
目录

前言	1.1
VSCode简介	1.2
界面布局	1.3
常用快捷键	1.4
代码编辑器常用功能	1.5
打开和新建项目	1.5.1
代码格式化	1.5.2
语法高亮	1.5.3
文件编码	1.5.4
搜索	1.5.5
普通搜索	1.5.5.1
正则搜索	1.5.5.2
查找函数定义	1.5.6
VSCode的智能好用之处	1.6
Git代码管理	1.7
集成终端	1.8
插件	1.9
插件市场	1.9.1
常用插件	1.9.2
Markdown Preview Enhanced	1.9.2.1
indent-rainbow	1.9.2.2
Bracket Pair Colorizer	1.9.2.3
Paste Image	1.9.2.4
文件图标主题	1.9.2.5
调试代码	1.10
调试Python	1.10.1
Mac中用VSCode调试Python	1.10.1.1
Windows中用VSCode调试Python	1.10.1.2
经验心得	1.10.1.3
性能	1.11
附录	1.12
参考资料	1.12.1

史上最好用的编辑器：VSCode

- 最新版本： `v1.2`
- 更新时间： `20200216`

简介

介绍史上用过的最好用的编辑器VSCode，以及各种好用的功能，包括但不限于常用快捷键，调试Python，常用插件，搜索功能，包括正则搜索，文件编码，git仓库代码管理，各种主题字体和插件，强大好用的插件市场，代码格式化等等各种好用的功能，且性能优越。是目前最值得推荐的编辑器，没有之一。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- [crifan/best_editor_vscode](#): 史上最好用的编辑器：VSCode

如何使用此Gitbook源码去生成发布为电子书

详见：[crifan/gitbook_template: demo how to use crifan gitbook template and demo](#)

在线浏览

- [史上最好用的编辑器：VSCode book.crifan.com](#)
- [史上最好用的编辑器：VSCode crifan.github.io](#)

离线下载阅读

- [史上最好用的编辑器：VSCode PDF](#)
- [史上最好用的编辑器：VSCode ePub](#)
- [史上最好用的编辑器：VSCode Mobi](#)

版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分内容参考自网络，均已备注了出处。如有发现侵犯您的版权，请通过邮箱联系我 `admin` 艾特 `crifan.com`，我会尽快删除。谢谢合作。

鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

更多其他电子书

本人 [crifan](#) 还写了其他 100+ 本电子书教程，感兴趣可移步至：

[crifan/crifan_ebook_readme: Crifan的电子书的使用说明](#)

crifan.com, 使用 [署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2021-01-16 23:59:24

VSCode简介

对于 代码编辑器 = Editor , IDE 来说, 目前有很多种。

尤其是编辑器方面, 自己之前也用过很多, 比如:

- Win平台
 - Notepad2
 - Notepad++
 - = npp
 - 等等
- Mac平台
 - Atom
 - Sublime
 - 等等

除了Win平台中Notepad++还算不错外, 其他很多都不够好用。

VSCode = Visual Studio Code , 是目前用过的最好用的、跨平台的、免费的、编辑器。

关于常见的编辑器和IDE的总结

详见: [编辑器和IDE总结](#)

VSCode的优点

- 轻量级
- 界面美观
 - 代码高亮效果好
- 功能强大且易用
 - 除了 文本编辑器 该有的功能
 - 支持 插件 : 可以无限扩展额外功能
 - 加上很多插件, 几乎可以实现很多复杂的, 高级的IDE才能实现的功能
 - 比如 调试Python代码 等等
 - 本身内置支持 git
 - 便于代码版本管理
- 跨平台
 - 支持多种平台:
 - Windows
 - Mac
 - Linux
 - 等等

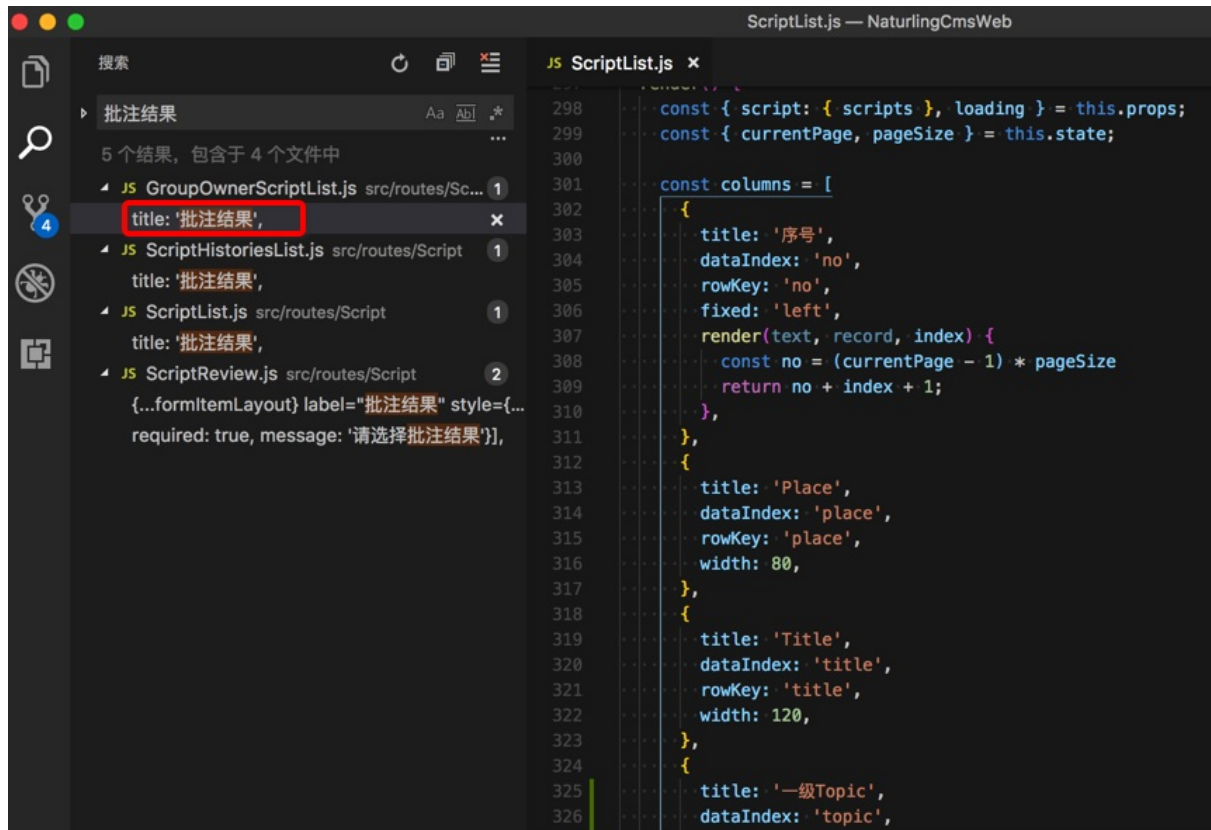
crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2019-03-17 23:23:20

界面布局

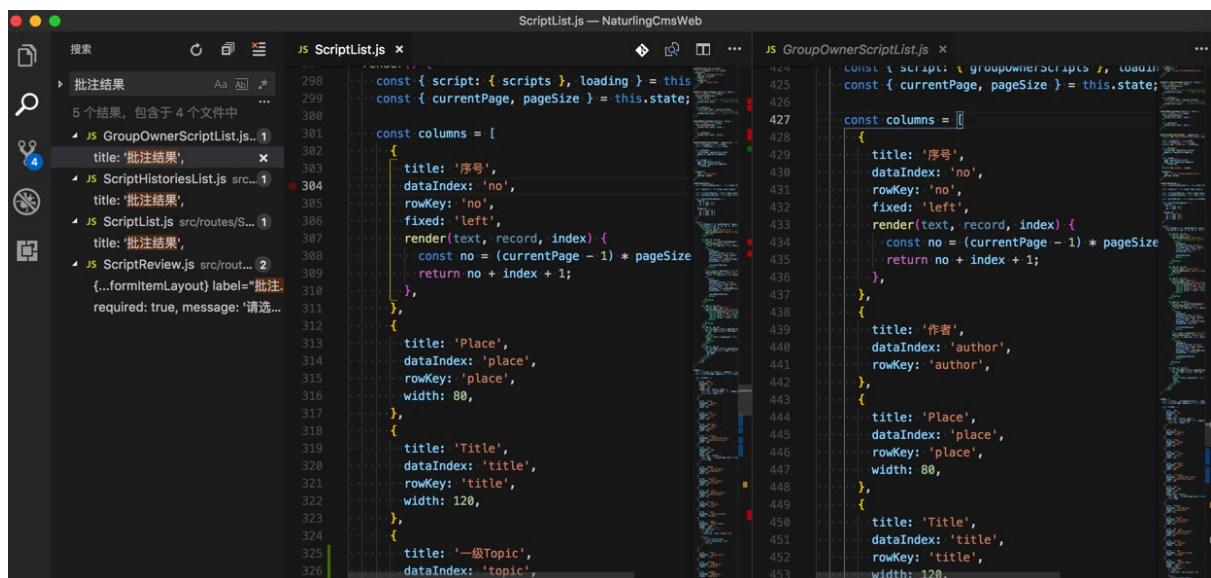
下面整理 VSCode 的界面和布局。

左右布局显示方便文件对比

在已经打开了一个文件：想要去和另外一个文件对比时， Command + 鼠标点击



即可把第二个文件放在右边，左右布局，便于对比2个文件的内容：



还是很方便的。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook 最后更新: 2019-03-16 15:20:30

常用快捷键

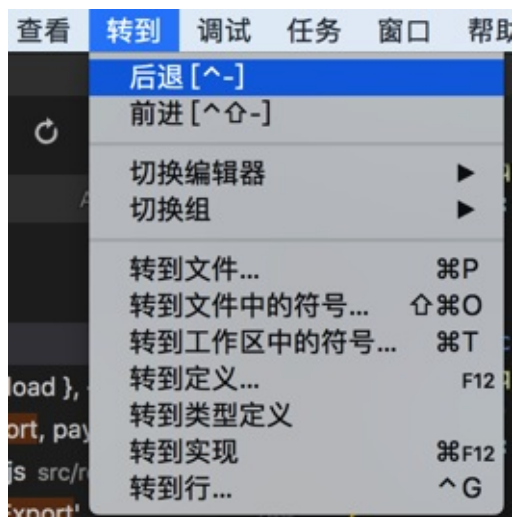
此处整理出 VSCode 中最常用的一些快捷键，方便查询和使用。

常用的快捷键

光标位置移动

- 鼠标/光标返回之前的位置 = 上一个位置 = 前一个位置：
 - Mac : `Control + -`
- 鼠标/光标返回之后的位置
 - Mac : `Shift + Control + -`

后来也看到了工具栏中有显示快捷键：



大小写转换

默认是没有的，可以自己去设置。比如：

- `Ctrl + Shift + l` : 转为小写
 - `l` : 表示 lowercase
- `Ctrl + Shift + u` : 转为大写
 - `u` : 表示 uppercase

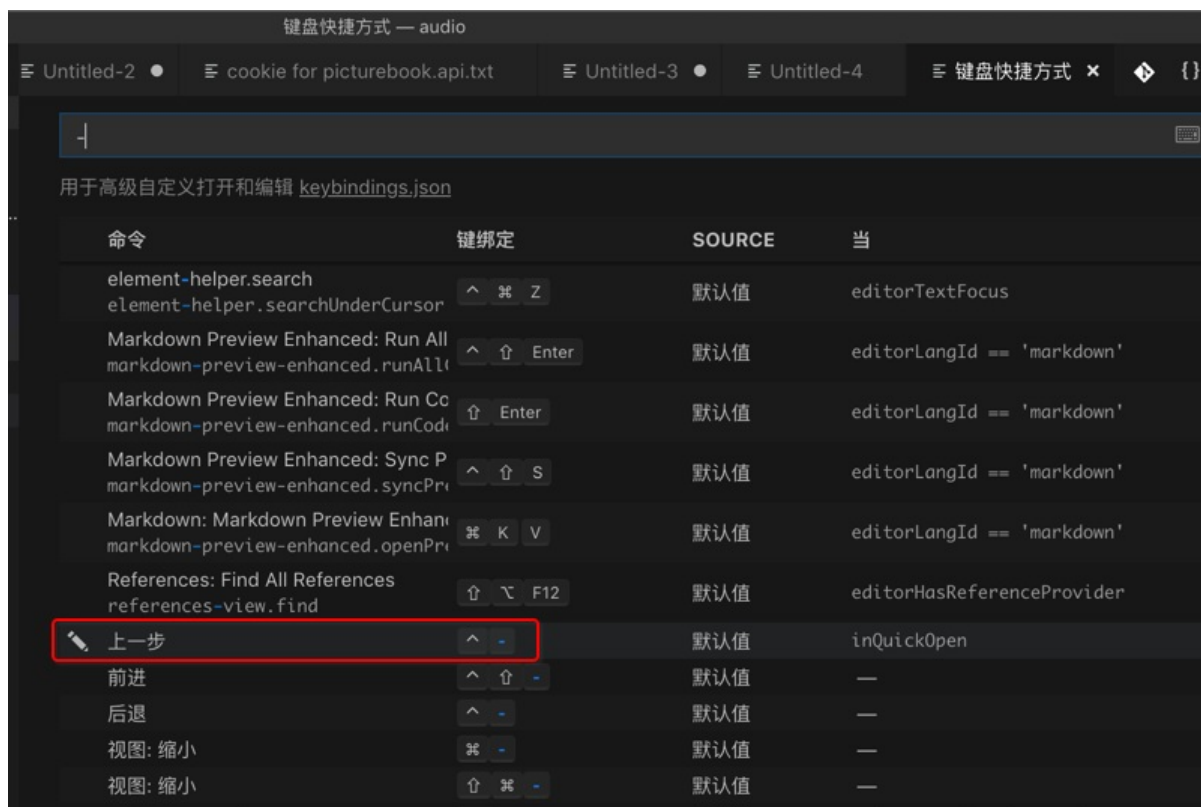
详见：[【已解决】VSCode中对选中内容切换大小写的快捷键](#)

如何找到某个功能的快捷键是什么

另外，可以去搜到自己要的快捷键是什么：



然后就可以通过搜短横线 - 就可以找到对应的快捷键的功能是 上一步：

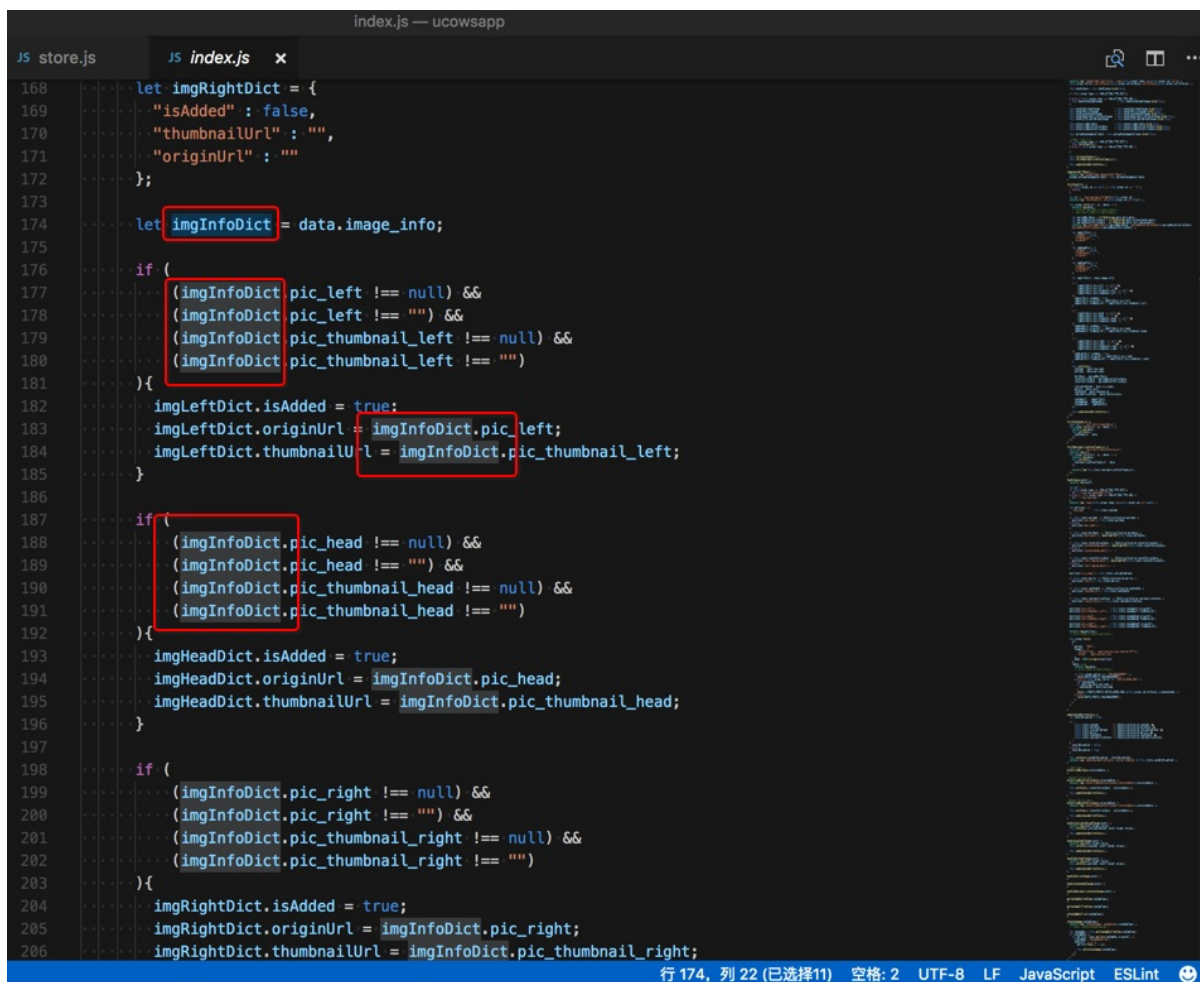


crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-16 20:59:08

代码编辑器常用功能

选中自动高亮

VSCode编辑器支持常见的，很方便的一个功能是：双击文字（变量），自动高亮其他部分的对应文字（变量）



The screenshot shows a VS Code editor window with a JavaScript file named 'index.js'. The code is as follows:

```
168 let imgRightDict = {
169   "isAdded" : false,
170   "thumbnailUrl" : "",
171   "originUrl" : ""
172 };
173
174 let imgInfoDict = data.image_info;
175
176 if (
177   (imgInfoDict.pic_left !== null) &&
178   (imgInfoDict.pic_left !== "") &&
179   (imgInfoDict.pic_thumbnail_left !== null) &&
180   (imgInfoDict.pic_thumbnail_left !== "")
181 ){
182   imgLeftDict.isAdded = true;
183   imgLeftDict.originUrl = imgInfoDict.pic_left;
184   imgLeftDict.thumbnailUrl = imgInfoDict.pic_thumbnail_left;
185 }
186
187 if (
188   (imgInfoDict.pic_head !== null) &&
189   (imgInfoDict.pic_head !== "") &&
190   (imgInfoDict.pic_thumbnail_head !== null) &&
191   (imgInfoDict.pic_thumbnail_head !== "")
192 ){
193   imgHeadDict.isAdded = true;
194   imgHeadDict.originUrl = imgInfoDict.pic_head;
195   imgHeadDict.thumbnailUrl = imgInfoDict.pic_thumbnail_head;
196 }
197
198 if (
199   (imgInfoDict.pic_right !== null) &&
200   (imgInfoDict.pic_right !== "") &&
201   (imgInfoDict.pic_thumbnail_right !== null) &&
202   (imgInfoDict.pic_thumbnail_right !== "")
203 ){
204   imgRightDict.isAdded = true;
205   imgRightDict.originUrl = imgInfoDict.pic_right;
206   imgRightDict.thumbnailUrl = imgInfoDict.pic_thumbnail_right;
```

Red boxes highlight the variable 'imgInfoDict' in the assignment on line 174 and its subsequent uses as a property access (e.g., 'imgInfoDict.pic_left') in the conditional checks on lines 177-180, 188-191, and 199-202. The status bar at the bottom indicates '行 174, 列 22 (已选择11) 空格: 2 UTF-8 LF JavaScript ESLint'.

-> 对于编写代码期间查看变量或函数的调用，很方便。

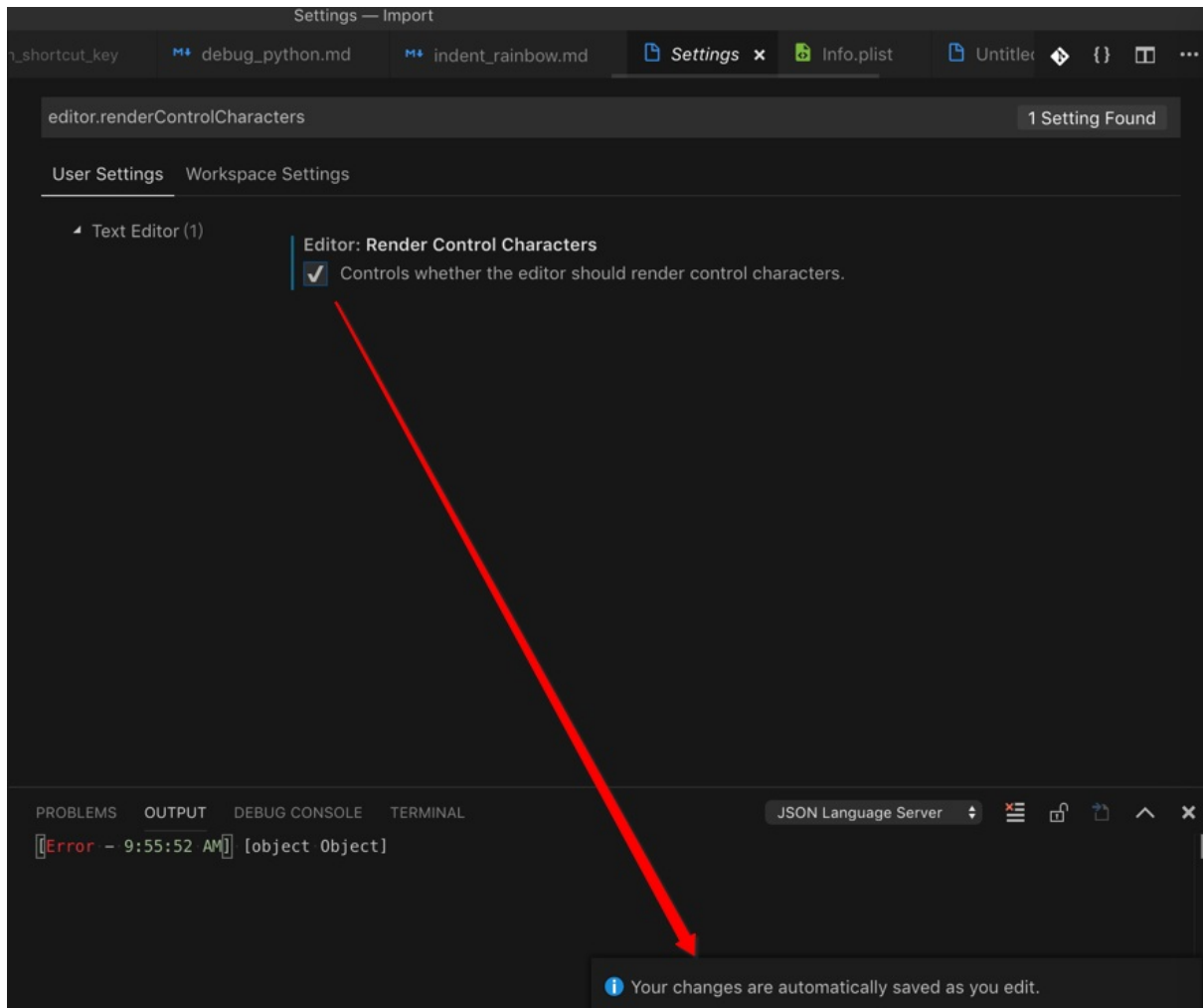
支持不同主题和字体

[【记录】VSCode试试其他主题和字体和插件](#)

