要在 主题模板中定义显示「文章系列」的列表。这两个插件的效果都能在本文末尾,评论区上方的部分看到。

bootstrapify 和 twitter_bootstrap_rst_directives

这两个插件让文章的 **正文** 套用上 Bootstrap 的样式。

bootstrapify 这个插件实现得比较简单,用beautifulsoup4 在静态网页的结果里面过滤元素,对 table, img, embed, if rame, video, object 这几个标签套用上 响应式嵌入对象的类 让他们更美观。

twitter_bootstrap_rst_directives 这个插件则是增

加了几个 reStructuredText 的 行内角色 和 指示符

。 它实现的 行内角色 包括: 用 :kbd: 实现如 Ctrl+C 这样的键盘快捷键, 用 :code: 嵌入代码片

段,用 :glyph: 嵌入字符图标。 它实现的 指示符 包括: labels 行内标签 , alerts 提示段落 , panels 嵌入面板 , 以及还有一个 media 混排图标 。

对其中的 panel 我改写了它在文章正文中的样式,在 lg 或者 xl 的屏幕宽度下,分别用 \(\frac{1}{2}\)和 \(\frac{1}{3}\) 大小的嵌入面板, 简单实现和正文

文字的图文混排。

除此以外我还在 twitter_bootstrap_rst_directives 这个插件里套用它的框架实现了两个额外 的

role

行内角色 , 分别是 : ruby: :通过 html5 的 <ruby>标签实现文字上方的注音(firefox下 不支持 ,会使用文字后的括号显示), 以及 : html: :在 行内插

raw

入 裸 html 标签(这属于 Markdown 的基本功能, 在 reStructuredText 这边由于要考虑多种输出格式

于是就比较麻烦了)。这两个 行内角色 的 实现代码 在这里 。

2015年2月3日更新

今天又在 twitter_bootstrap_rst_directives 里增加

了两个行内角色。 一个是 : twi: 用来写 twitter 用户的链接,比如 @farseerfc ,另一个是 : irc: 用来指向 freenode 的 channel ,比如 #yssyd3。

2015年2月14日更新

今天增加了.. friend::用来写好友链接,以及

fref 用来引用好友, 比如 LQYMGT 这样。

extract_toc 和 summary

最后是这两个有点「名不副实」的插件。

toc

reStructuredText 原本就有自动生成 目录 的功能,用起来也非常简单,只需要在想要插入目录的地方写一行 .. contents::,剩下的都由 docutils 自动生成了。 只是当然这样生成的目录肯定会插入在文章的正文里,而 extract_toc 这个插件的作用就是简单地 把这个目录抽取出来,让模板能在别的地方放置这个目录。比如我这里就把目录放在了一个 panel 里。

summar

然后 Pelican 也原本就有从文章中抽取 总结 显示在 文章列表的功能。 Pelican 原始的实现似乎是按照 文字数抽取前半段,不总是适合作为总结。 于是这 个 summary 插件的作用其实是允许在正文中以特 殊的注释的方式标注哪些部分应该被抽出来作为总 结。 summary 这个插件原本的实现只允许抽取一 段文字,我又对它的实现做了少许扩充,允许标注 多段 文字合并起来作为总结。

2015年1月29日更新

今天在 extract_toc 插件的帮助下,在侧边栏里放了一个 Bootstrap affix 的目录, 它保持在页面的右侧位置不变,方便导航到文章的各个地方。具体实现方法除了 Bootstrap 3 的 Affix 文档 ,还参考了这篇更详细的说明。

结语

这个博客的配置都可以在 github 上找到 ,包括用来 自动生成整个博客的 Makefile ,由于比较长,这里就不再贴了。

折腾这个主题前后历时两个月,期间学会了不少东西,也算是不错的收获吧。 现在既然基础打好了,接下来就要开始多写博客了。(希望拖延症不会再犯……)

最近发现除了我的博客之外还有一个网站 Kansas Linux Fest fork 了我的主题,不过他们用了我修改的早期版本,还是原本的 Bootstrap 3 和 bootstrap-material-design 样式。自己草草修改的东西被别人用到果然还是有点小激动呢, 以及接下

来不能马马虎虎地写 commit 消息了。