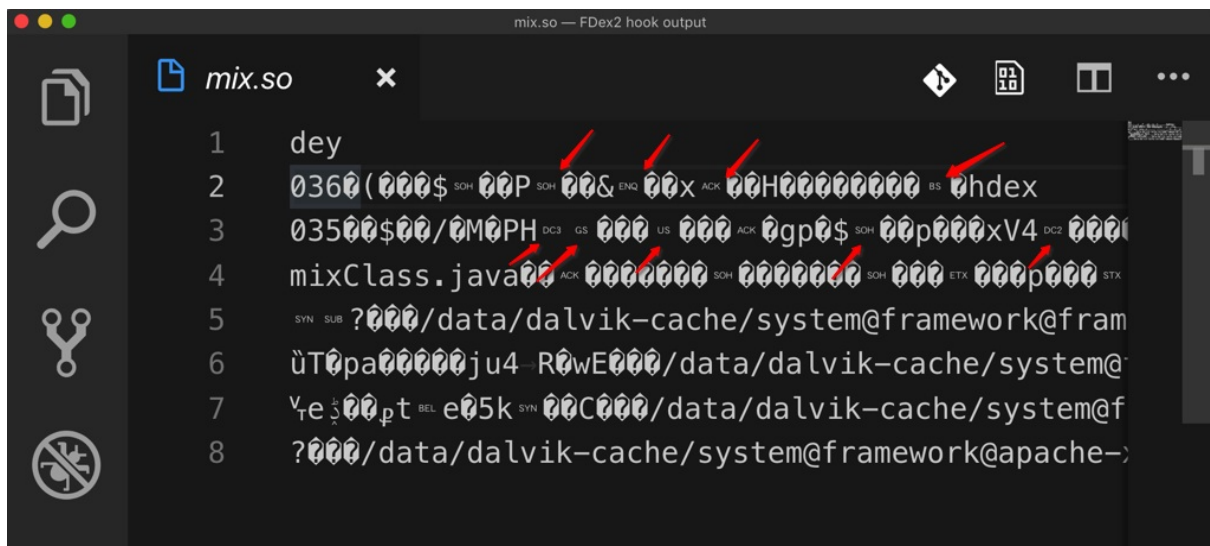
显示特殊的不可见的控制字符

设置显示控制字符：

Code -> 首选项 -> Settings -> 搜 editor.renderControlCharacters -> 勾选: Editor: Render Control Characters 中的 Controls whether the editor should render control characters



效果:



详见:

【已解决】VSCode中显示特殊的不可见的控制字符

设置Tab的宽度=空格的个数

Code-》首选项-》设置-》点击：

```
// 一个制表符等于的空格数。该设置在 `editor.detectIndentation` 启用时根据文件内容进行重写。  
"editor.tabSize": 4,
```

左边的编辑按钮 -》复制到设置-》把 4 改为 2：

```
"editor.tabSize": 2
```

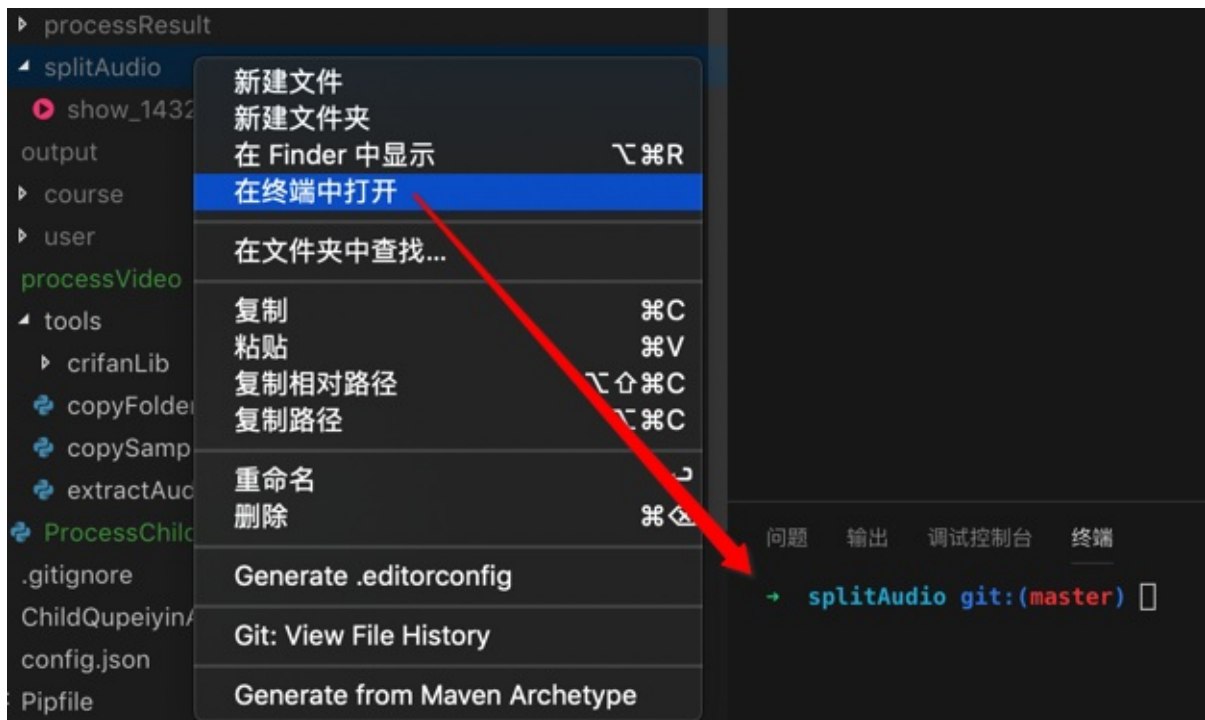
即可立刻生效：按Tab后，缩进就从4个空格的宽度，变成2个空格的宽度了。

详见：[【已解决】VSCode中如何设置Tab缩进为2个空格而不是4个空格的宽度](#)

其他方便好用的功能

快速在终端中打开文件夹

右键某文件夹 -》在终端中打开



则可以直接快速调用内置终端打开对应文件夹，然后就可以继续在终端中做事情了，比如此处去用ffmpeg分割mp3。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2019-03-17 23:08:52

打开和新建项目

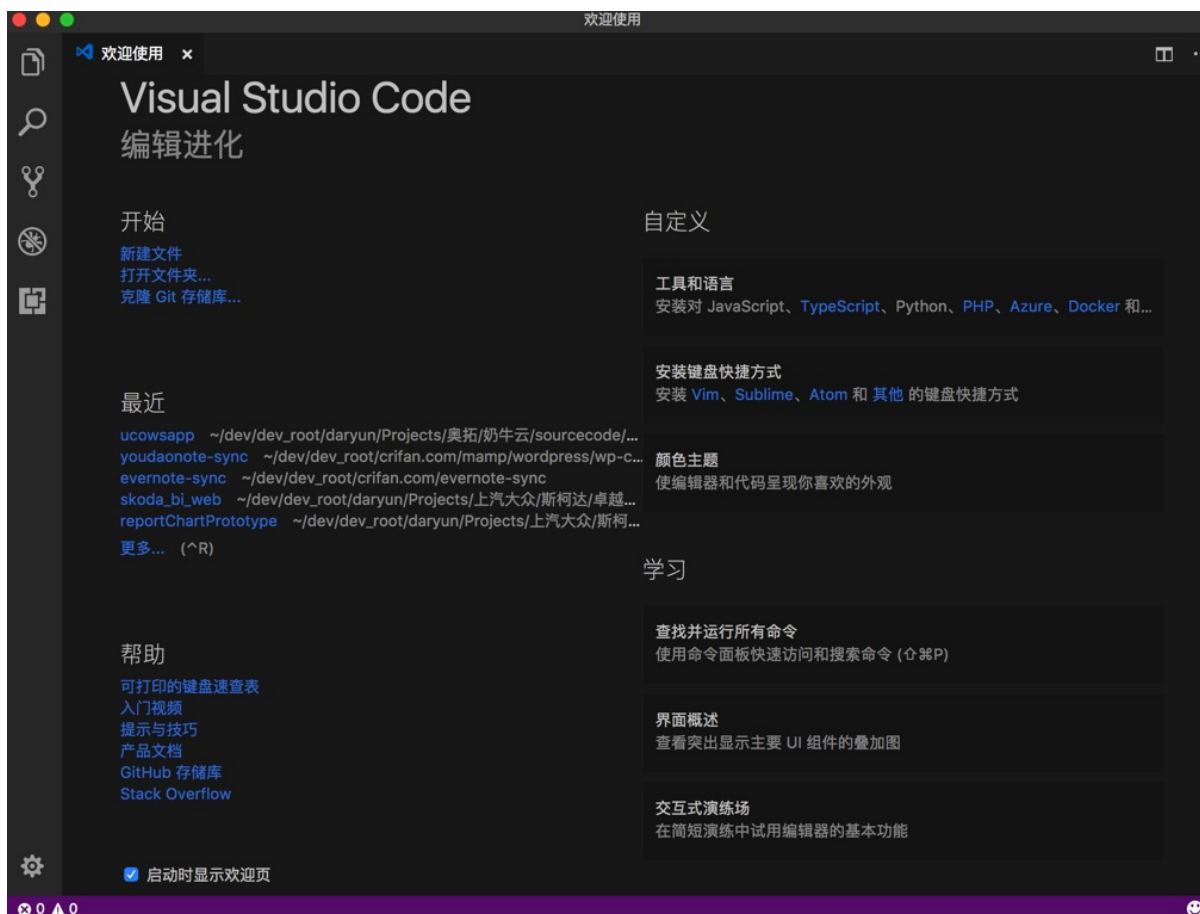
新建项目很方便

普通新建项目

文件 -> 新建窗口

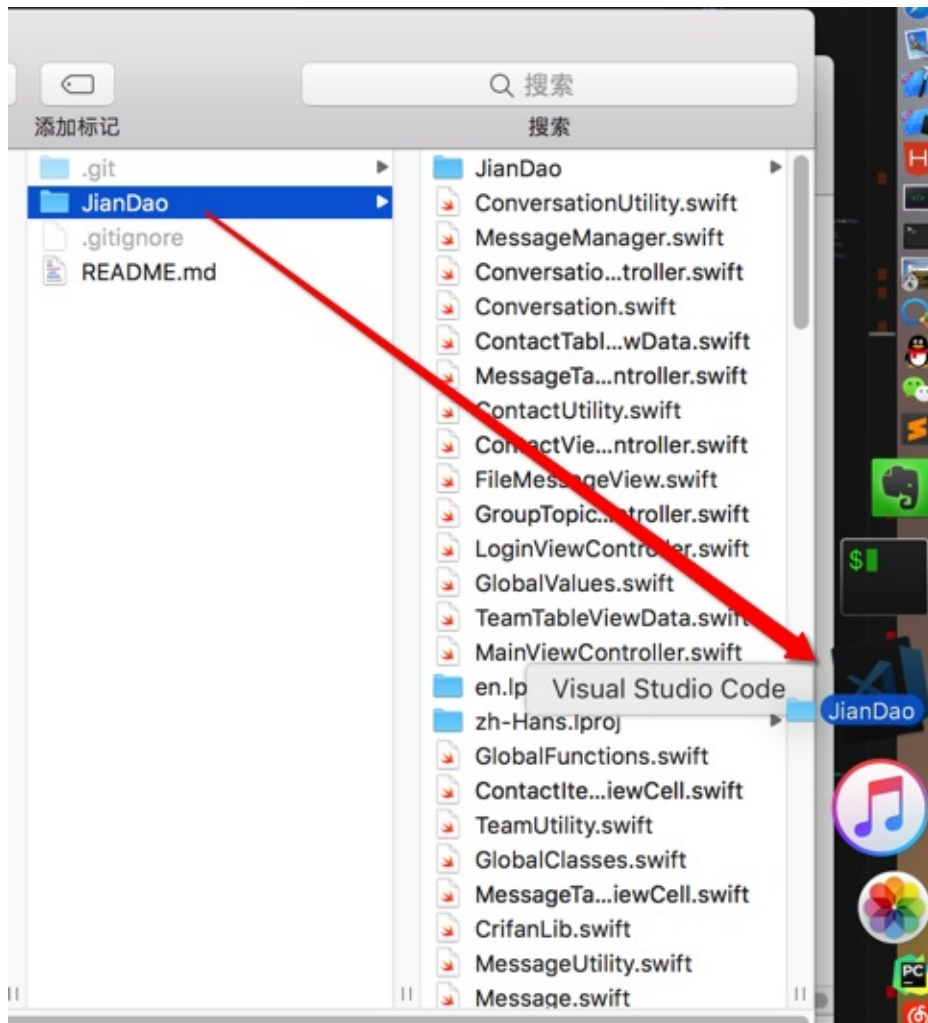


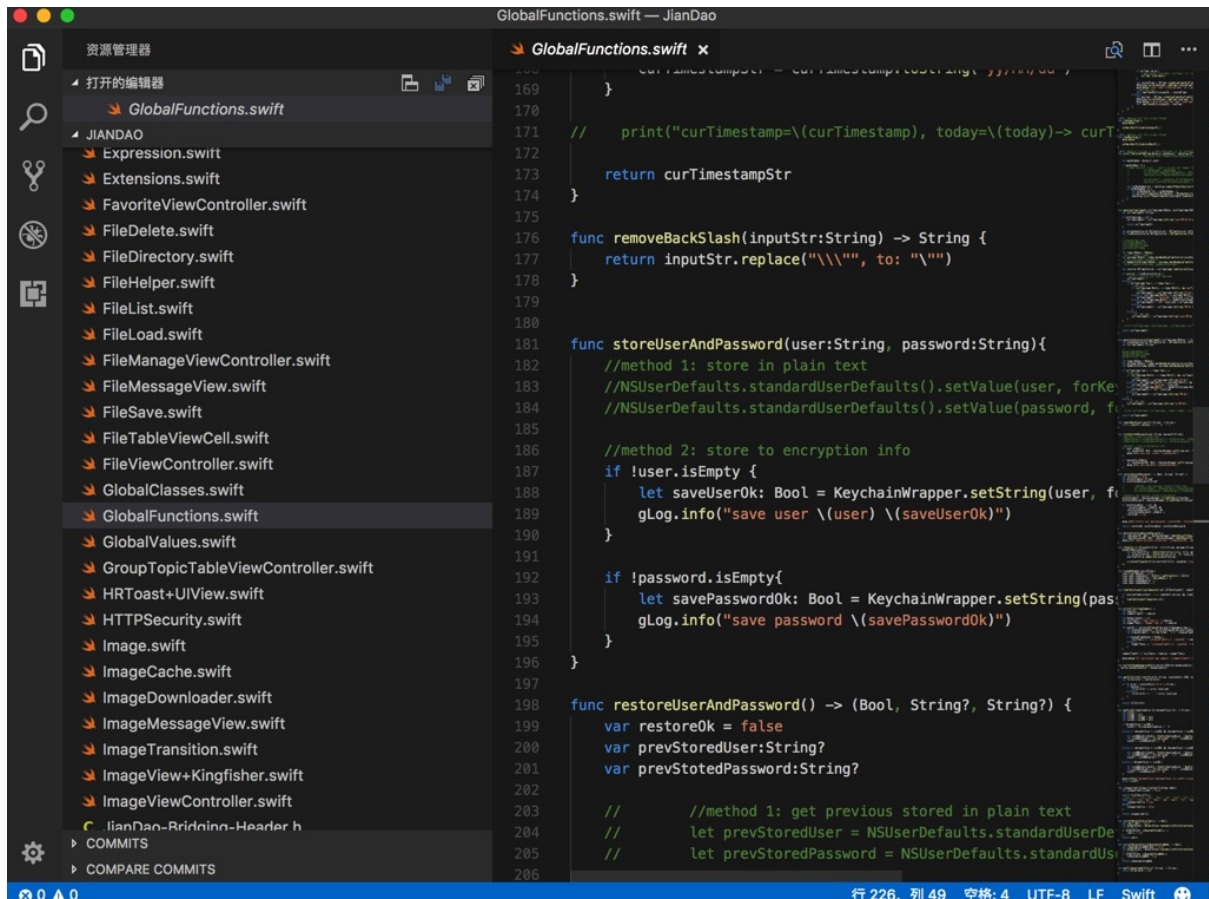
即可新建项目和窗口了：



直接拖动文件夹新建项目

另外，还支持，直接拖动文件到到VSCode图标，即可新建项目：





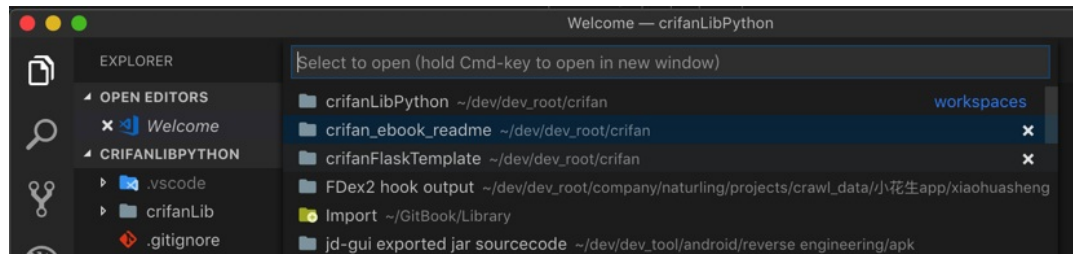
非常方便 -> 用了后, 就离不开了。

打开项目

打开之前曾打开过的项目, 有多种方式:

- 从新建窗口中的 Recent中打开:

-
- 如果Recent中没有，点击More，也会列出来更多之前打开过的项目：



- 文件->最近打开->选择某项目：

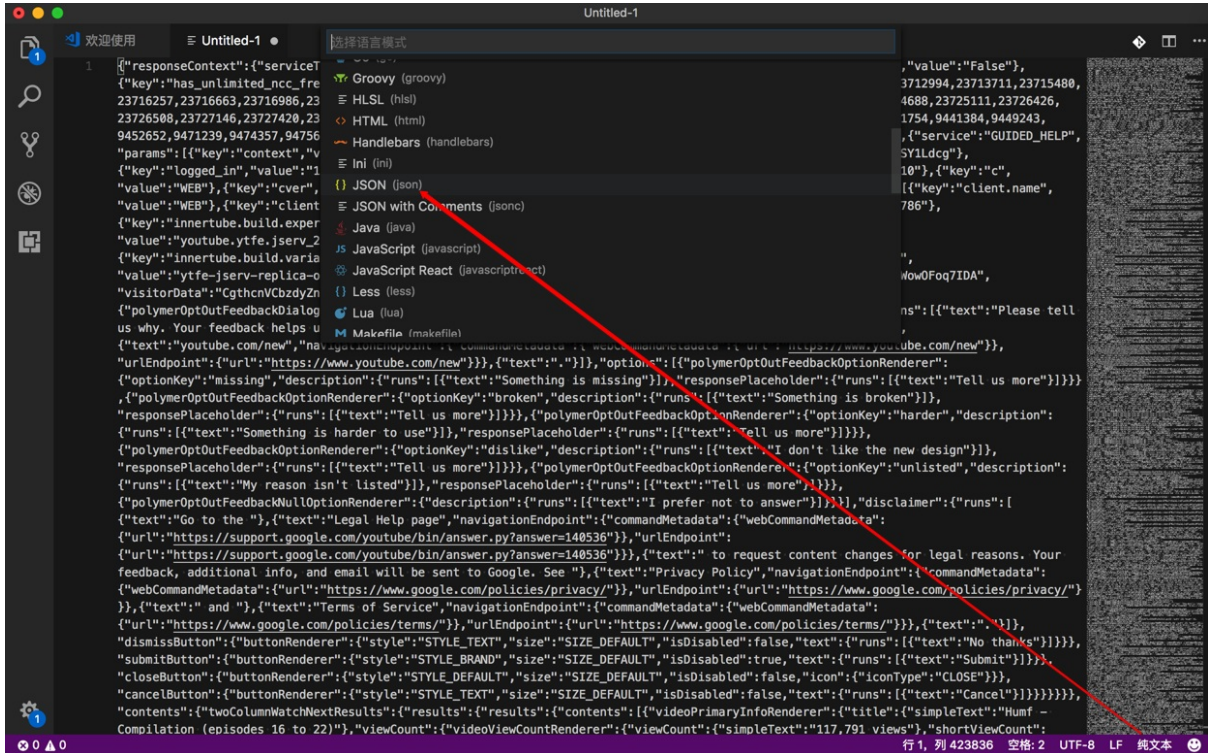
○

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新： 2019-03-17 22:56:48

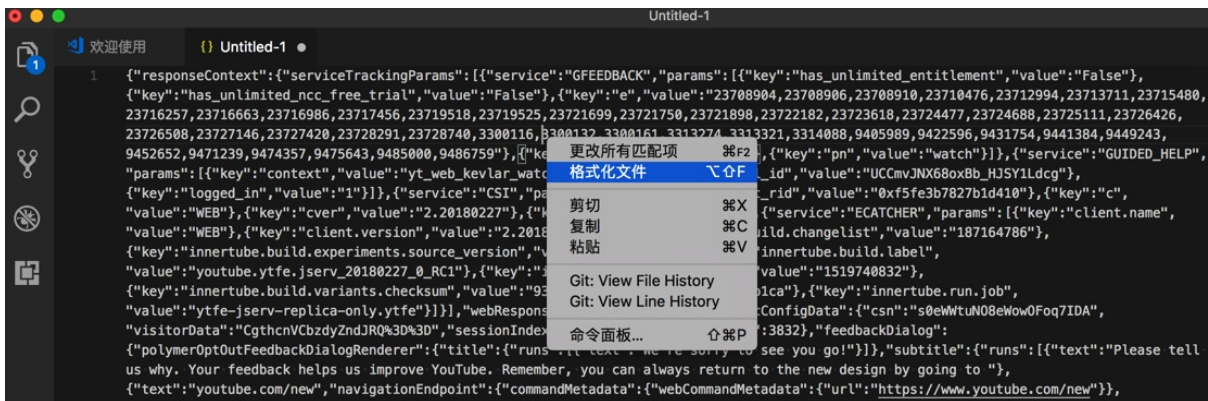
代码格式化

比如想要把一段超长的字符串，是json格式，想要格式化，便于查看内容。

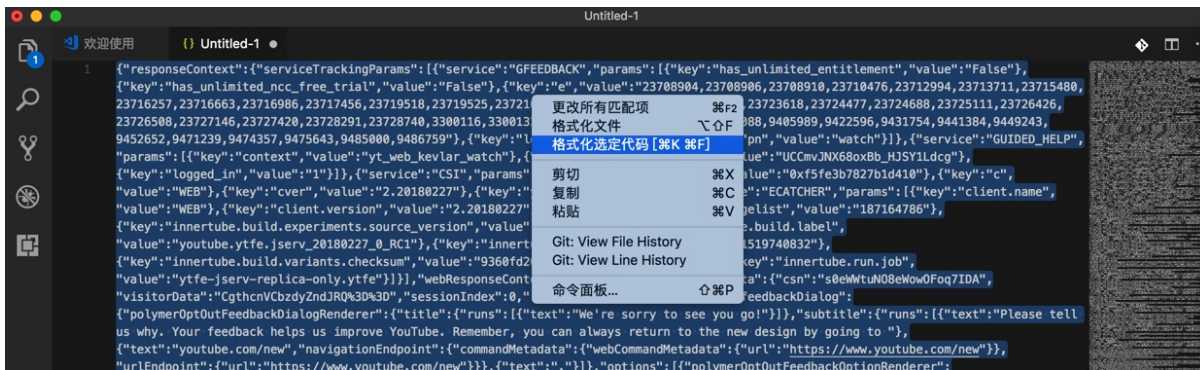
先把字符串内容粘贴过去。默认的文件格式是 纯文本，所以去设置为 JSON



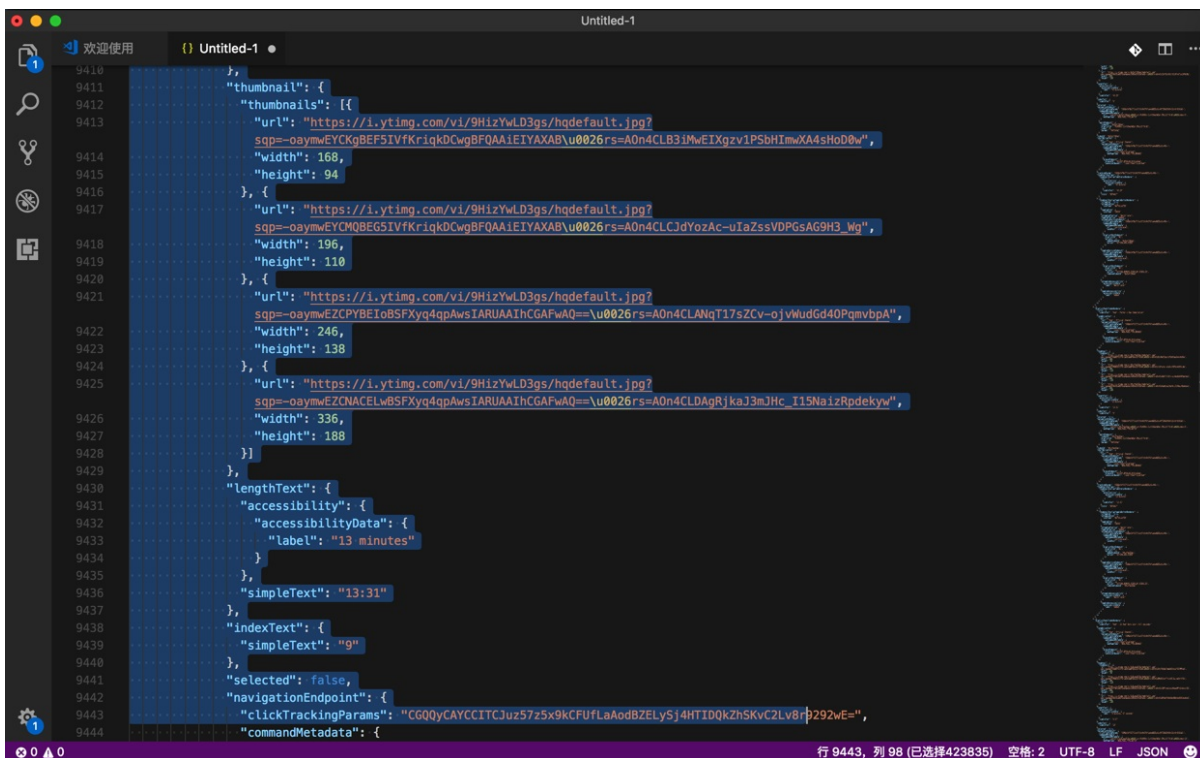
然后再去右键-> 格式化文件



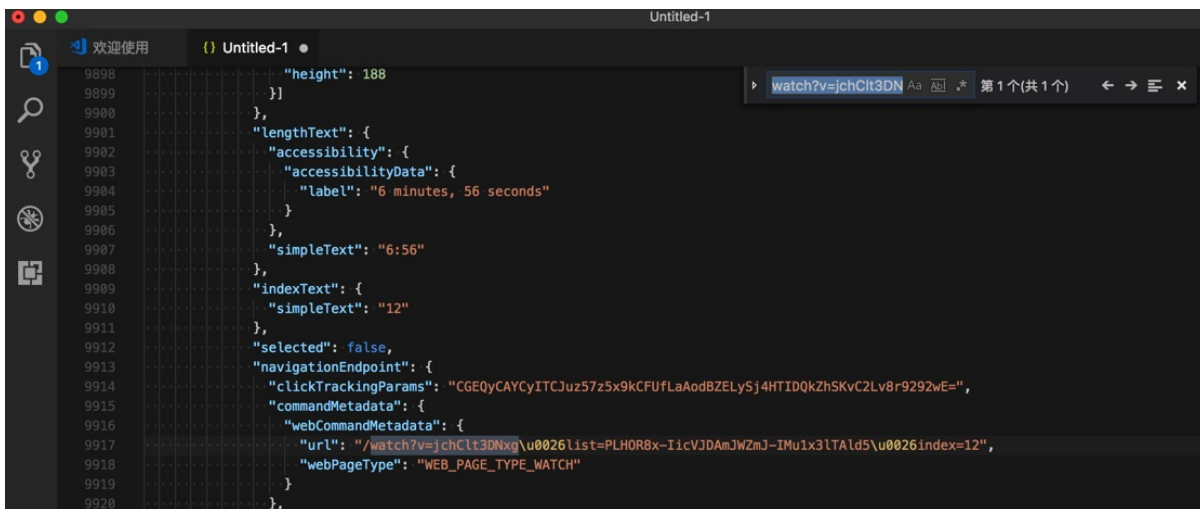
或者: 全选-> 右键-> 格式化选定代码:



即可看到格式化后的，有了代码高亮，且带缩进的内容了：



然后可以方便的查看（搜索到的）内容：

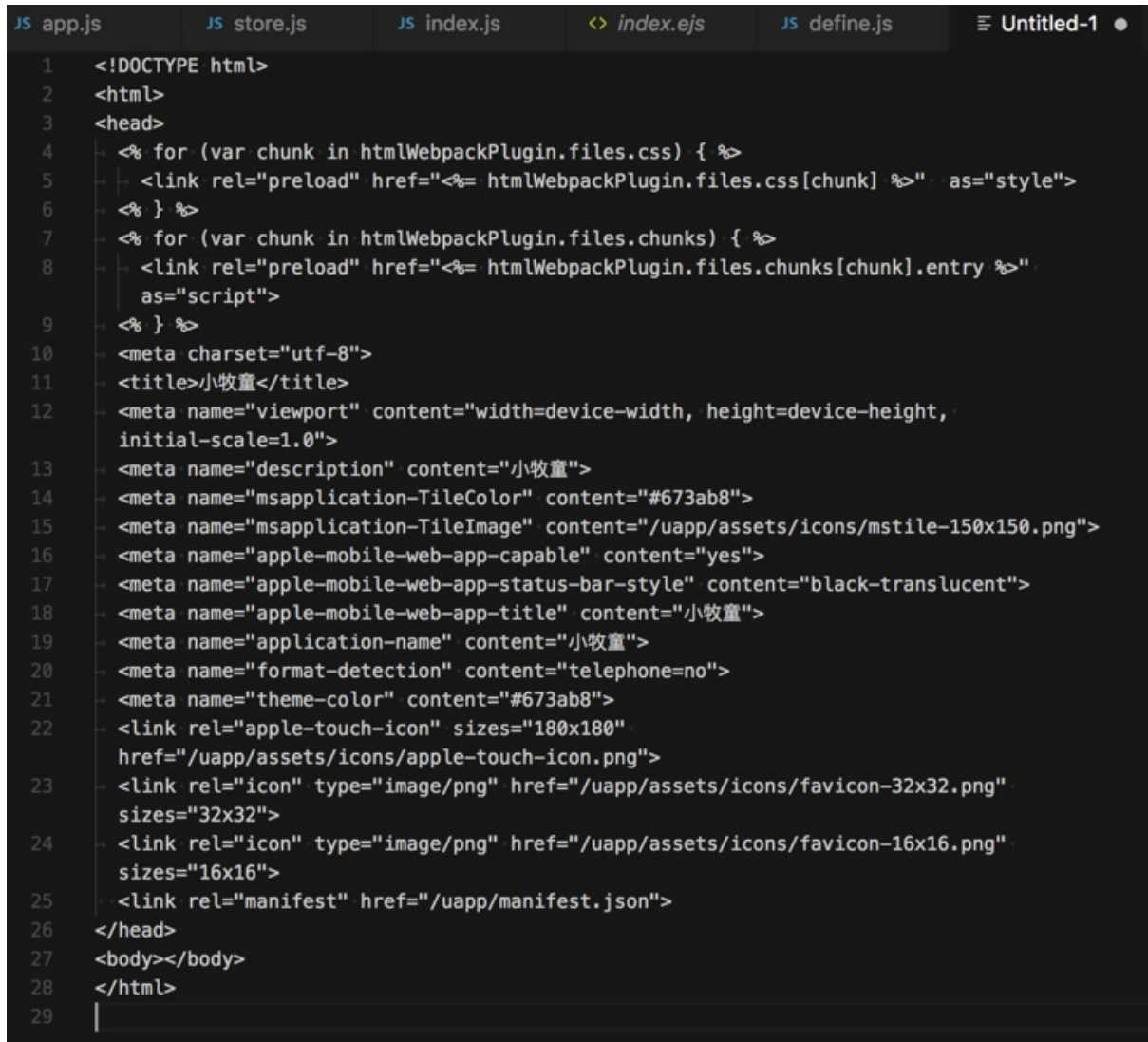


语法高亮

设置文件和代码的语法高亮：

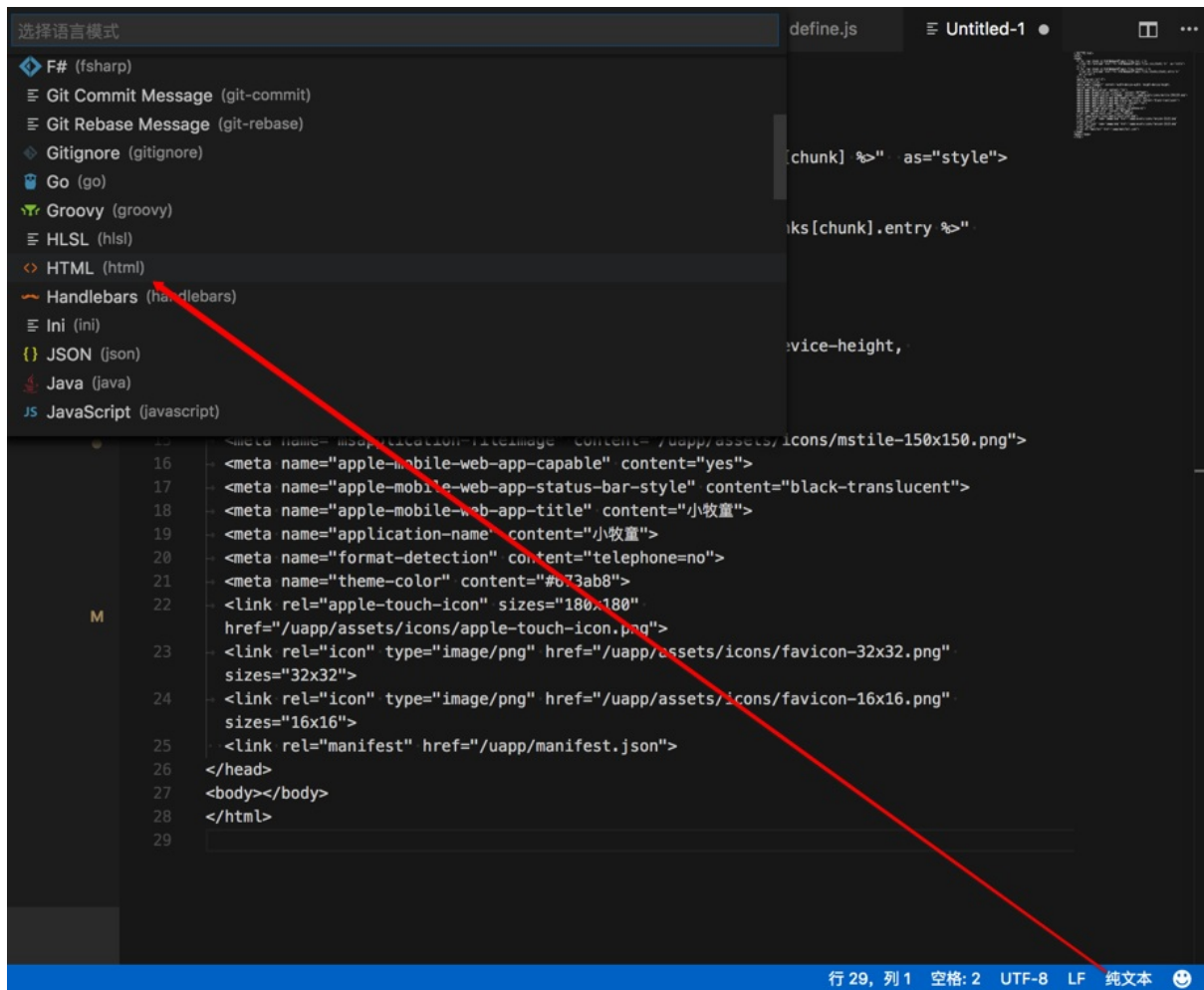
对于未保存的文件时，需要设置文件类型，才能使得语法高亮生效

比如新建文件，粘贴html代码，此时代码无法自动高亮



```
JS app.js JS store.js JS index.js <> index.ejs JS define.js ☰ Untitled-1 ●
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <% for (var chunk in htmlWebpackPlugin.files.css) { %>
5     <link rel="preload" href="<%= htmlWebpackPlugin.files.css[chunk] %>" as="style">
6   <% } %>
7   <% for (var chunk in htmlWebpackPlugin.files.chunks) { %>
8     <link rel="preload" href="<%= htmlWebpackPlugin.files.chunks[chunk].entry %>"
9     as="script">
10    <% } %>
11    <meta charset="utf-8">
12    <title>小牧童</title>
13    <meta name="viewport" content="width=device-width, height=device-height,
14    initial-scale=1.0">
15    <meta name="description" content="小牧童">
16    <meta name="msapplication-TileColor" content="#673ab8">
17    <meta name="msapplication-TileImage" content="/uapp/assets/icons/mstile-150x150.png">
18    <meta name="apple-mobile-web-app-capable" content="yes">
19    <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">
20    <meta name="apple-mobile-web-app-title" content="小牧童">
21    <meta name="application-name" content="小牧童">
22    <meta name="format-detection" content="telephone=no">
23    <meta name="theme-color" content="#673ab8">
24    <link rel="apple-touch-icon" sizes="180x180"
25    href="/uapp/assets/icons/apple-touch-icon.png">
26    <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-32x32.png"
27    sizes="32x32">
28    <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-16x16.png"
29    sizes="16x16">
30    <link rel="manifest" href="/uapp/manifest.json">
31  </head>
32  <body></body>
33 </html>
```

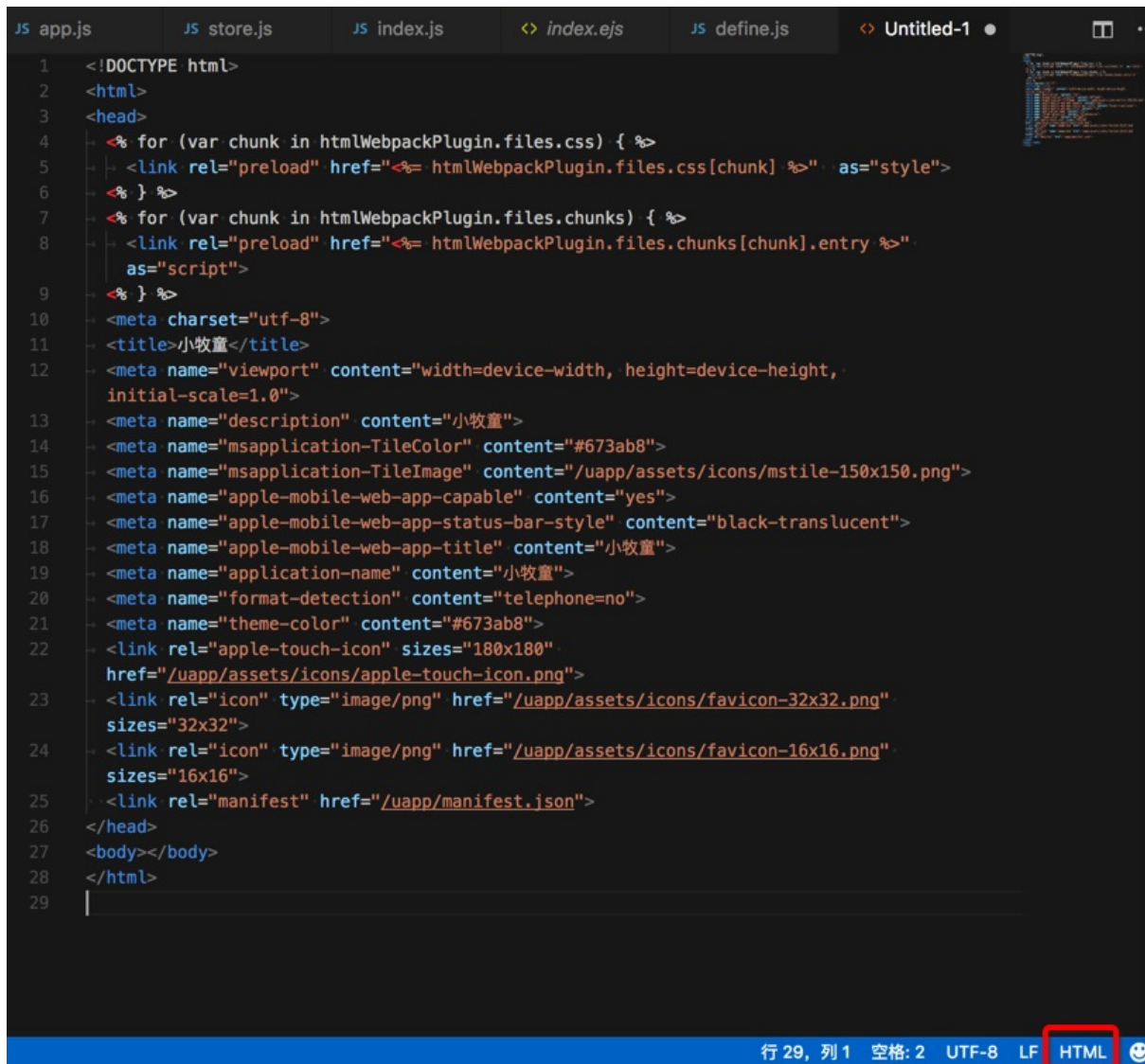
点击右下角的 纯文本，在弹出的语言列表中选择 HTML：



The screenshot shows a code editor interface. On the left, a '选择语言模式' (Select Language Mode) menu is open, listing various languages like F#, Git Commit Message, Groovy, HTML, JavaScript, etc. The 'HTML (html)' option is highlighted with a red arrow. The main editor area shows HTML code with syntax highlighting. The code includes meta tags for mobile app capabilities and titles, and link tags for icons and a manifest file. The status bar at the bottom indicates '行 29, 列 1 空格: 2 UTF-8 LF 纯文本'.

```
15 <meta name="apple-application-header-image" content="/uapp/assets/icons/mstile-150x150.png">
16 <meta name="apple-mobile-web-app-capable" content="yes">
17 <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">
18 <meta name="apple-mobile-web-app-title" content="小牧童">
19 <meta name="application-name" content="小牧童">
20 <meta name="format-detection" content="telephone=no">
21 <meta name="theme-color" content="#073ab8">
22 <link rel="apple-touch-icon" sizes="180x180"
23 href="/uapp/assets/icons/apple-touch-icon.png">
24 <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-32x32.png"
25 sizes="32x32">
26 <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-16x16.png"
27 sizes="16x16">
28 <link rel="manifest" href="/uapp/manifest.json">
29 </head>
30 <body></body>
31 </html>
```

即可看到HTML代码高亮的效果了：



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <% for (var chunk in htmlWebpackPlugin.files.css) { %>
5     <link rel="preload" href="<%= htmlWebpackPlugin.files.css [chunk] %>" as="style">
6   <% } %>
7   <% for (var chunk in htmlWebpackPlugin.files.chunks) { %>
8     <link rel="preload" href="<%= htmlWebpackPlugin.files.chunks [chunk].entry %>"
9     as="script">
10    <% } %>
11    <meta charset="utf-8">
12    <title>小牧童</title>
13    <meta name="viewport" content="width=device-width, height=device-height,
14    initial-scale=1.0">
15    <meta name="description" content="小牧童">
16    <meta name="msapplication-TileColor" content="#673ab8">
17    <meta name="msapplication-TileImage" content="/uapp/assets/icons/mstile-150x150.png">
18    <meta name="apple-mobile-web-app-capable" content="yes">
19    <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">
20    <meta name="apple-mobile-web-app-title" content="小牧童">
21    <meta name="application-name" content="小牧童">
22    <meta name="format-detection" content="telephone=no">
23    <meta name="theme-color" content="#673ab8">
24    <link rel="apple-touch-icon" sizes="180x180"
25    href="/uapp/assets/icons/apple-touch-icon.png">
26    <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-32x32.png"
27    sizes="32x32">
28    <link rel="icon" type="image/png" href="/uapp/assets/icons/favicon-16x16.png"
29    sizes="16x16">
30    <link rel="manifest" href="/uapp/manifest.json">
31  </head>
32  <body></body>
33 </html>
```

行 29, 列 1 空格: 2 UTF-8 LF HTML

支持log类型的语法高亮

无意间发现，VSCode连log格式，都可以支持，都可以语法高亮：


```

232 [Sun Apr 29 11:27:44 2018] [notice] FastCGI: process manager initialized (pid 28669)
233 [Sun Apr 29 11:27:44 2018] [notice] Apache/2.2.32 (Unix) mod_wsgi/3.5 Python/2.7.13 PHP/7.1.6 mod_ss
234 [Mon Apr 30 15:13:27 2018] [notice] caught SIGTERM, shutting down
235 [Fri Jun 22 21:57:11 2018] [notice] Digest: generating secret for digest authentication ...
236 [Fri Jun 22 21:57:11 2018] [notice] Digest: done
237 [Fri Jun 22 21:57:11 2018] [notice] FastCGI: process manager initialized (pid 47606)
238 [Fri Jun 22 21:57:11 2018] [notice] Apache/2.2.32 (Unix) mod_wsgi/3.5 Python/2.7.13 PHP/7.1.6 mod_ss
239 [Fri Jun 22 21:57:20 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
240 [Fri Jun 22 21:57:20 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
241 [Fri Jun 22 21:57:20 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
242 unknown: fatal: file /etc/postfix/main.cf: parameter mail_owner: user postfix has same user ID as _p
243 [Fri Jun 22 22:12:15 2018] [error] [client :::] File does not exist: /Applications/MAMP/bin/mamp/far
244 [Fri Jun 22 22:18:56 2018] [notice] caught SIGTERM, shutting down
245 [Fri Jun 22 22:18:56 2018] [notice] Digest: generating secret for digest authentication ...
246 [Fri Jun 22 22:18:56 2018] [notice] Digest: done
247 [Fri Jun 22 22:18:56 2018] [notice] FastCGI: process manager initialized (pid 48973)
248 [Fri Jun 22 22:18:56 2018] [notice] Apache/2.2.32 (Unix) mod_wsgi/3.5 Python/2.7.13 PHP/7.1.6 mod_ss
249 [Fri Jun 22 22:25:58 2018] [error] [client :::] client denied by server configuration: /Users/crifan
250 [Fri Jun 22 22:49:57 2018] [error] [client :::] client denied by server configuration: /Users/crifan
251 [Fri Jun 22 22:49:57 2018] [error] [client :::] client denied by server configuration: /Users/crifan
252 [Sat Jun 23 12:18:33 2018] [error] [client :::] client denied by server configuration: /Users/crifan
253 [Sat Jun 23 12:24:22 2018] [error] [client :::] client denied by server configuration: /Users/crifan
254 [Thu Jun 28 20:53:34 2018] [notice] Digest: generating secret for digest authentication ...
255 [Thu Jun 28 20:53:34 2018] [notice] Digest: done
256 [Thu Jun 28 20:53:34 2018] [notice] FastCGI: process manager initialized (pid 30197)
257 [Thu Jun 28 20:53:34 2018] [notice] Apache/2.2.32 (Unix) mod_wsgi/3.5 Python/2.7.13 PHP/7.1.6 mod_ss
258 [Thu Jun 28 20:53:42 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
259 [Thu Jun 28 20:53:42 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
260 [Thu Jun 28 20:53:51 2018] [error] [client :::] File does not exist: /Users/crifan/dev/dev_root/crif
261 [Thu Jun 28 20:54:05 2018] [error] [client :::] client denied by server configuration: /Users/crifan
262 [Thu Jun 28 21:10:34 2018] [error] [client :::] File does not exist: /Applications/MAMP/bin/mamp/wp_
263

```

很方便查看内容。

另外一个截图：

```

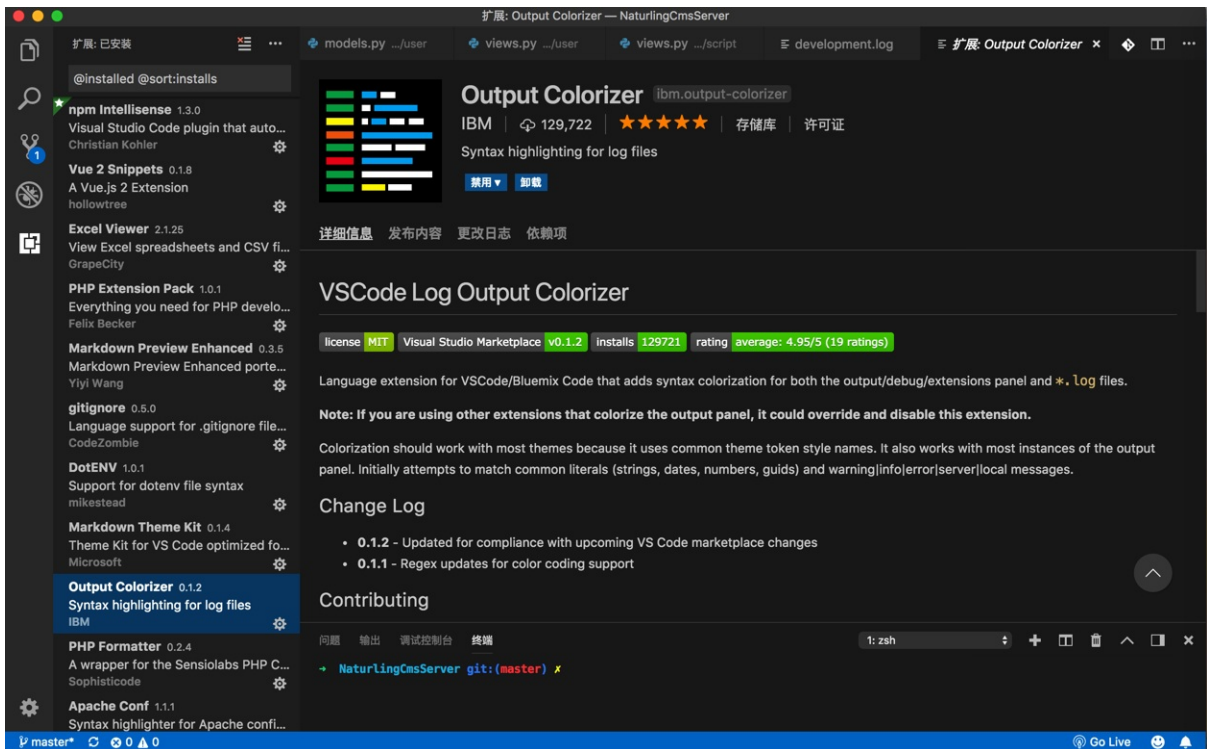
9 DEBUG 2018-07-09 15:17:28,015 utils 9314 123145337683968 (0.001) SET SESSION TRANSACTION ISOLATION LEVEL
10 DEBUG 2018-07-09 15:17:28,017 utils 9314 123145337683968 (0.002) SELECT `user_user`.`password`, `user_use
11 ERROR 2018-07-09 15:17:28,164 exception 9314 123145337683968 Internal Server Error: /api/v1/jwt-token-aut
Traceback (most recent call last):
13 File "/usr/local/lib/python3.6/site-packages/django/core/handlers/exception.py", line 35, in inner
14 response = get_response(request)
15 File "/usr/local/lib/python3.6/site-packages/django/core/handlers/base.py", line 158, in _get_response
16 response = self.process_exception_by_middleware(e, request)
17 File "/usr/local/lib/python3.6/site-packages/django/core/handlers/base.py", line 156, in _get_response
18 response = response.render()
19 File "/usr/local/lib/python3.6/site-packages/django/template/response.py", line 106, in render
20 self.content = self.rendered_content
21 File "/usr/local/lib/python3.6/site-packages/rest_framework/response.py", line 72, in rendered_content
22 ret = renderer.render(self.data, accepted_media_type, context)
23 File "/usr/local/lib/python3.6/site-packages/rest_framework/renderers.py", line 105, in render
24 allow_nan=not self.strict, separators=separators
25 File "/usr/local/lib/python3.6/site-packages/rest_framework/utils/json.py", line 28, in dumps
26 return json.dumps(*args, **kwargs)
27 File "/usr/local/Cellar/python/3.6.4/Frameworks/Python.framework/Versions/3.6/lib/python3.6/json/_in
28 **kw).encode(obj)
29 File "/usr/local/Cellar/python/3.6.4/Frameworks/Python.framework/Versions/3.6/lib/python3.6/json/enco
30 chunks = self.iterencode(o, _one_shot=True)
31 File "/usr/local/Cellar/python/3.6.4/Frameworks/Python.framework/Versions/3.6/lib/python3.6/json/enco
32 return _iterencode(o, 0)
33 File "/usr/local/lib/python3.6/site-packages/rest_framework/utils/encoders.py", line 68, in default
34 return super(JSONEncoder, self).default(obj)
35 File "/usr/local/Cellar/python/3.6.4/Frameworks/Python.framework/Versions/3.6/lib/python3.6/json/enco
36 o.__class__.__name__)
37 TypeError: Object of type 'User' is not JSON serializable
38 ERROR 2018-07-09 15:17:28,173 exception 9314 123145337683968 Internal Server Error: /api/v1/jwt-token-aut

```

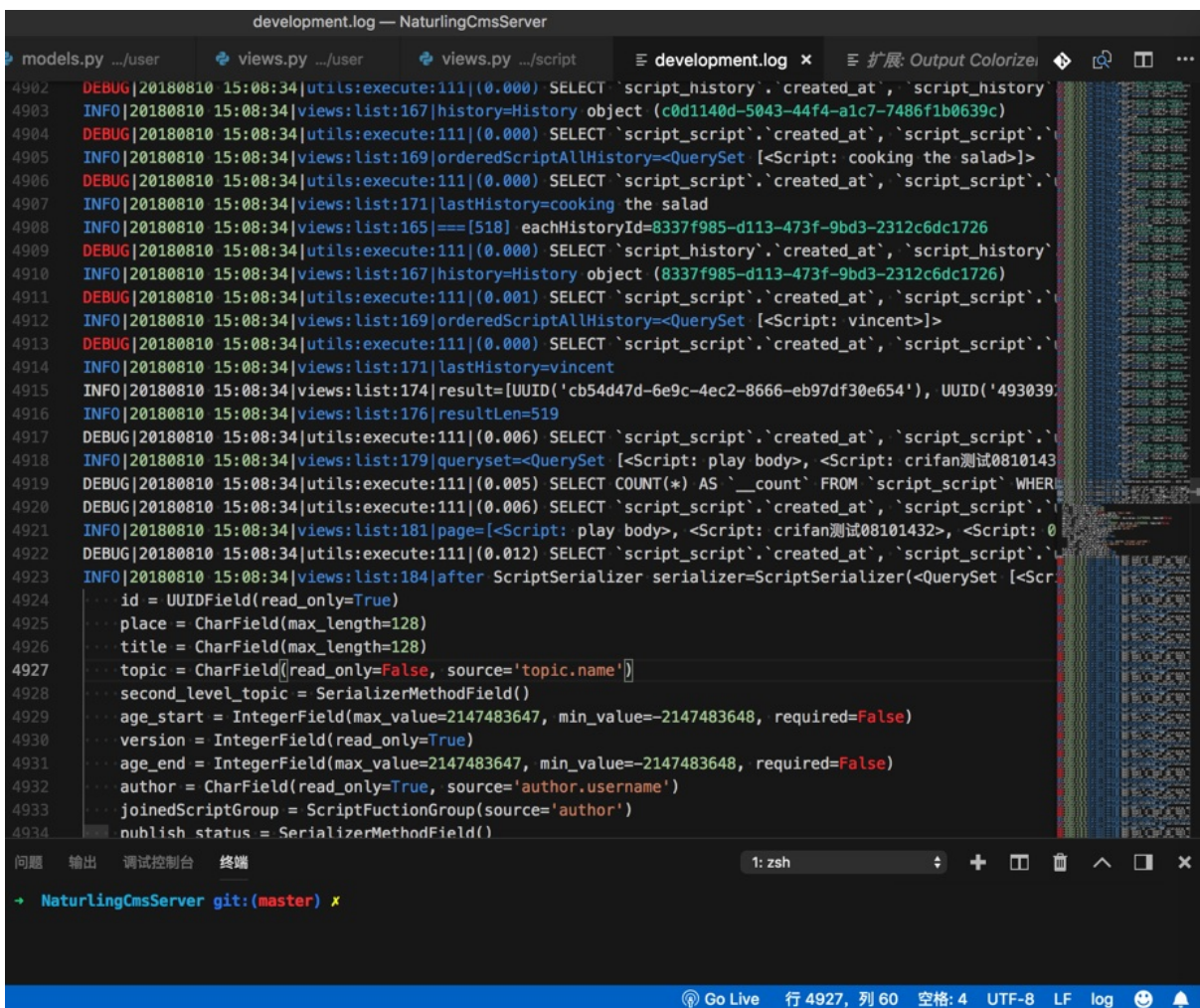
-> 后来不知道为何突然log文件丢失语法高亮了

所以又去找了个插件：

Output Colorizer



安装后，效果也很不错：



crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-17
23:12:20

文件编码

支持文件编码格式自动检测

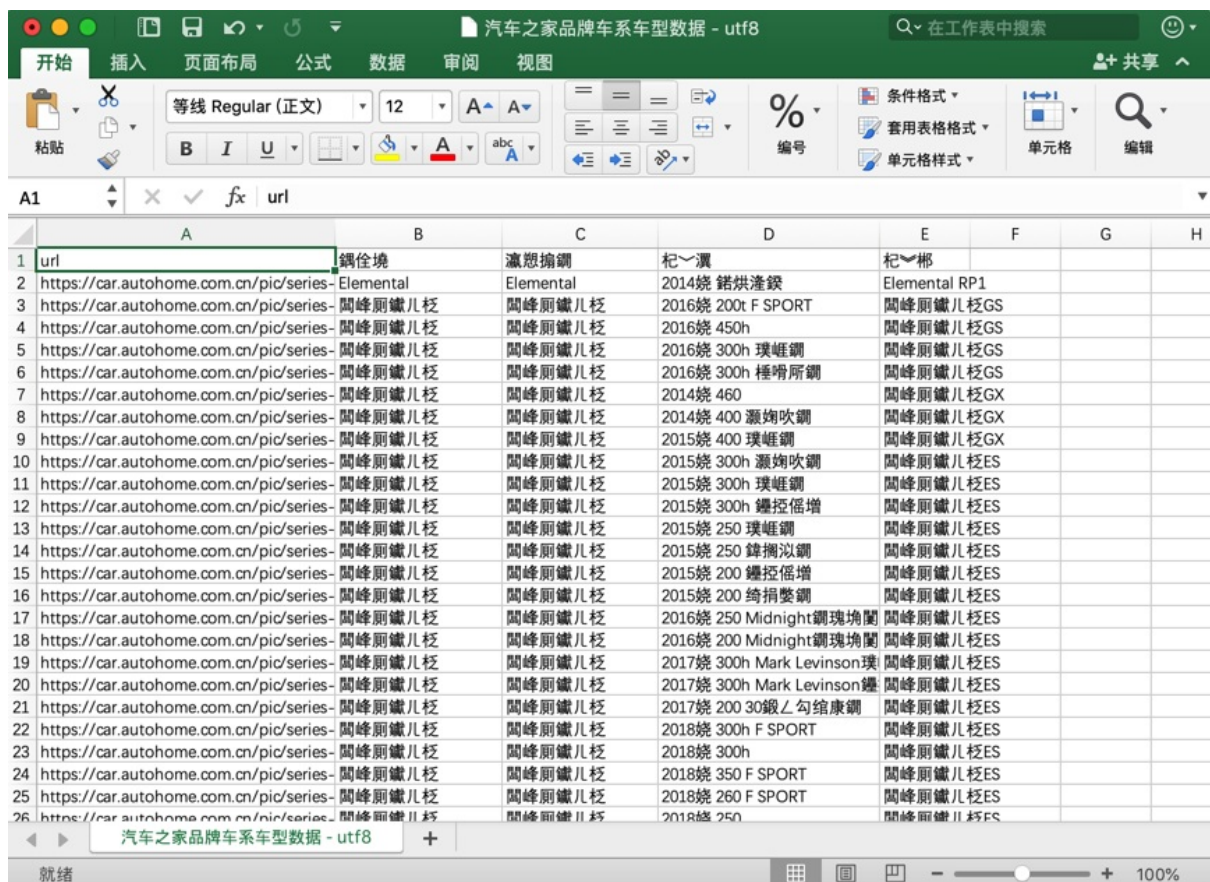
且可以更改编码重新保存。

VSCoDe中文件编码应用案例

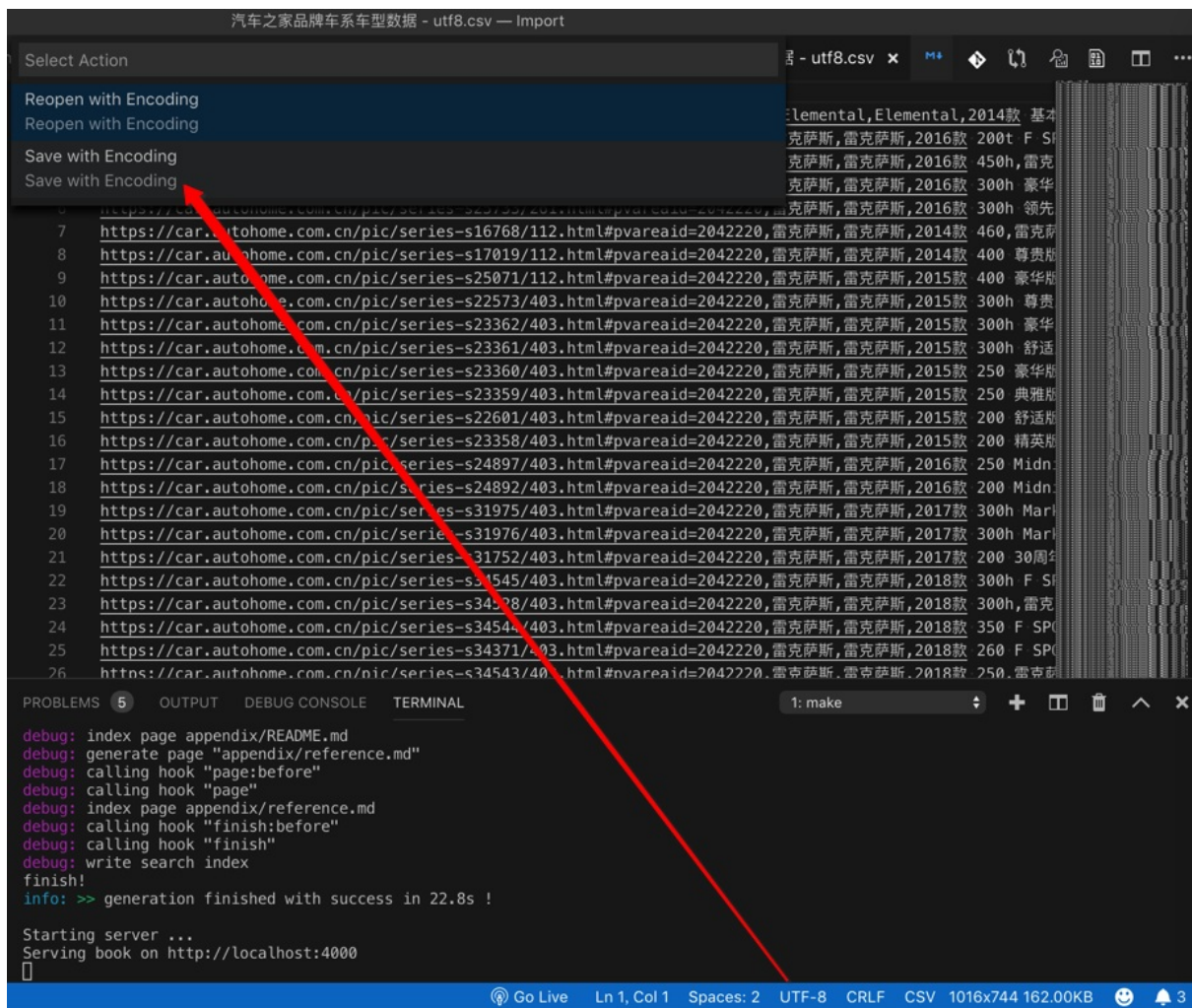
把utf-8编码转换为gb18030使得Mac中Excel打开csv不再乱码

比如之前遇到过的：

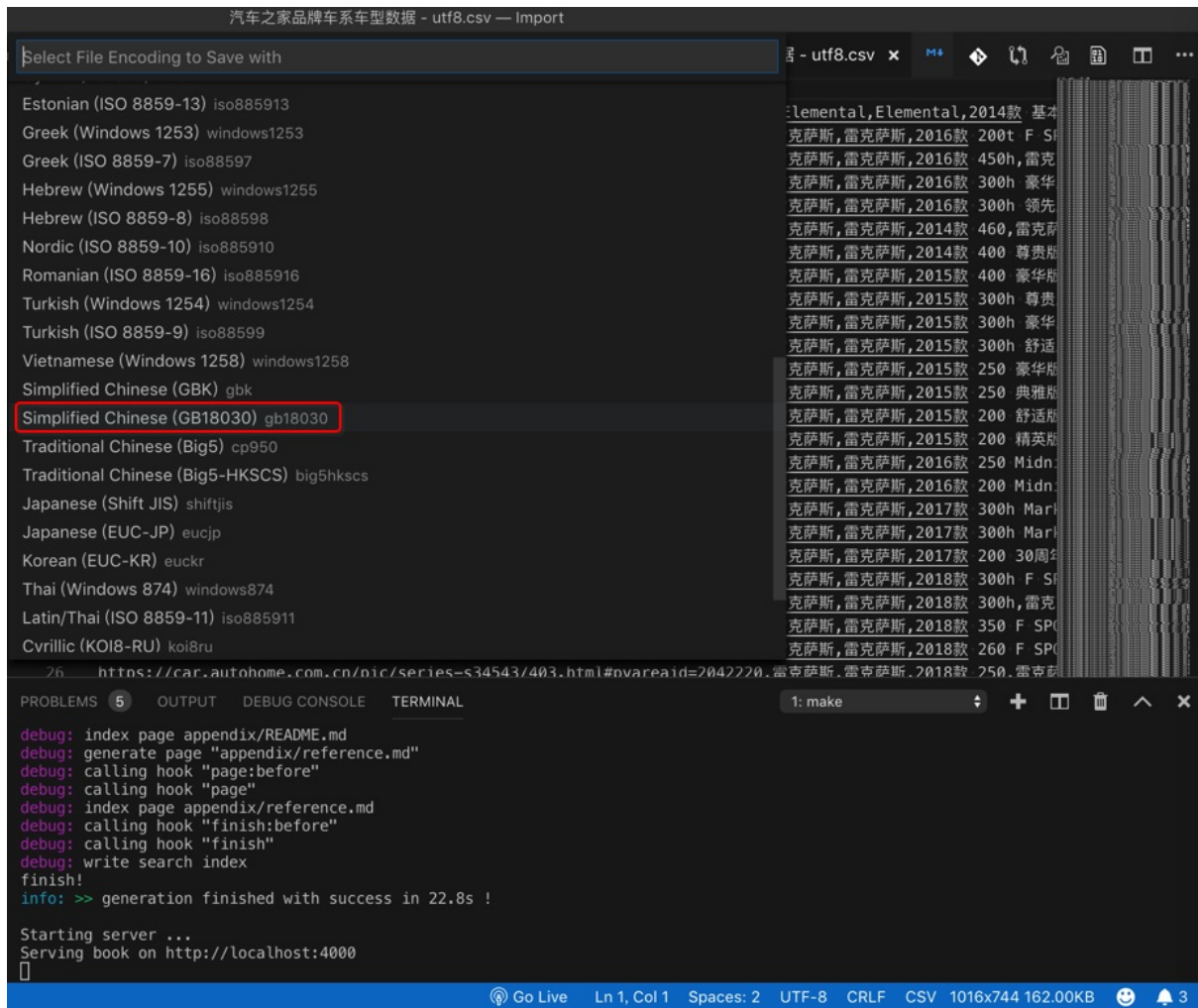
已有一个 utf-8 编码的csv文件，但是mac中用Excel打开，竟然乱码：



然后去用VSCoDe打开后，点击右下角的编码，选择 Save with Encoding：



选择中文的 (比 GBK 范畴更广的) GB18030 :



重新保存后，再去用Mac中Excel打开，就正常显示没有乱码了：

	A	B	C	D	E
1	url	品牌	子品牌	车型	车系
2	https://car.autohome.com.cn/pic/series-s19501/3548.html#pvareaid=2042220	Elemental	Elemental	2014款 基本型	Elemental RP1
3	https://car.autohome.com.cn/pic/series-s23525/261.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 200t F SPORT	雷克萨斯GS
4	https://car.autohome.com.cn/pic/series-s24494/261.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 450h	雷克萨斯GS
5	https://car.autohome.com.cn/pic/series-s25734/261.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 300h 豪华版	雷克萨斯GS
6	https://car.autohome.com.cn/pic/series-s25733/261.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 300h 领先版	雷克萨斯GS
7	https://car.autohome.com.cn/pic/series-s16768/112.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2014款 460	雷克萨斯GX
8	https://car.autohome.com.cn/pic/series-s17019/112.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2014款 400 尊贵版	雷克萨斯GX
9	https://car.autohome.com.cn/pic/series-s25071/112.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 400 豪华版	雷克萨斯GX
10	https://car.autohome.com.cn/pic/series-s22573/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 300h 尊贵版	雷克萨斯ES
11	https://car.autohome.com.cn/pic/series-s23362/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 300h 豪华版	雷克萨斯ES
12	https://car.autohome.com.cn/pic/series-s23361/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 300h 舒适版	雷克萨斯ES
13	https://car.autohome.com.cn/pic/series-s23360/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 250 豪华版	雷克萨斯ES
14	https://car.autohome.com.cn/pic/series-s23359/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 250 典雅版	雷克萨斯ES
15	https://car.autohome.com.cn/pic/series-s22601/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 200 舒适版	雷克萨斯ES
16	https://car.autohome.com.cn/pic/series-s23358/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2015款 200 精英版	雷克萨斯ES
17	https://car.autohome.com.cn/pic/series-s24897/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 250 Midnight特别版	雷克萨斯ES
18	https://car.autohome.com.cn/pic/series-s24892/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2016款 200 Midnight特别版	雷克萨斯ES
19	https://car.autohome.com.cn/pic/series-s31975/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2017款 300h Mark Levinson特别版	雷克萨斯ES
20	https://car.autohome.com.cn/pic/series-s31976/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2017款 300h Mark Levinson特别版	雷克萨斯ES
21	https://car.autohome.com.cn/pic/series-s31752/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2017款 200 30周年纪念版	雷克萨斯ES
22	https://car.autohome.com.cn/pic/series-s34545/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2018款 300h F SPORT	雷克萨斯ES
23	https://car.autohome.com.cn/pic/series-s34528/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2018款 300h	雷克萨斯ES
24	https://car.autohome.com.cn/pic/series-s34544/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2018款 350 F SPORT	雷克萨斯ES
25	https://car.autohome.com.cn/pic/series-s34371/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2018款 260 F SPORT	雷克萨斯ES
26	https://car.autohome.com.cn/pic/series-s34543/403.html#pvareaid=2042220	雷克萨斯	雷克萨斯	2018款 250	雷克萨斯ES

用VSCode检测出实际编码帮助找到问题原因

详见：

[【已解决】后台导入csv文件后解析成功但导入失败](#)

中就用到了 VSCode中识别并转换文件的编码，效果很不错。

因为可以自动根据文件的内容去推断最可能的实际的编码。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2019-03-17 22:40:54

搜索和替换

支持多种范围的 搜索 和 替换：

- 当前文件内的
- 全局的=项目范围内的

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2019-03-17 23:06:22

普通搜索

注意事项

有些东西搜不到是因为被默认排除了 node_modules

如果在VSCode中无法搜索到，确定存在的内容，则很可能是和我此处遇到的现象一样：

很多项目通过npm安装很多库到node_modules中，通过bower安装库到bower_components中，

而VSCode此处默认是通过：

```
"search.exclude": {
  "**/node_modules": true,
  "**/bower_components": true
};
```

排除了这两个文件夹到，从而导致搜索不到。

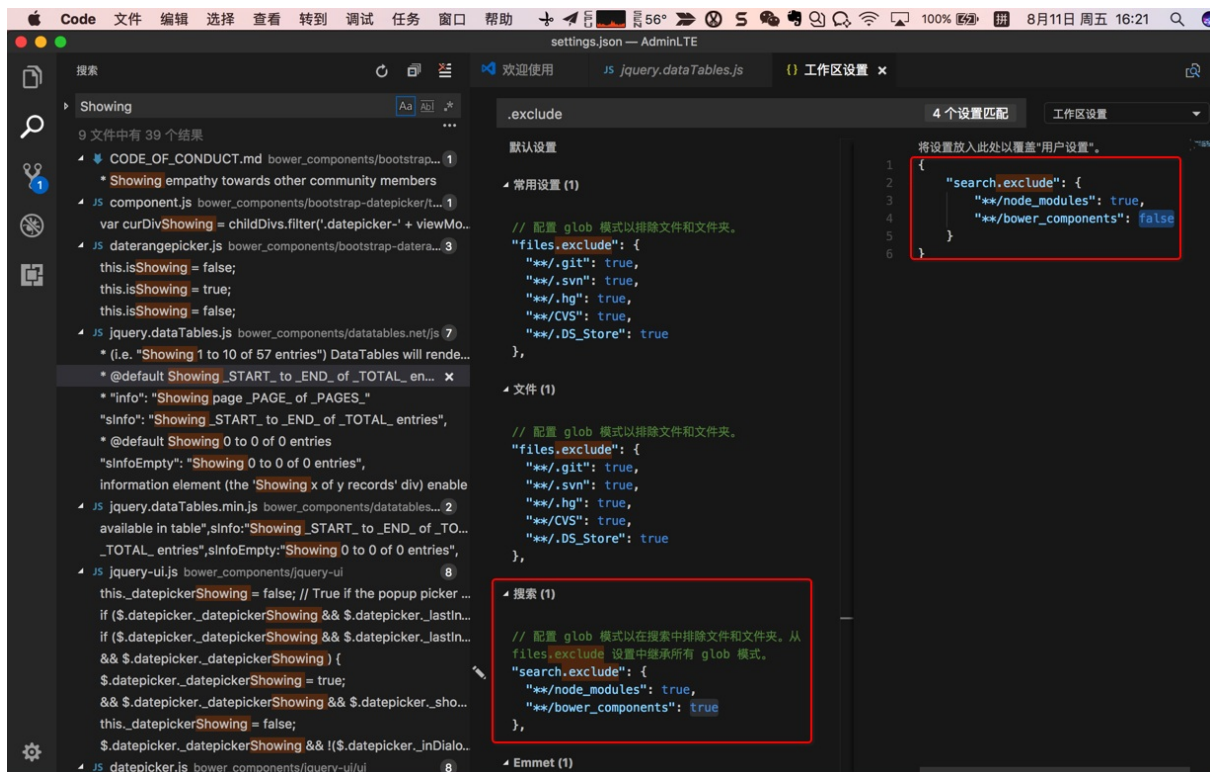
解决办法：

对于此处，取消掉bower_components的排除设置，改为false，不排除。

Code-》首选项-》设置-》搜索：

拷贝上述的配置，放到右边到用户设置（或者是工作区设置）

把bower_components改为false即可：



详见：[【已解决】VSCode中搜索不到实际上是存在的字符串](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2019-03-16

21:12:29

正则搜索

TODO:

- 整理更多的用正则搜索的例子
- 整理VSCode内部所使用的正则的库是哪一个

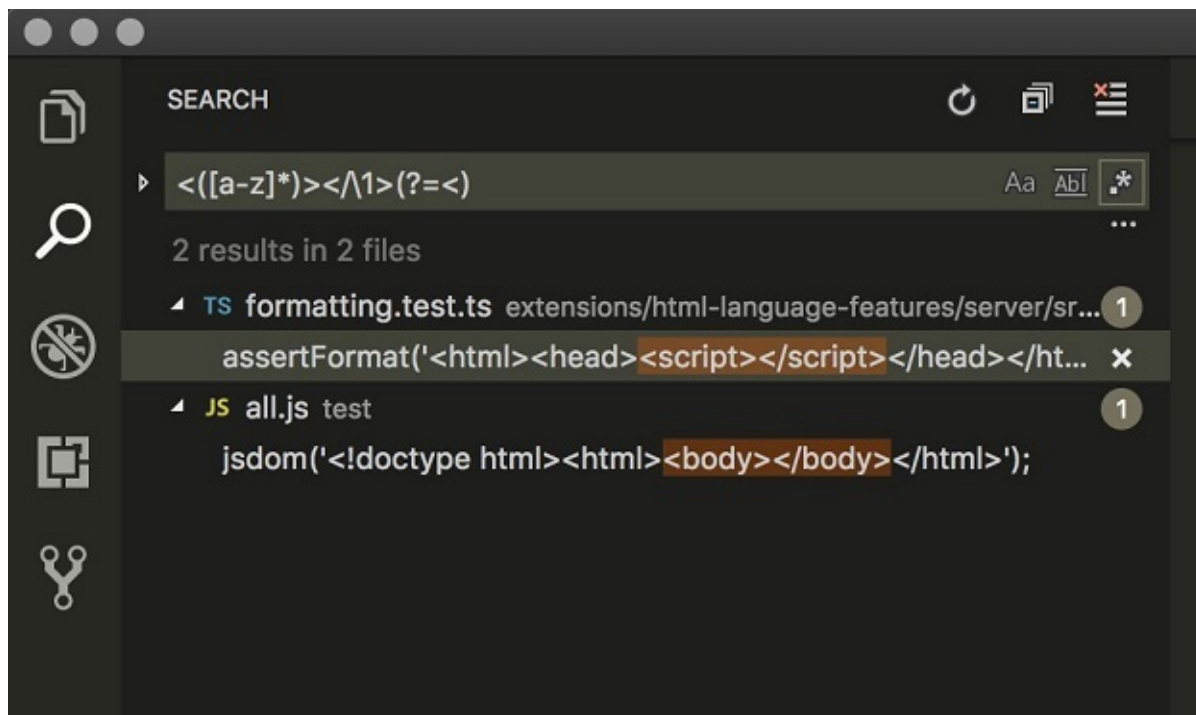
VSCode的搜索中支持正则的高级语法，比如：

- 【已解决】VSCode中如何使用正则表达式去替换且被替换中使用分组group
- 正则去替换掉特殊不可见字符：
 - 【已解决】替换掉VSCode中显示出的特殊字符：NAK

向后引用

后来新增了高级的正则搜索中的，后向引用和前向引用：[Visual Studio Code October 2018 -> Backreferences and lookahead in search](#)

效果：



以及[Multiline search](#)

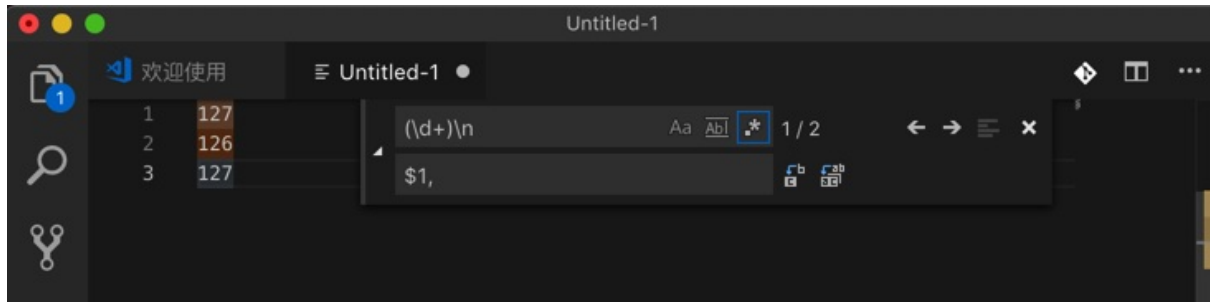
所以去更新了帖子 [Visual Studio Code 的正则匹配好用吗？](#) - 知乎的回答：

试了试其想要的效果：

```
127
126
127
```

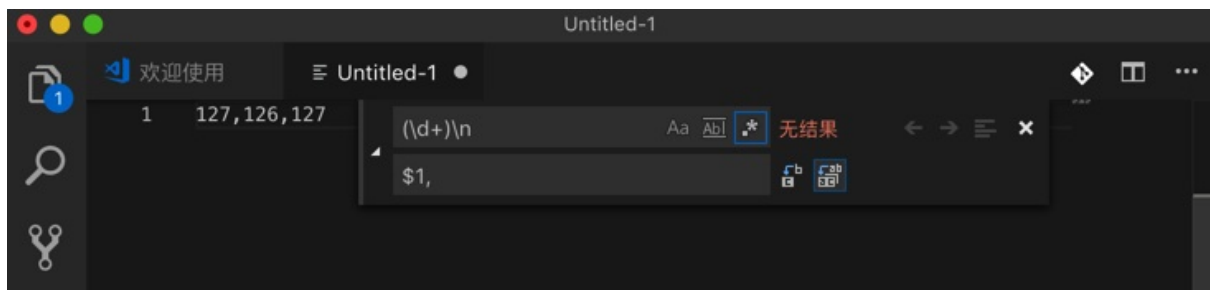
用正则替换：

```
(\d+)\n  
$1,
```



即可替换为:

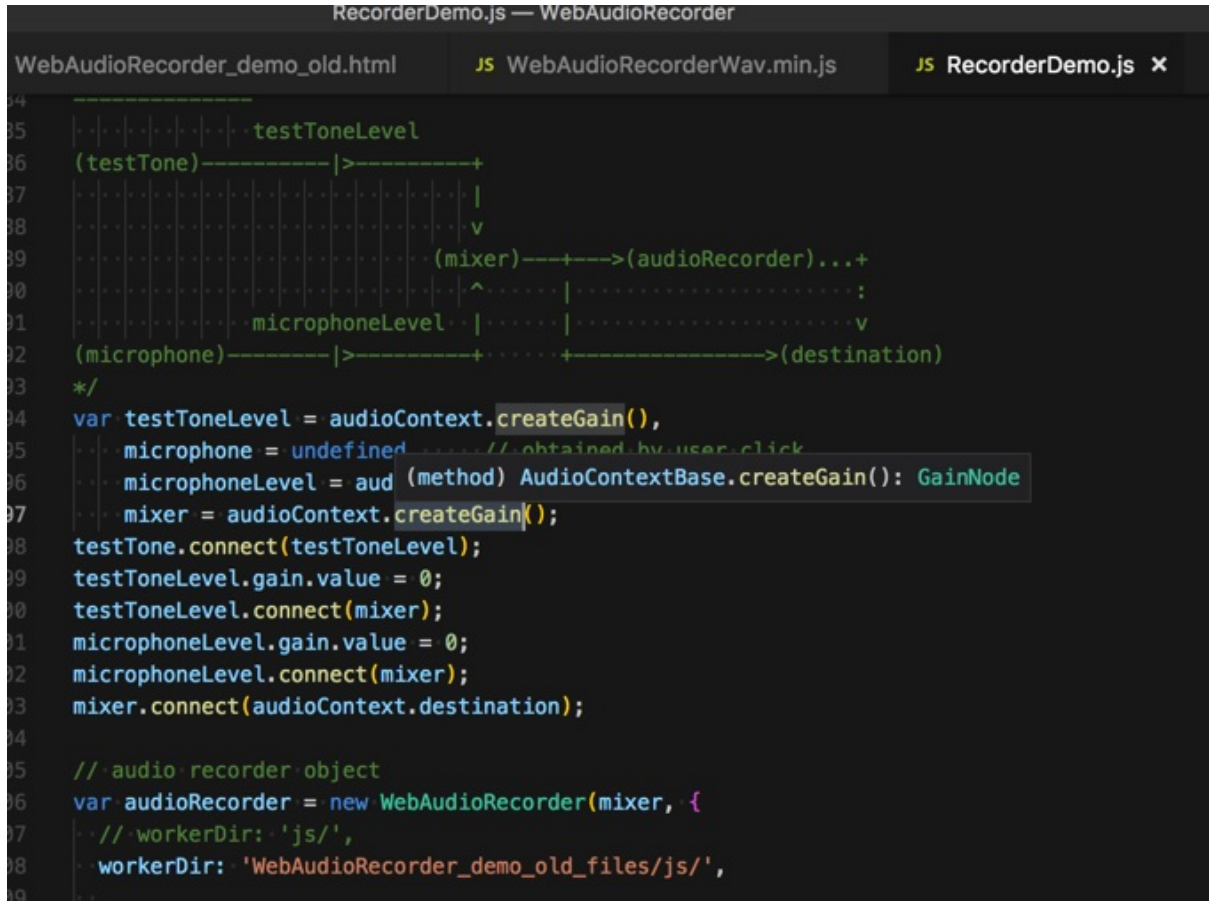
```
127,126,127
```



crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-17 23:18:58

查找函数定义

选择某个函数：

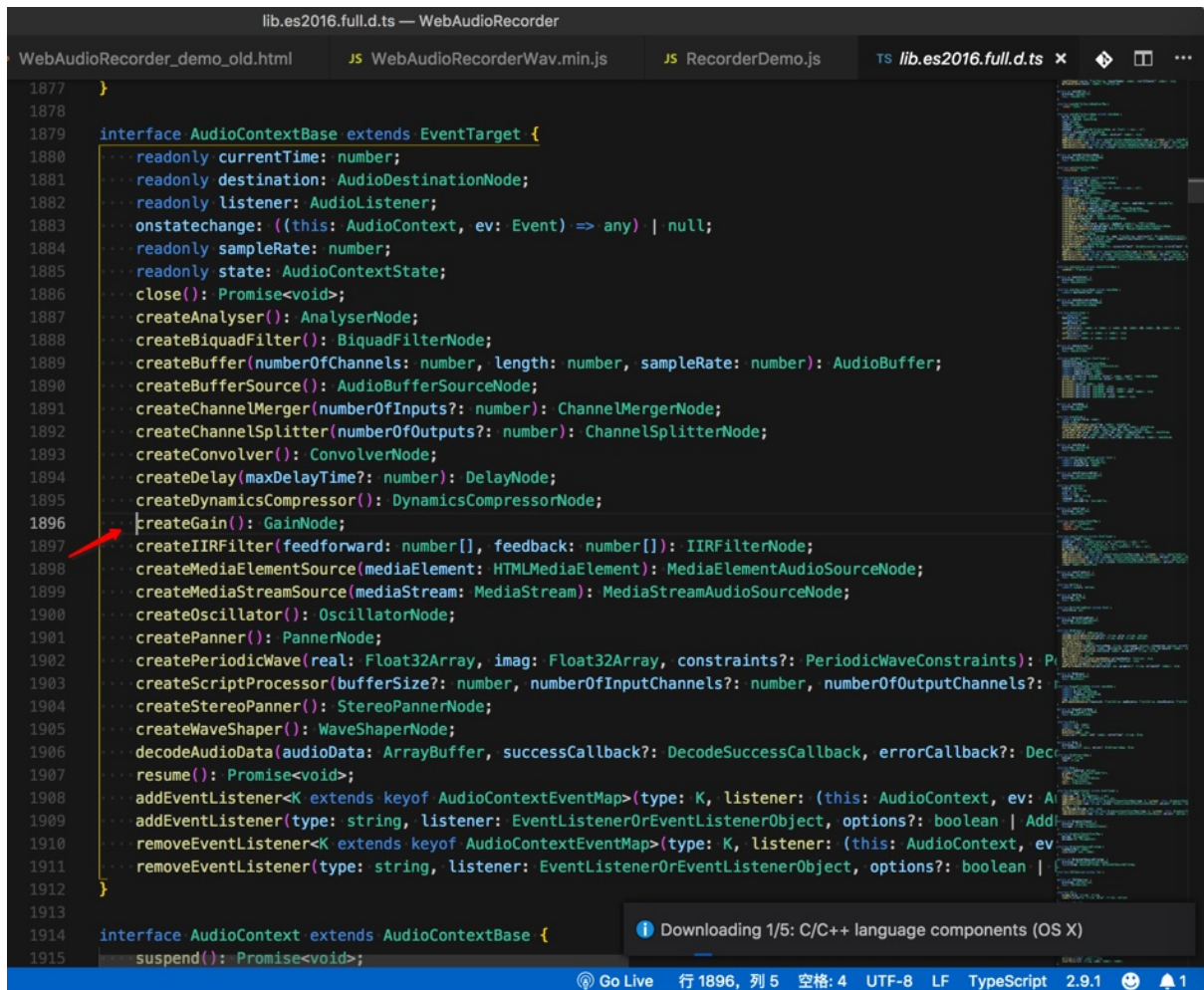


The screenshot shows a code editor window titled "RecorderDemo.js — WebAudioRecorder". The editor has three tabs: "WebAudioRecorder_demo_old.html", "JS WebAudioRecorderWav.min.js", and "JS RecorderDemo.js x". The code in the active tab is as follows:

```
34 .....
35 ..... testToneLevel
36 (testTone)-----|>-----+
37 .....|
38 .....v
39 .....(mixer)---+--->(audioRecorder)...+
40 .....^.....|.....:
41 ..... microphoneLevel.....|.....v
42 (microphone)-----|>-----+.....+----->(destination)
43 */
44 var testToneLevel = audioContext.createGain(),
45     microphone = undefined // obtained by user click
46     microphoneLevel = aud (method) AudioContextBase.createGain(): GainNode
47     mixer = audioContext.createGain();
48 testTone.connect(testToneLevel);
49 testToneLevel.gain.value = 0;
50 testToneLevel.connect(mixer);
51 microphoneLevel.gain.value = 0;
52 microphoneLevel.connect(mixer);
53 mixer.connect(audioContext.destination);
54
55 // audio recorder object
56 var audioRecorder = new WebAudioRecorder(mixer, {
57     // workerDir: 'js/',
58     workerDir: 'WebAudioRecorder_demo_old_files/js/',
59 ..
```

The function definition `audioContext.createGain()` on line 44 is highlighted in yellow. A tooltip is visible over the `createGain()` call on line 46, showing the signature `createGain(): GainNode`.

Command + 鼠标点击，可以找到此处js的函数定义：



The screenshot shows a code editor with a dark theme. The main window displays a TypeScript interface definition for `AudioContextBase` extending `EventTarget`. The interface includes various methods for creating and managing audio nodes and contexts. A red arrow points to the `createGain()` method. The editor has several tabs open, including `lib.es2016.full.d.ts`. The status bar at the bottom shows "Go Live" and other editor settings.

```
1877 }
1878
1879 interface AudioContextBase extends EventTarget {
1880     readonly currentTime: number;
1881     readonly destination: AudioDestinationNode;
1882     readonly listener: AudioListener;
1883     onstatechange: ((this: AudioContext, ev: Event) => any) | null;
1884     readonly sampleRate: number;
1885     readonly state: AudioContextState;
1886     close(): Promise<void>;
1887     createAnalyser(): AnalyserNode;
1888     createBiquadFilter(): BiquadFilterNode;
1889     createBuffer(numberOfChannels: number, length: number, sampleRate: number): AudioBuffer;
1890     createBufferSource(): AudioBufferSourceNode;
1891     createChannelMerger(numberOfInputs?: number): ChannelMergerNode;
1892     createChannelSplitter(numberOfOutputs?: number): ChannelSplitterNode;
1893     createConvolver(): ConvolverNode;
1894     createDelay(maxDelayTime?: number): DelayNode;
1895     createDynamicsCompressor(): DynamicsCompressorNode;
1896     createGain(): GainNode;
1897     createIIRFilter(feedforward: number[], feedback: number[]): IIRFilterNode;
1898     createMediaElementSource(mediaElement: HTMLMediaElement): MediaElementAudioSourceNode;
1899     createMediaStreamSource(mediaStream: MediaStream): MediaStreamAudioSourceNode;
1900     createOscillator(): OscillatorNode;
1901     createPanner(): PannerNode;
1902     createPeriodicWave(real: Float32Array, imag: Float32Array, constraints?: PeriodicWaveConstraints): P
1903     createScriptProcessor(bufferSize?: number, numberOfInputChannels?: number, numberOfOutputChannels?:
1904     createStereoPanner(): StereoPannerNode;
1905     createWaveShaper(): WaveShaperNode;
1906     decodeAudioData(audioData: ArrayBuffer, successCallback?: DecodeSuccessCallback, errorCallback?: Dec
1907     resume(): Promise<void>;
1908     addEventListener<K extends keyof AudioContextEventMap>(type: K, listener: (this: AudioContext, ev: A
1909     addEventListener(type: string, listener: EventListenerOrEventListenerObject, options?: boolean | Add
1910     removeEventListener<K extends keyof AudioContextEventMap>(type: K, listener: (this: AudioContext, ev
1911     removeEventListener(type: string, listener: EventListenerOrEventListenerObject, options?: boolean |
1912 }
1913
1914 interface AudioContext extends AudioContextBase {
1915     suspend(): Promise<void>;
```

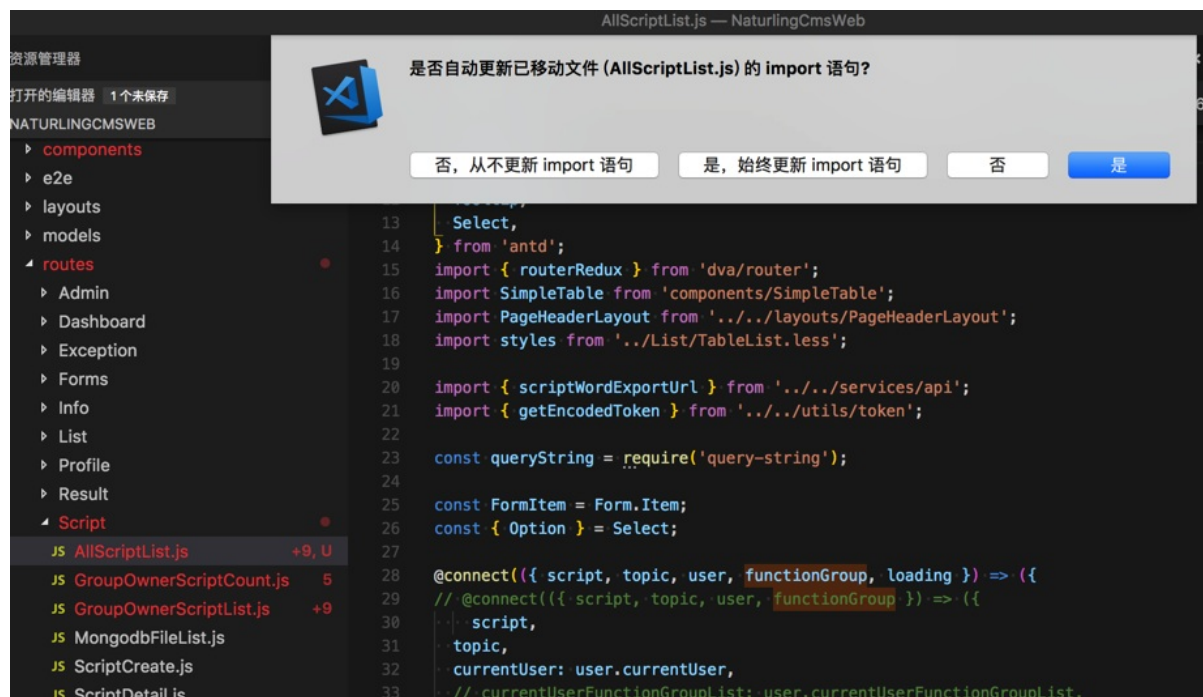
效果不错。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-17 11:35:41

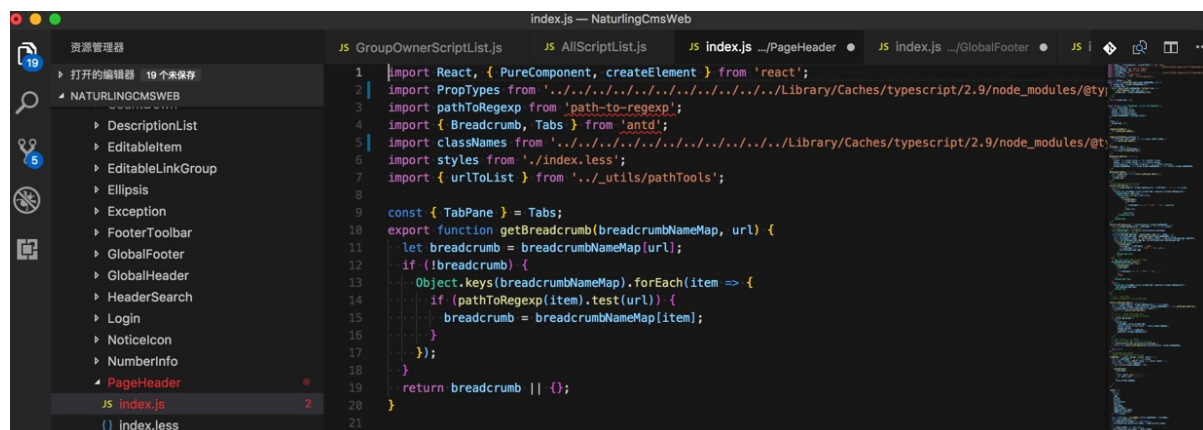
VSCode的智能好用之处

自动更新import语句

当从已有的一个文件，复制并改名后，VSCode可以自动检测出来，提示是否需要更新import语句



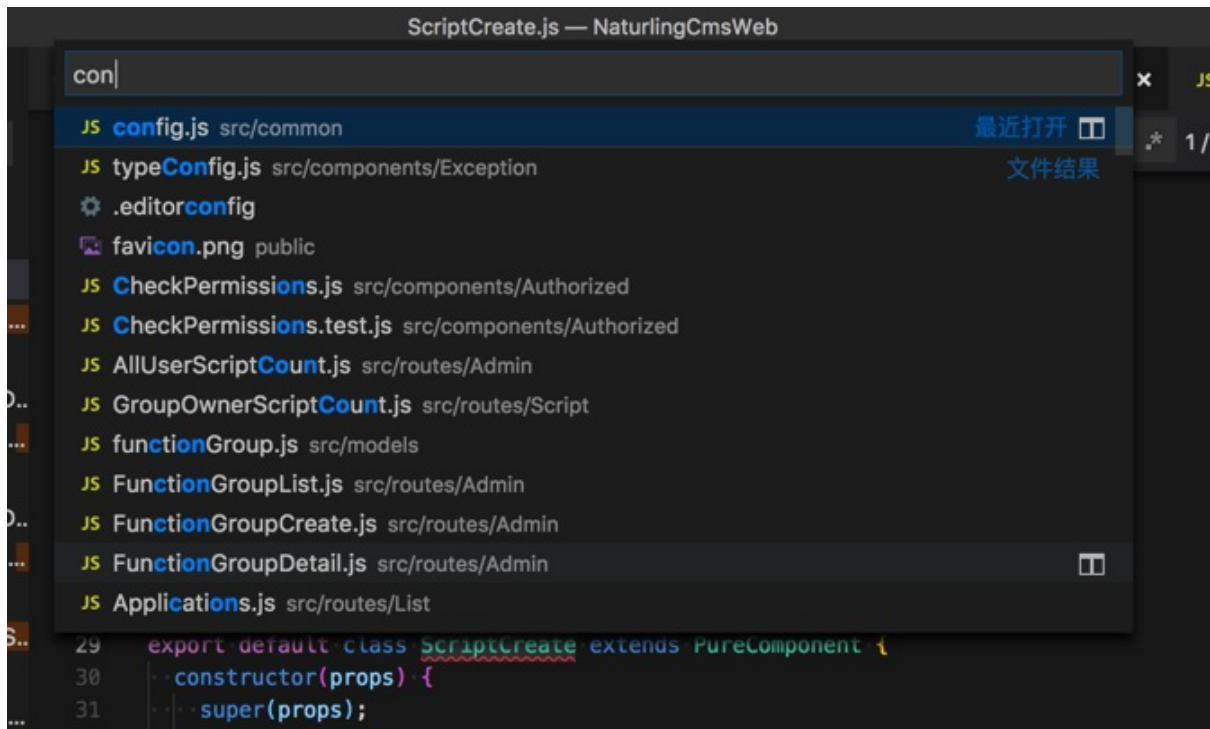
即可自动更新import语句:



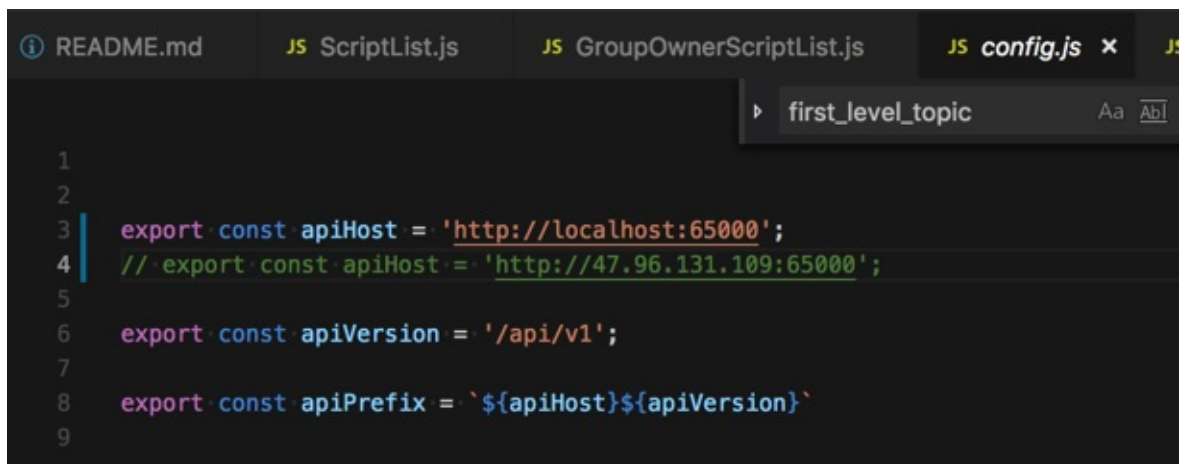
不过，此处更新后的内容不是我要的，竟然把本来正常的都改错误了，所以还要自己去改回来。以后慎用这个自动修改import的功能。

快速跳转文件

Command + P 后，输入（部分）文件名（支持模糊搜索）：



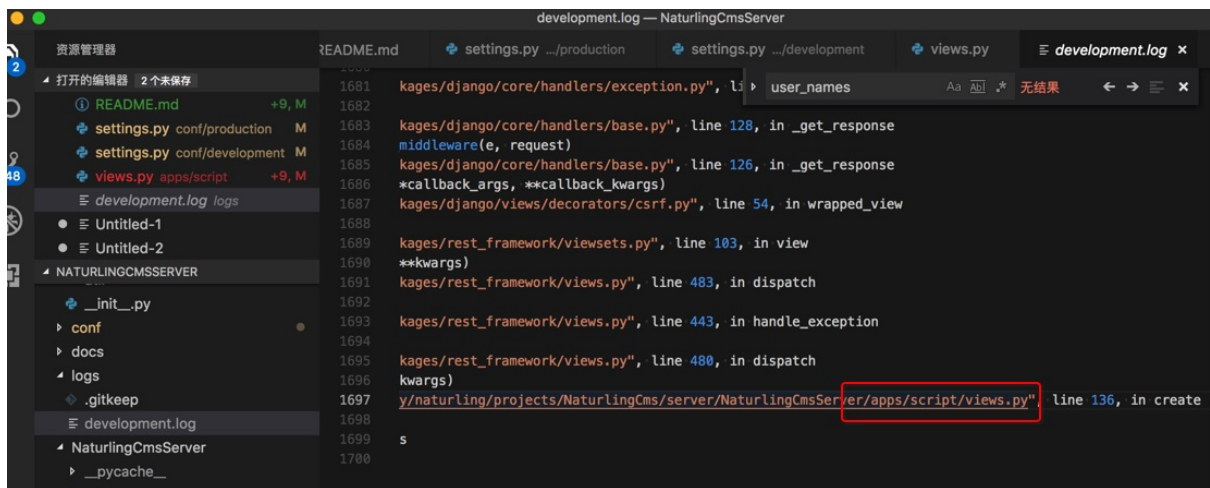
选中回车即可跳转文件：



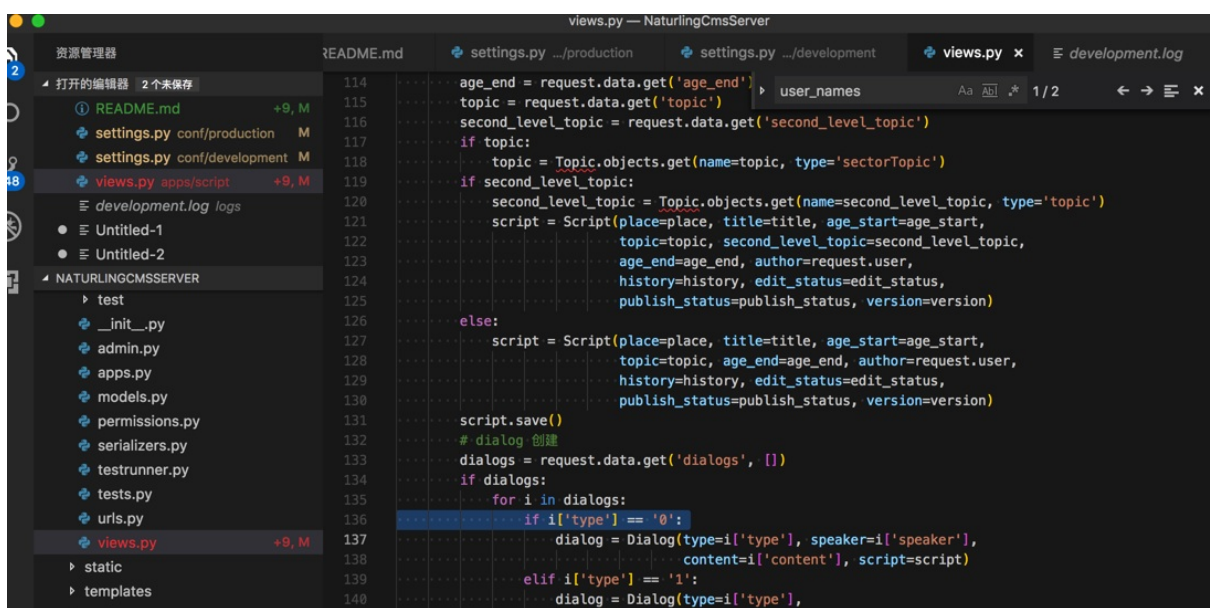
log日志中点击文件路径可以跳转到该文件

比如调试期间出错了，点击对应log错误日志中的文件路径：

```
File "/usr/local/lib/python3.6/site-packages/rest_framework/views.py", line 483, in dispatch
response = self.handle_exception(exc)
File "/usr/local/lib/python3.6/site-packages/rest_framework/views.py", line 443, in handle_exception
self.raise_uncaught_exception(exc)
File "/usr/local/lib/python3.6/site-packages/rest_framework/views.py", line 480, in dispatch
response = handler(request, *args, **kwargs)
File "/Users/crifan/dev/dev_root/company/naturling/projects/xxx/server/xxx/apps/script/views.py", line 136, in create
if i['type'] == '0':
TypeError: string indices must be integers
```



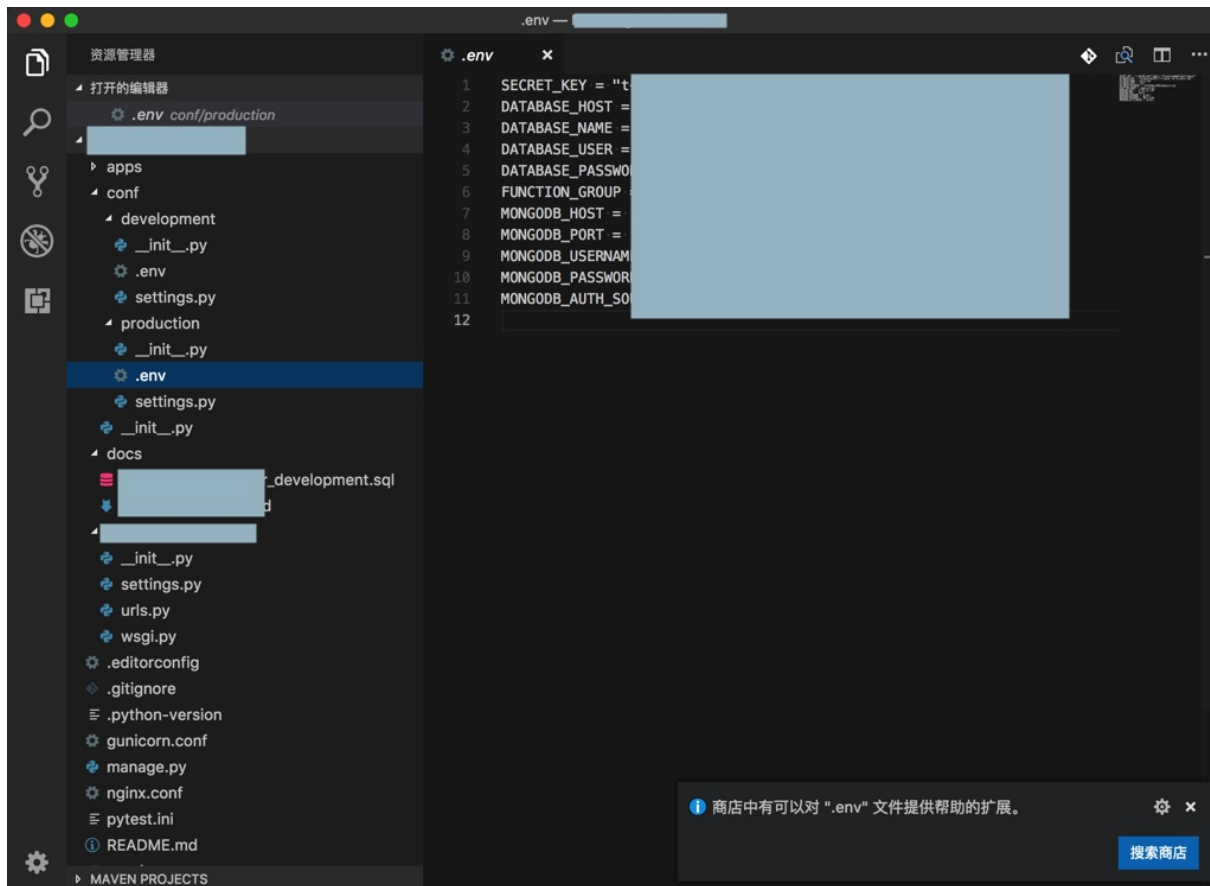
可以跳转到对应的文件：



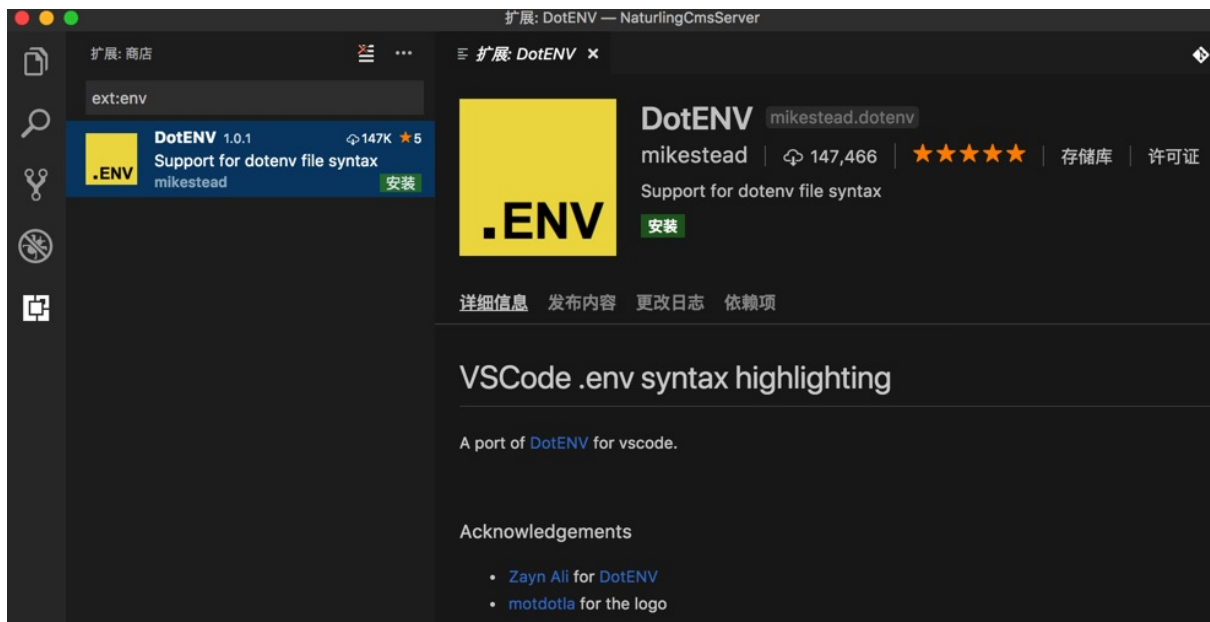
方便调试。

提示安装支持相应文件的插件

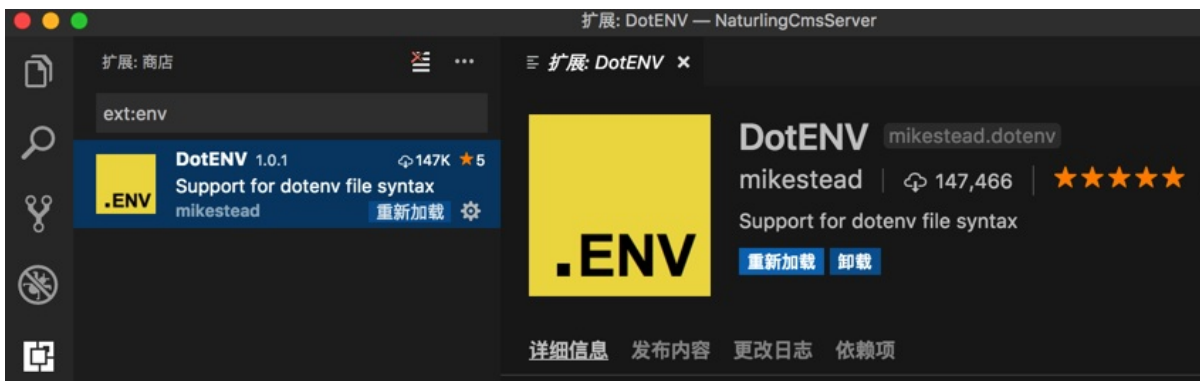
首次打开 .env ，则提示是否要安装 .env 的插件，点击 搜索商店：



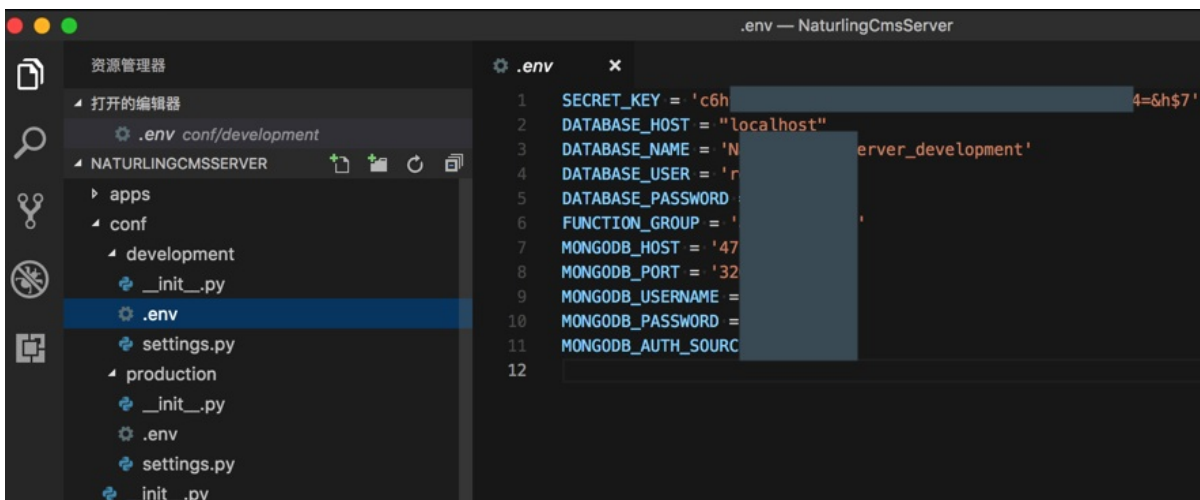
然后点击安装插件:



安装后, 重启加载:

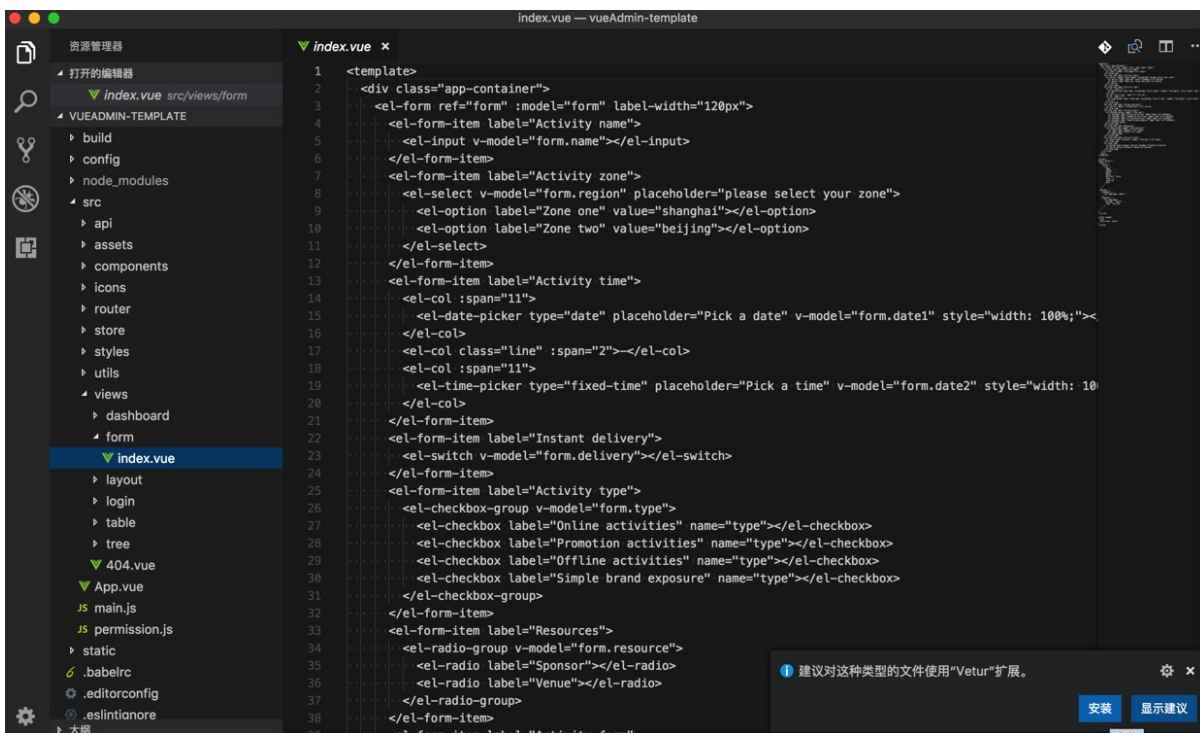


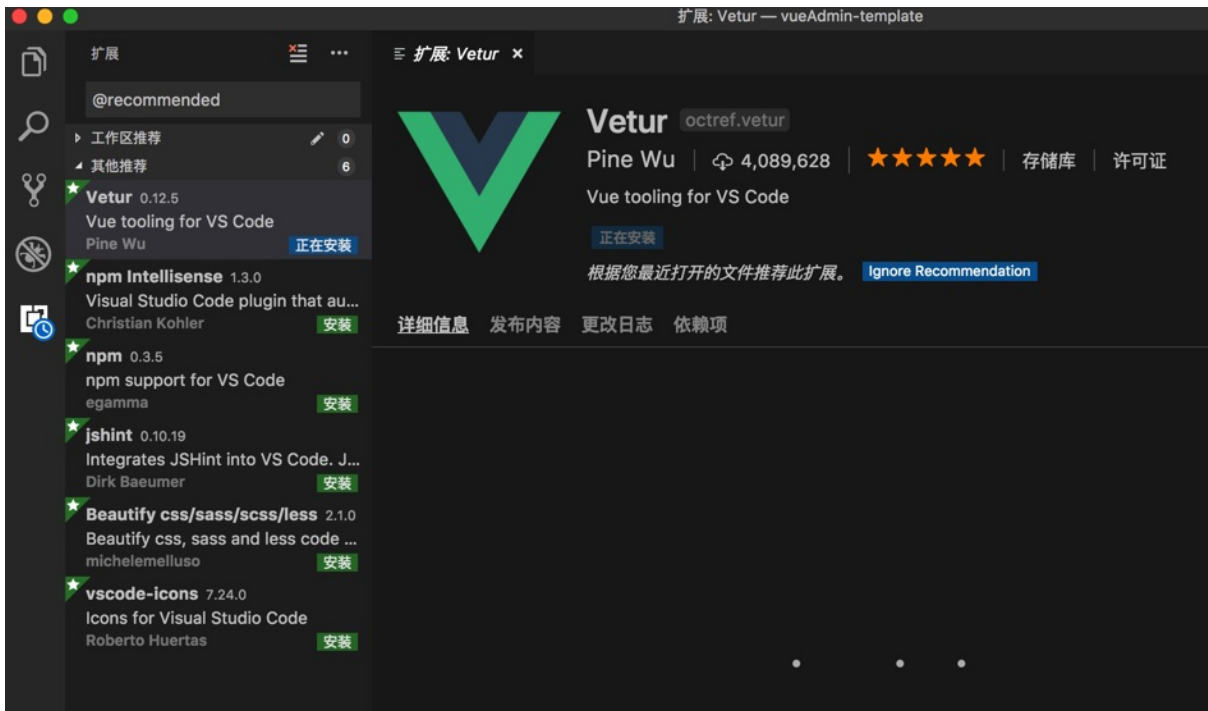
然后.env就可以语法高亮了:



另外类似的情况还有:

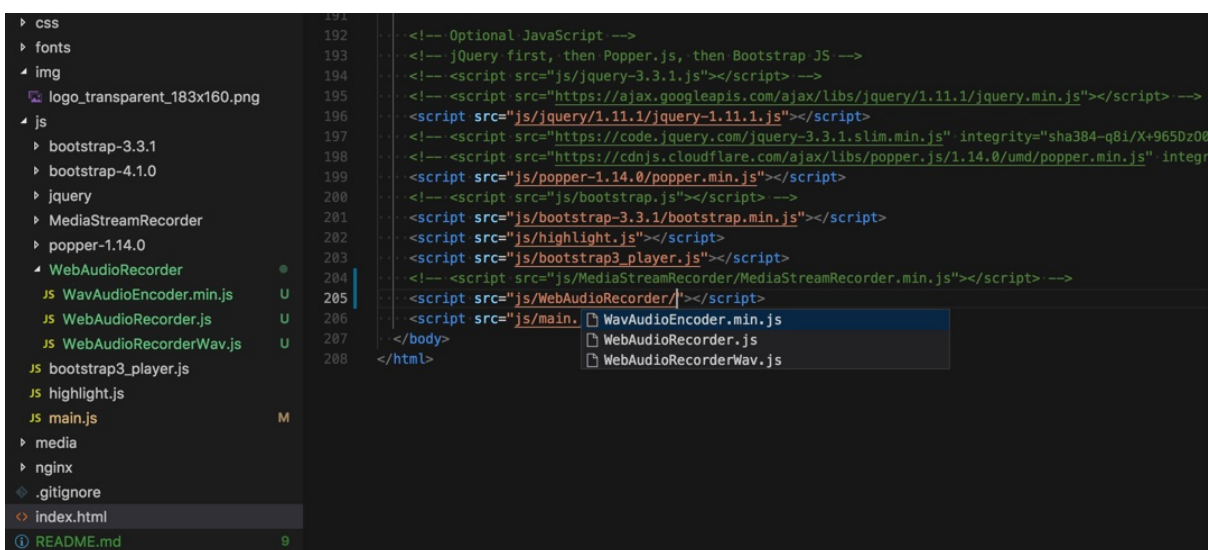
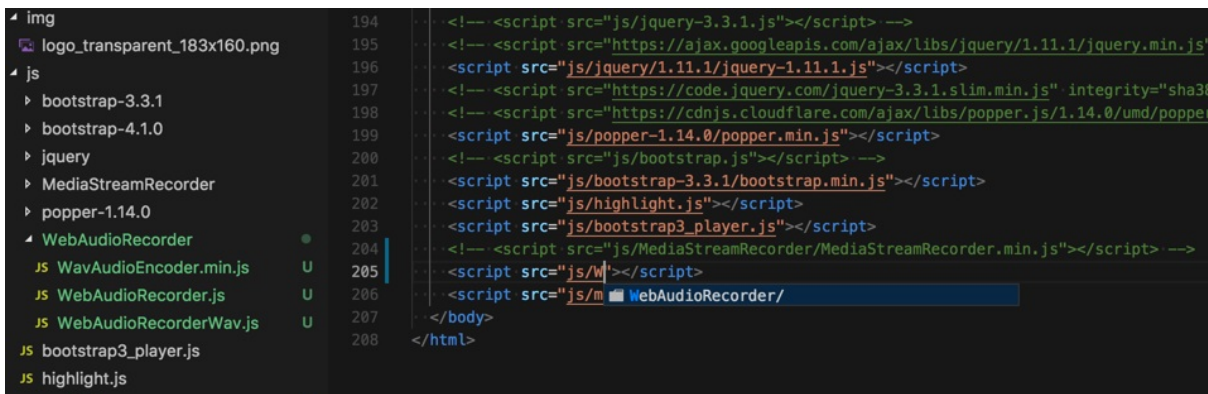
首次打开vue提示安装对应的插件:





根据路径动态提示文件

刚新建个文件夹，加入了几个js文件后，然后再去html中输入路径后，即可动态匹配路径和文件：



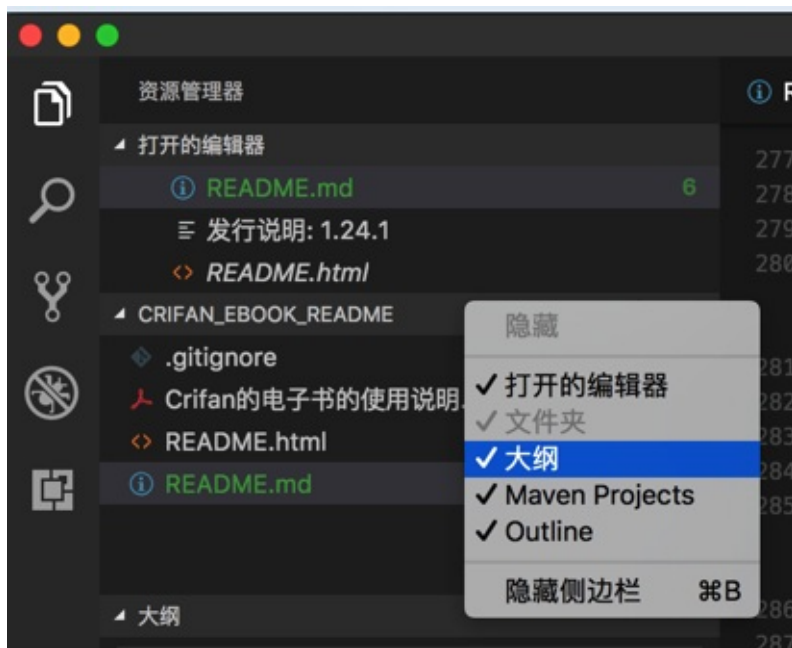
很是智能和贴心。

显示大纲

VSCode 1.24版本支持支持 大纲 = Outline = 目录：

Preview: Outline view - Symbol tree outline and navigation for your projects.

右键项目条，选中 大纲：



点击大纲中的某行后，可以跳转到对应位置，比如：

- Markdown：

-
- Html

◦

未使用变量检测

VSCode 1.24版本支持支持自动检测未使用的变量的提示：

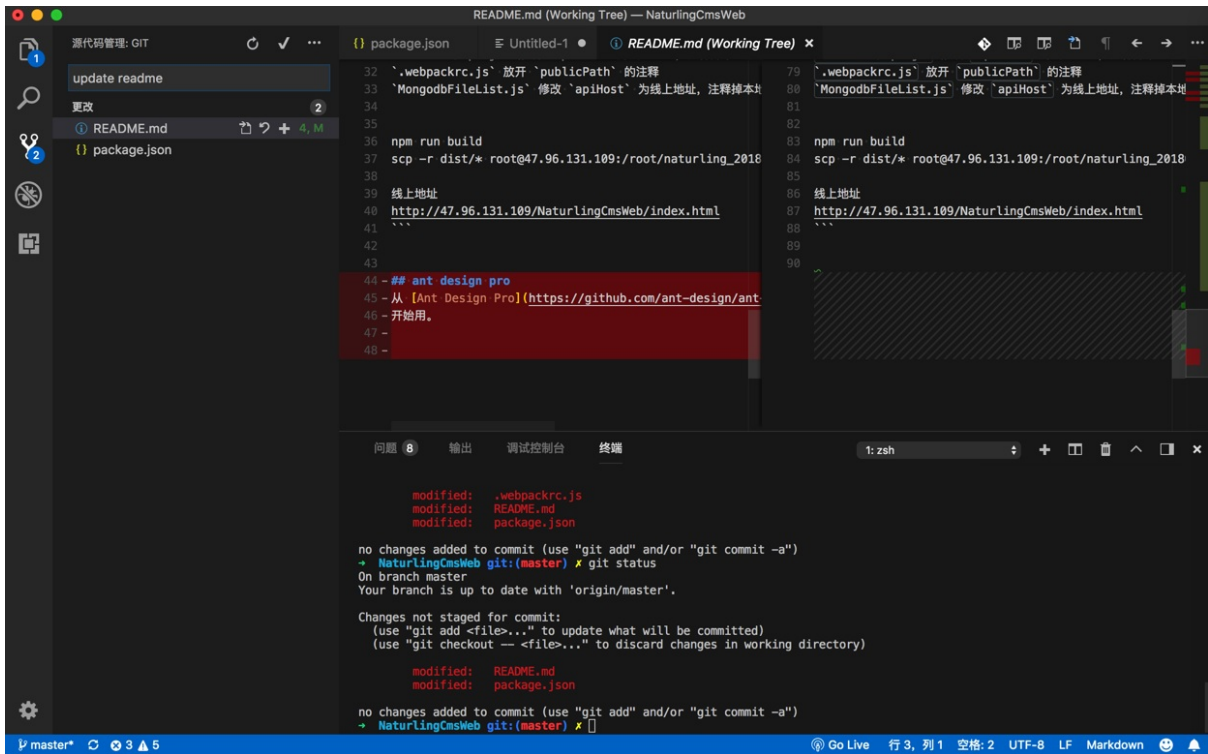
Unused variable detection - Unused variables are greyed-out in your JavaScript/TypeScript files

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2019-03-17

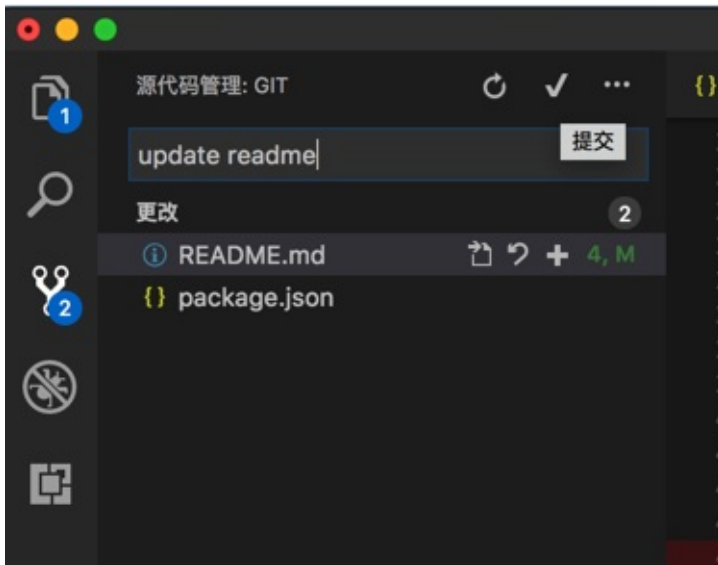
12:08:56

Git代码管理

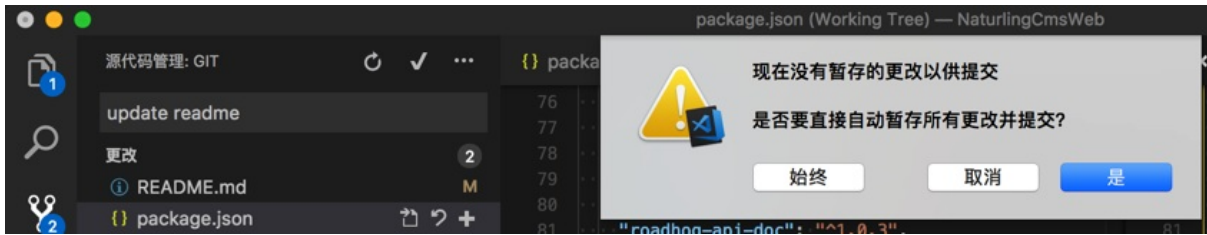
VSCode中用git管理代码仓库



然后尝试用VSCode中的git界面去提交代码:



会有提示:



不过去 `git add` 后:

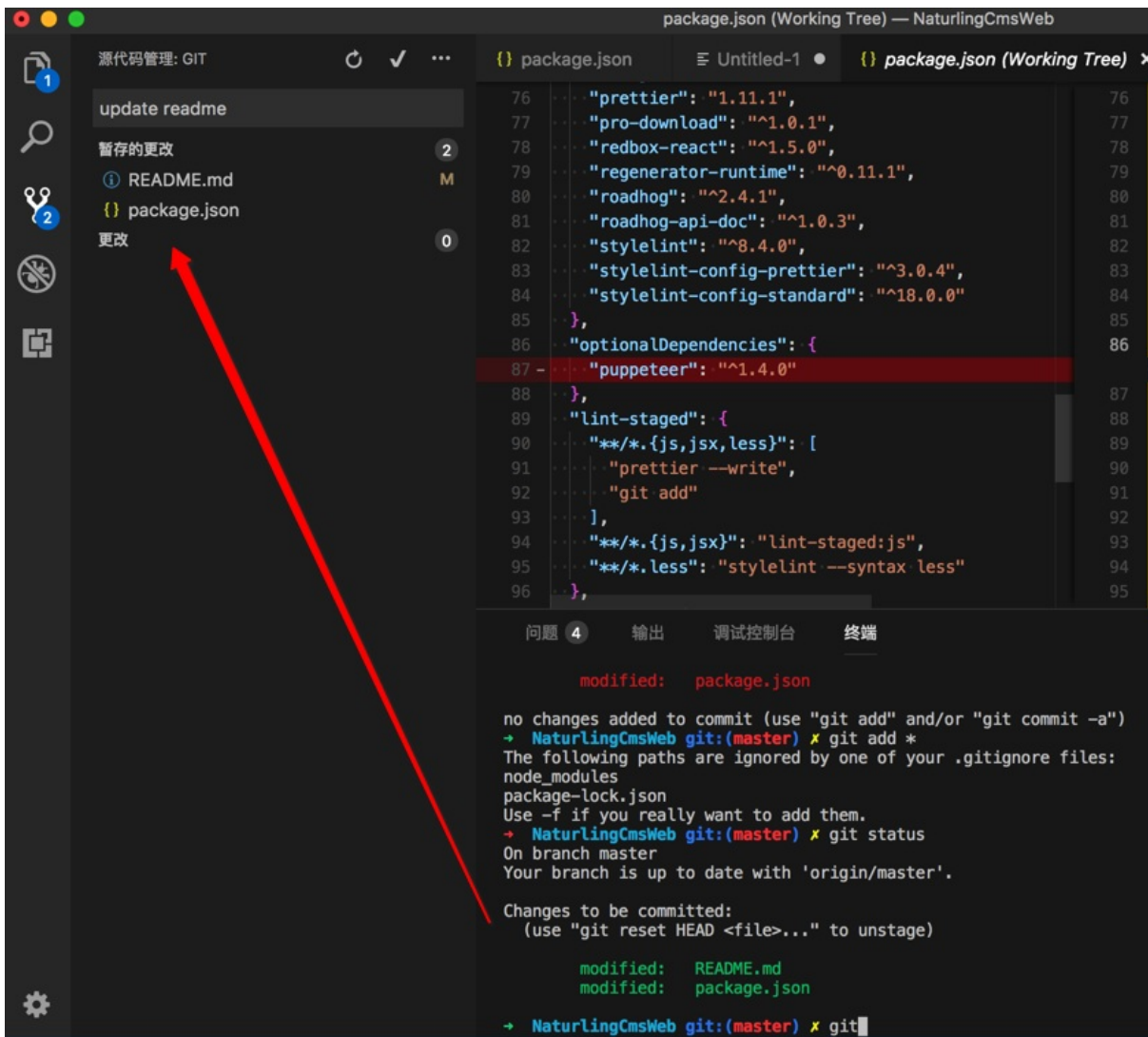
```

→ NaturlingCmsWeb git:(master) X git add *
The following paths are ignored by one of your .gitignore files:
node_modules
package-lock.json
Use -f if you really want to add them.
→ NaturlingCmsWeb git:(master) X git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md
        modified:   package.json
  
```

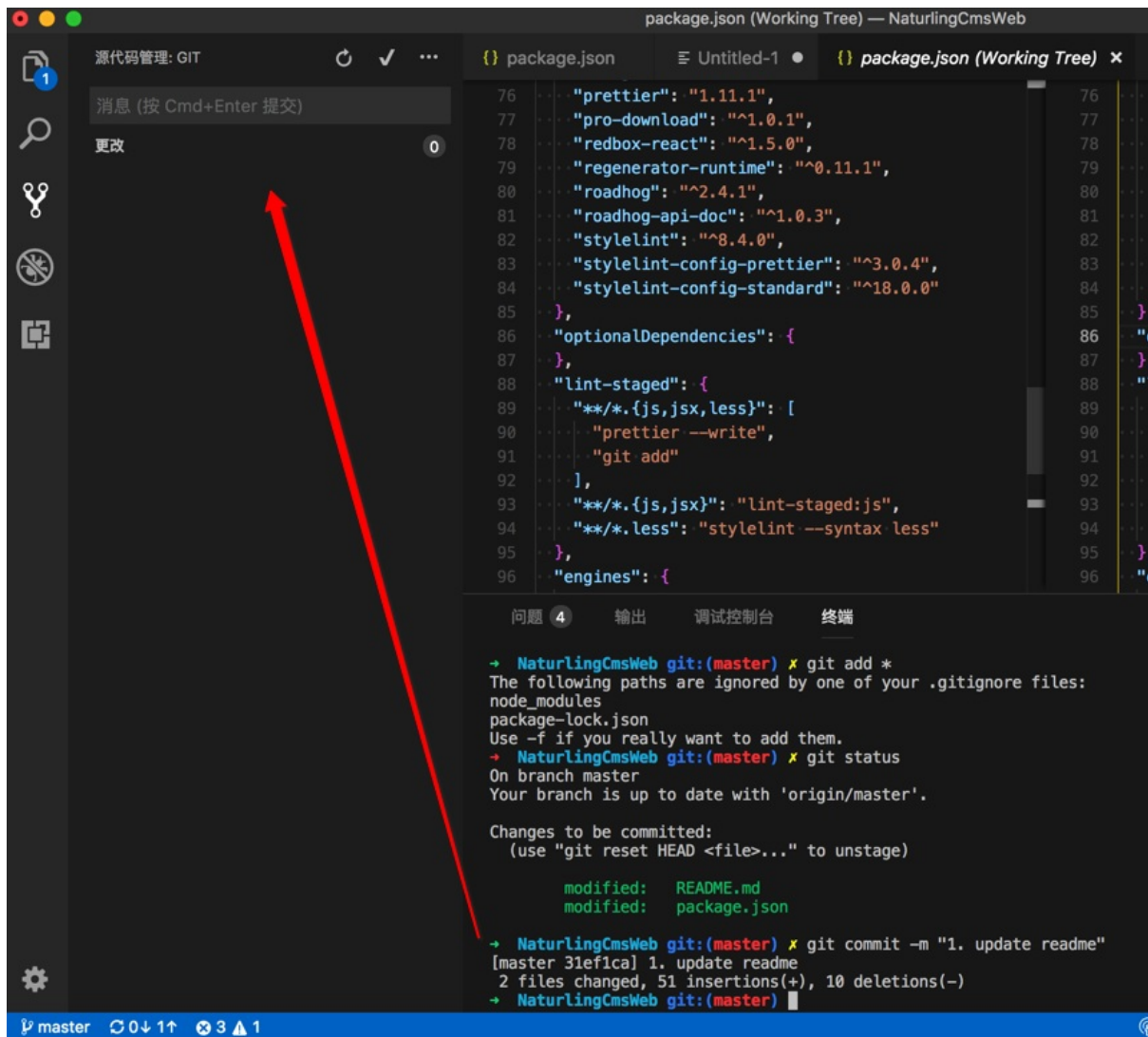
git就显示出对应的已经添加了的改动的文件了:



暂存的更改。而 更改 中就是空了。

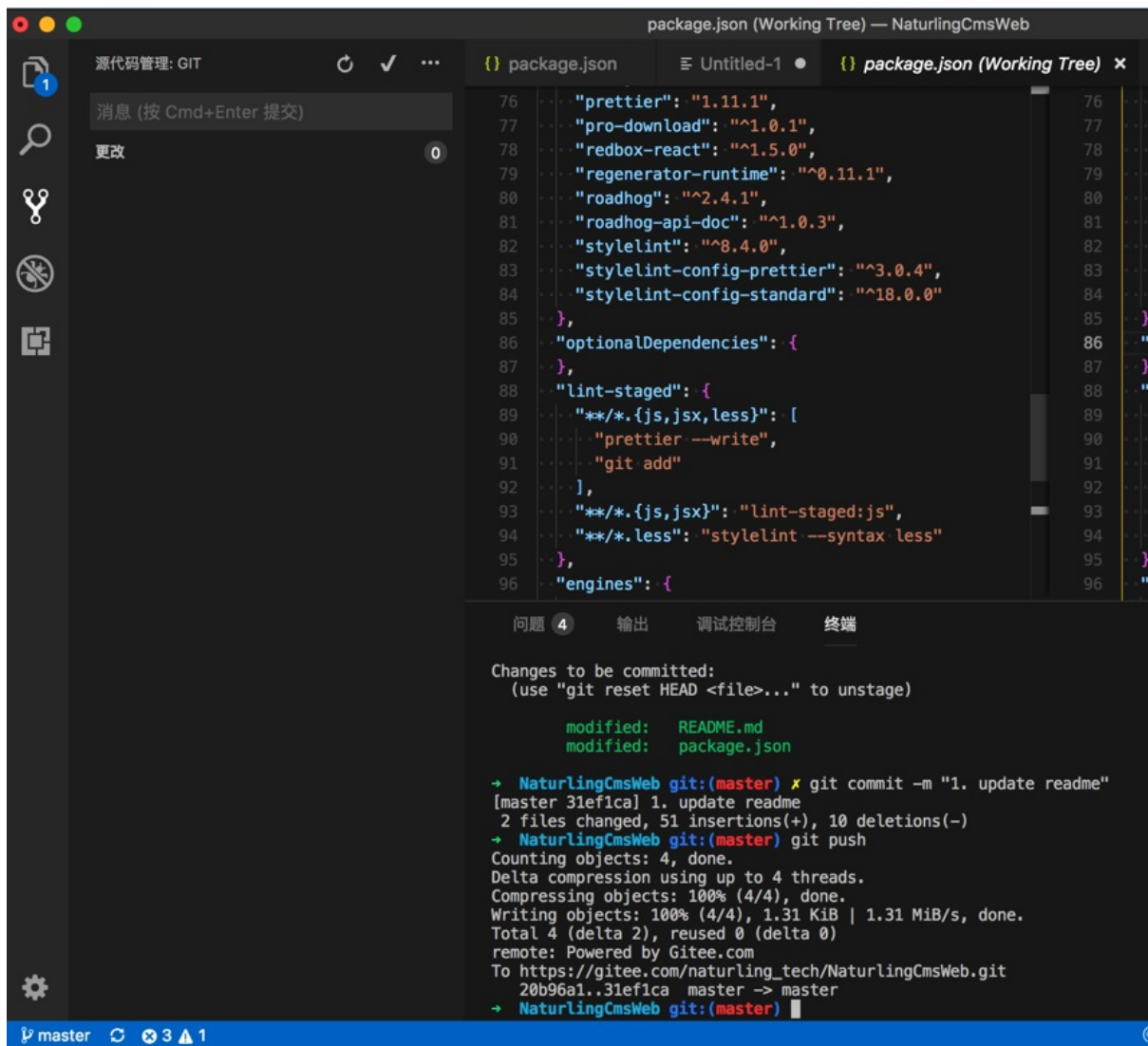
-> 说明：暂存的更改，就是add进去的，本地临时保存的改动

```
→ NaturlingCmsWeb git:(master) ✕ git commit -m "1. update readme"
[master 31ef1ca] 1. update readme
2 files changed, 51 insertions(+), 10 deletions(-)
```



然后再去push:

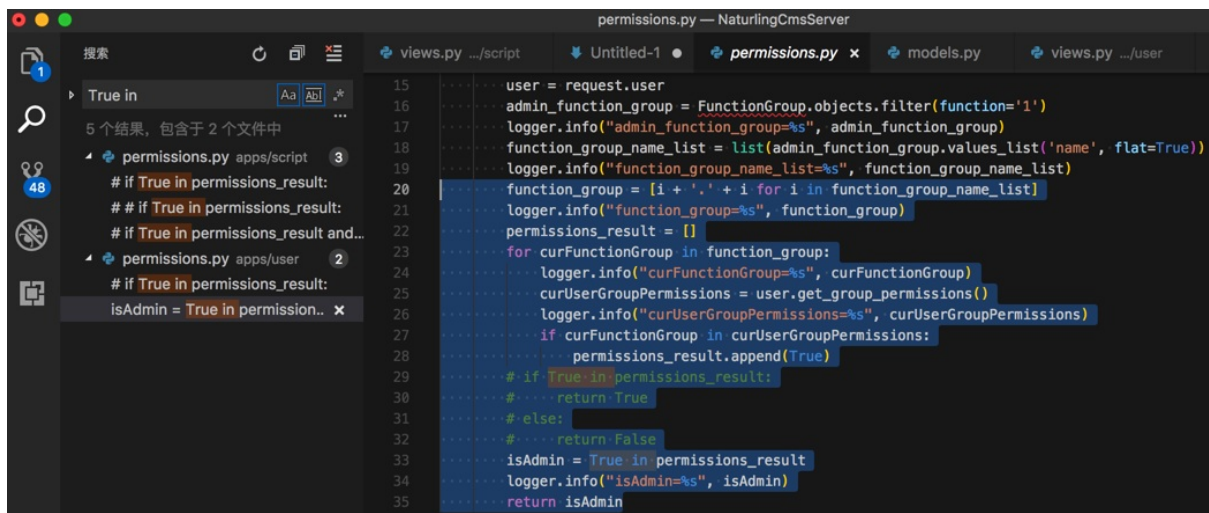
```
→ NaturlingCmsWeb git:(master) git push
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.31 KiB | 1.31 MiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Powered by Gitee.com
To https://gitee.com/naturling_tech/NaturlingCmsWeb.git
20b96a1..31ef1ca master -> master
```

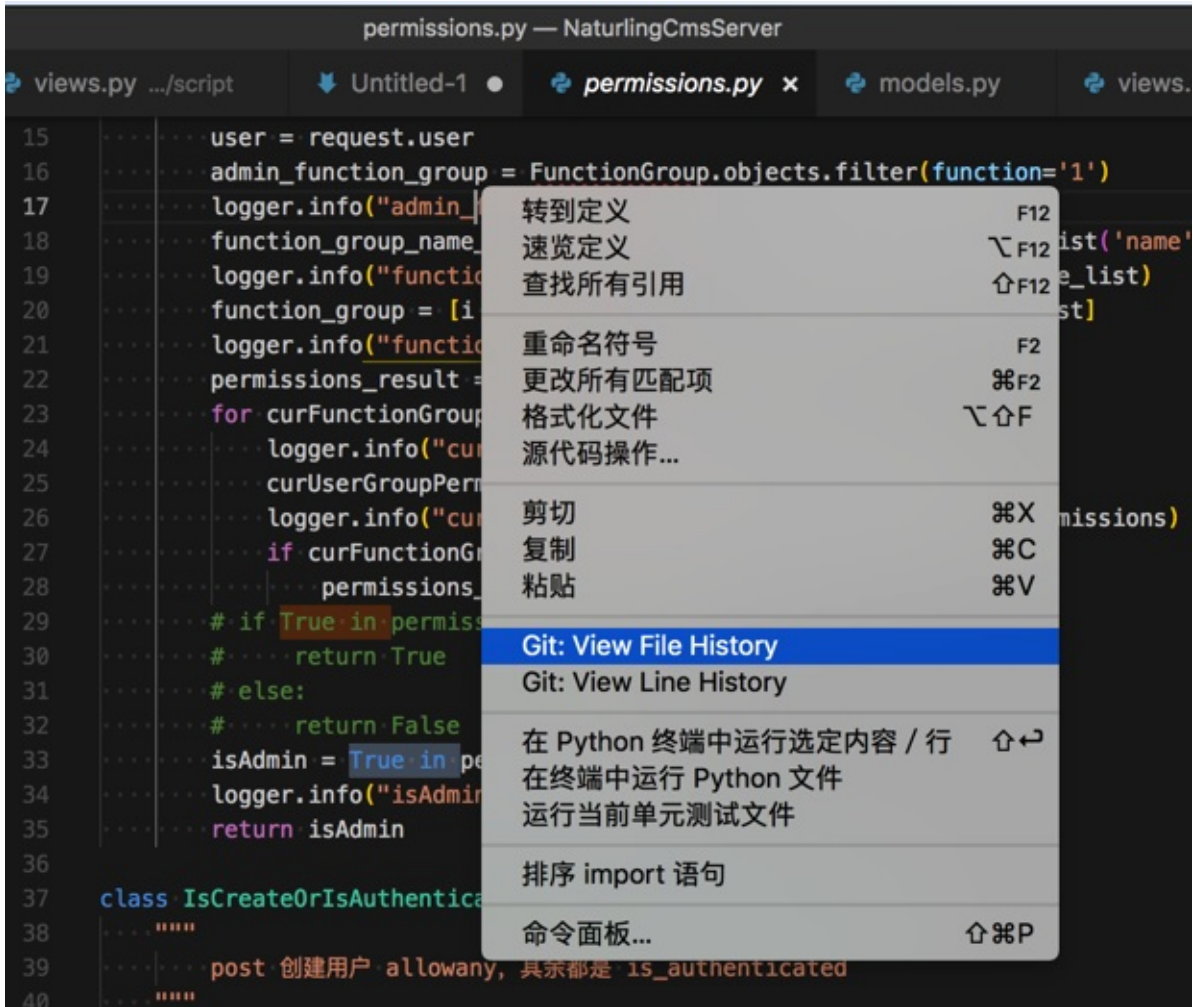
查看文件历史版本和改动差异

如果需要对比之前某次的文件代码和当前最新代码有何改动，可以通过 `File History` 文件历史，得到我们要的效果：

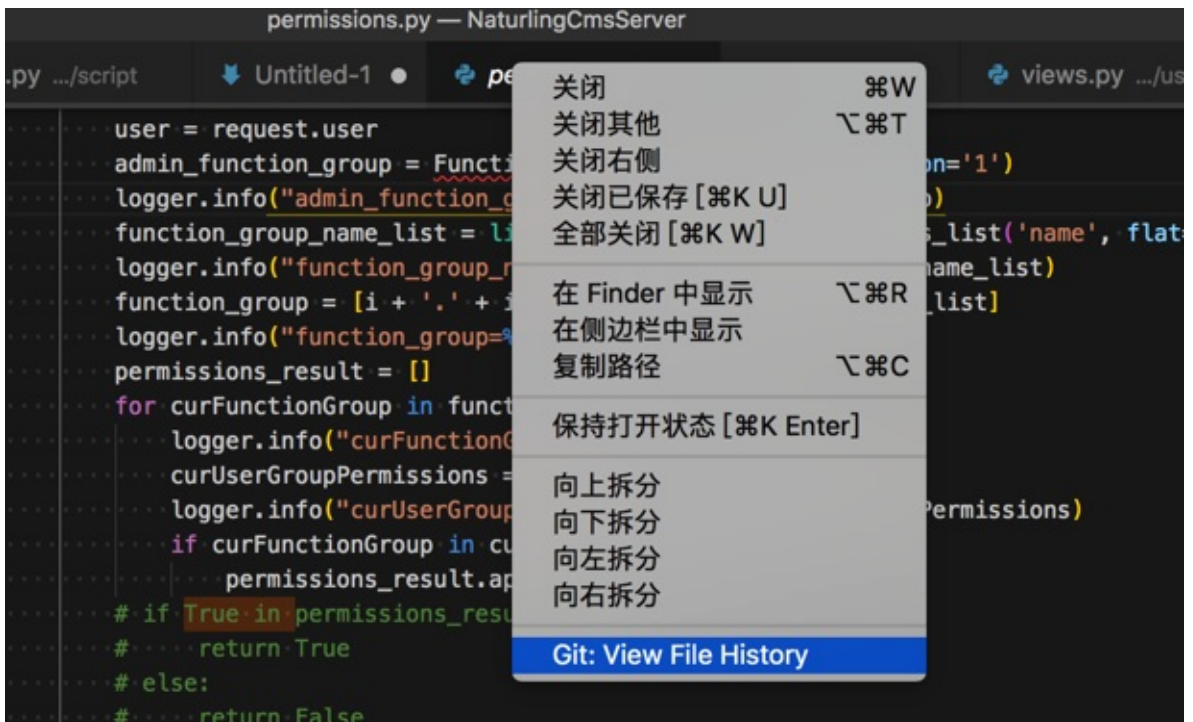
比如想要对比这段代码和之前的写法有何不同：



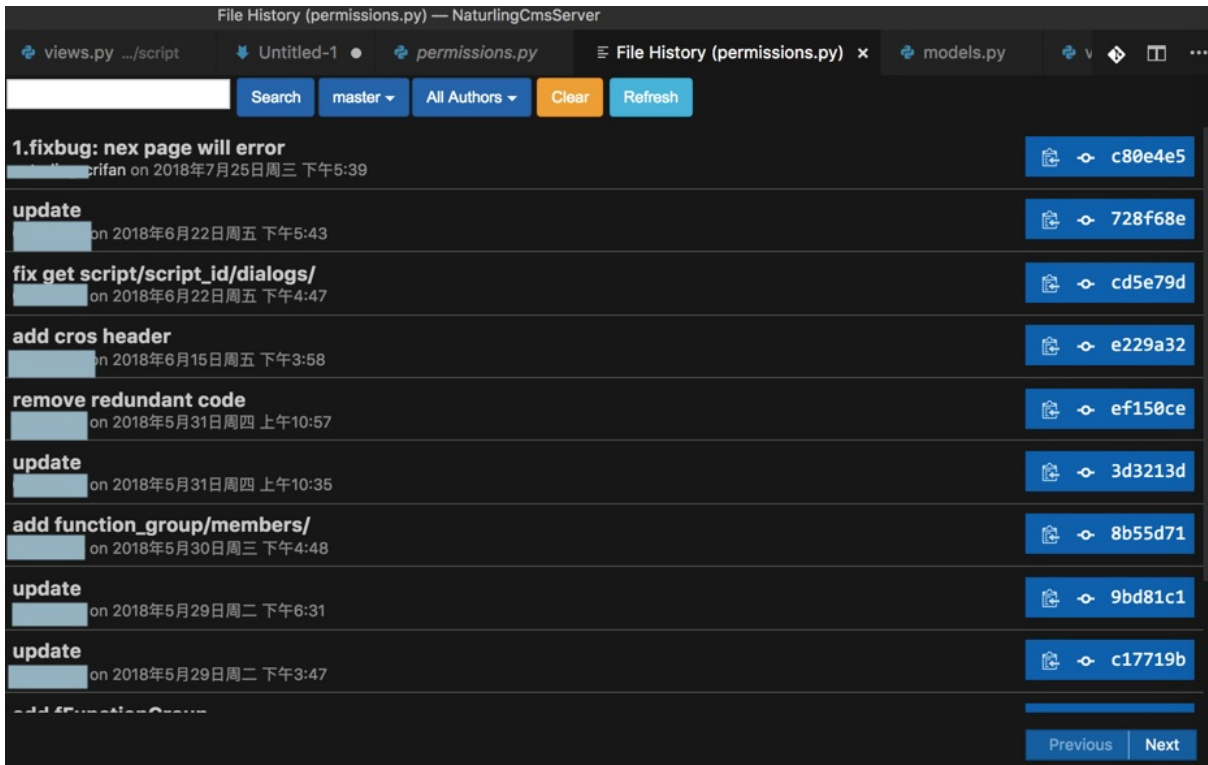
则可以：文件中右键选择 Git: View File History



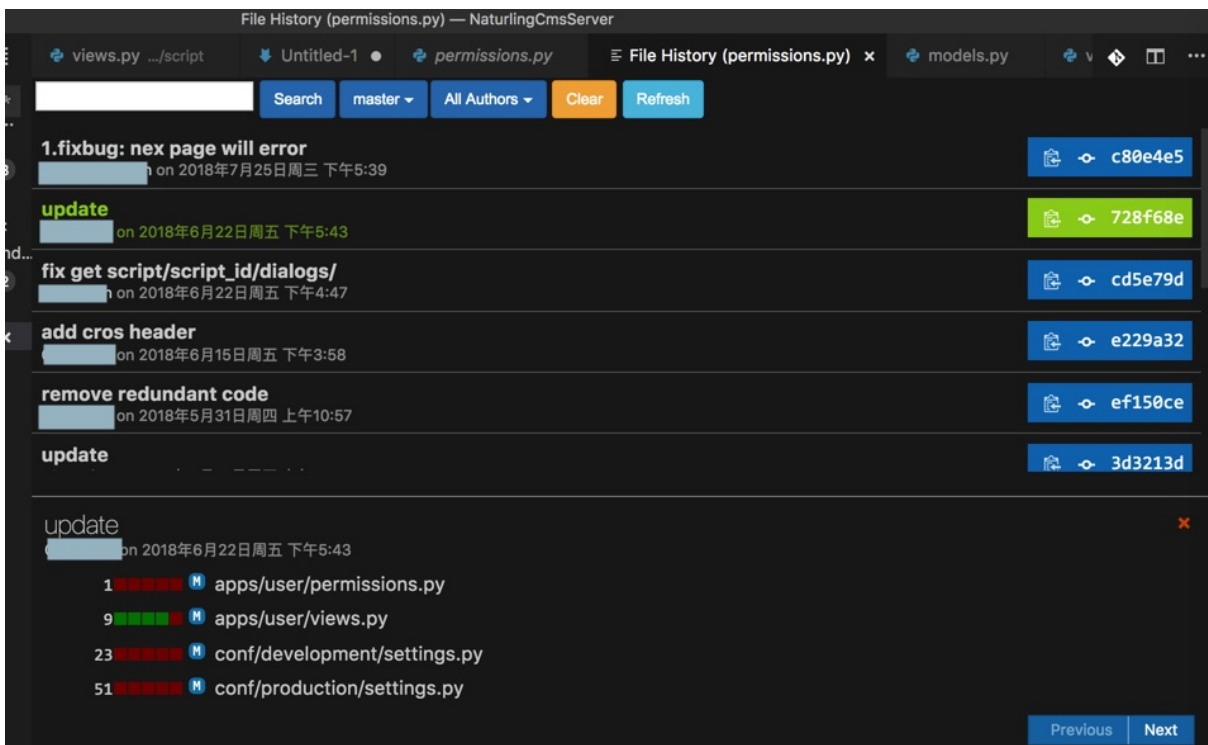
或Tab页中右键选择：



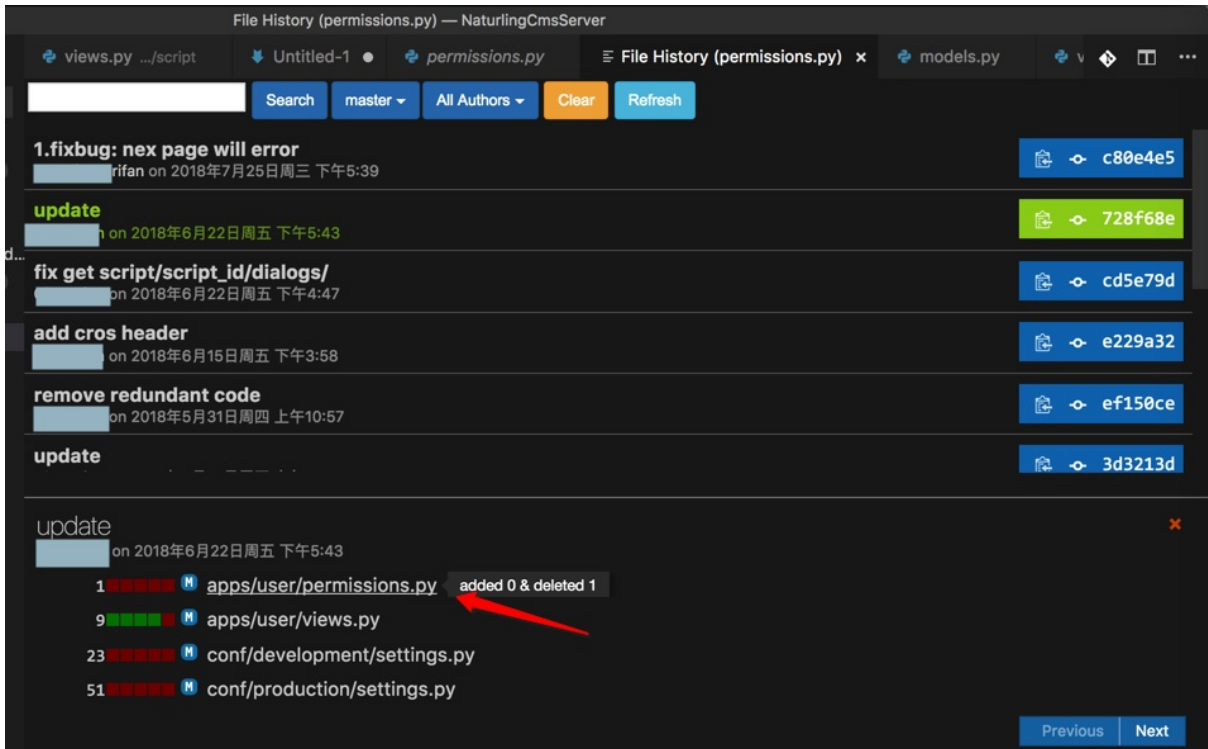
然后就可以看到各个历史版本了：



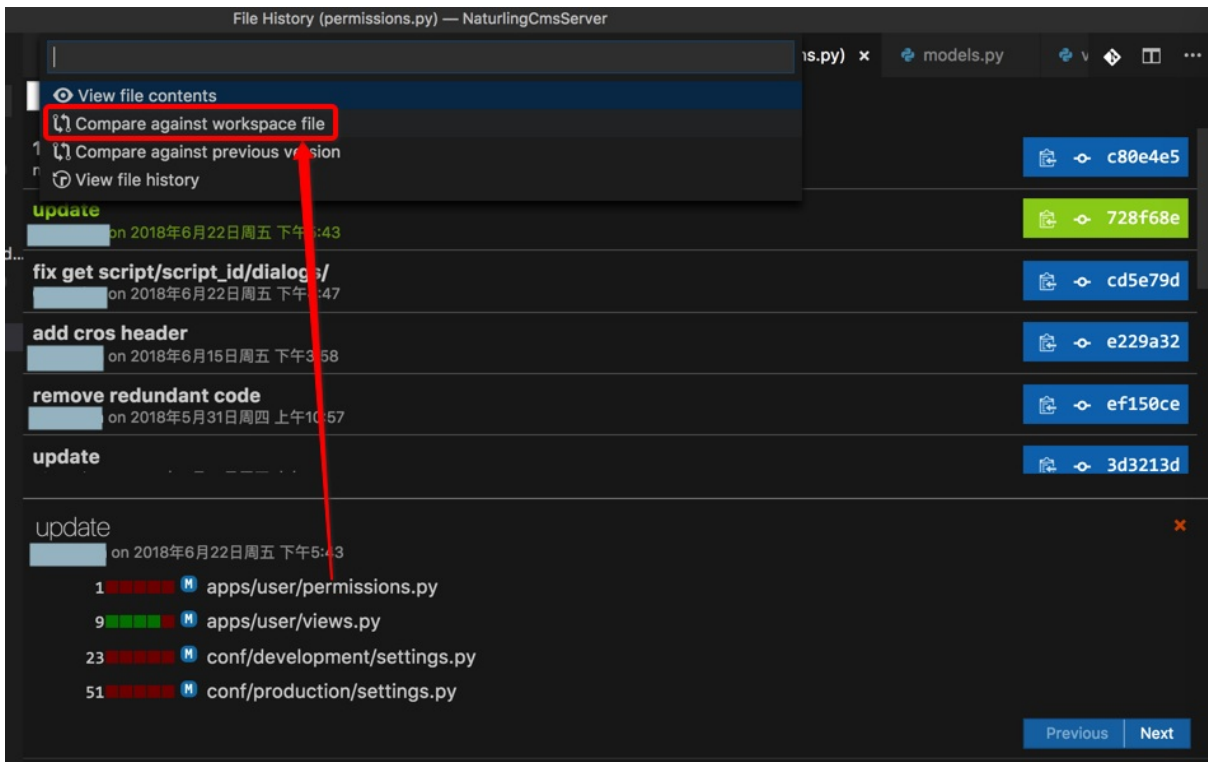
点击其中一个版本可以查看提交详情：



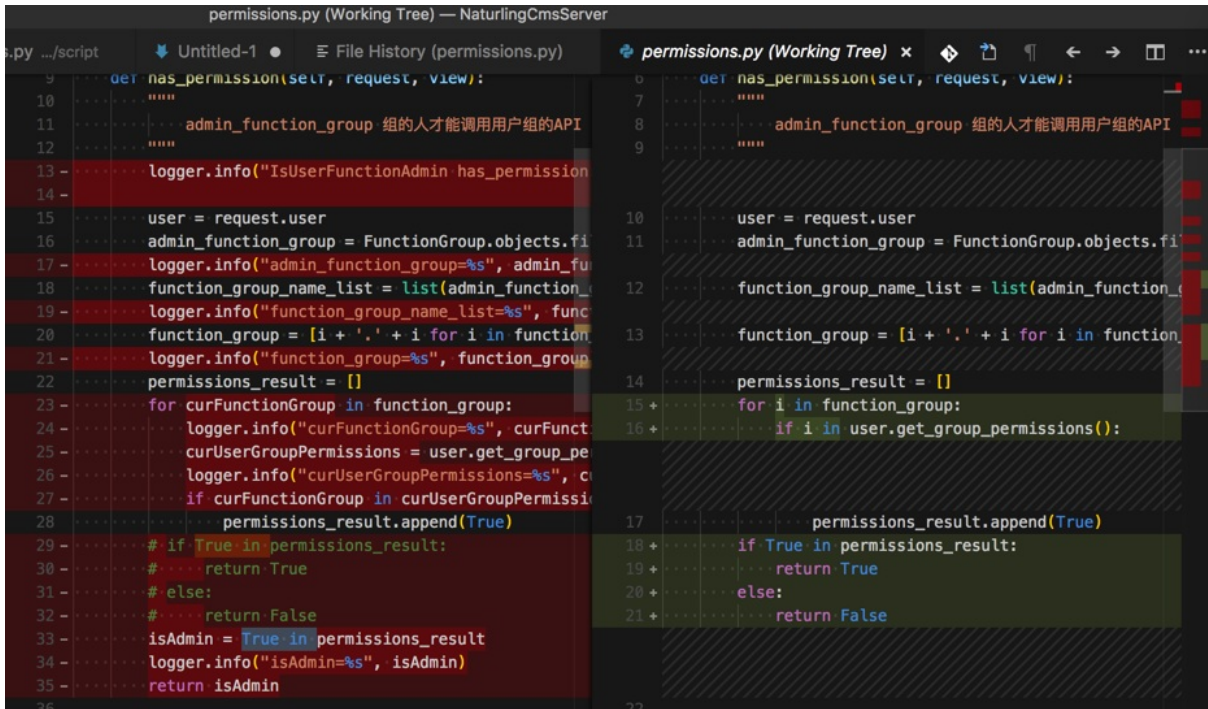
点击某次提交中的某个文件：



选择: Compare against workspace file

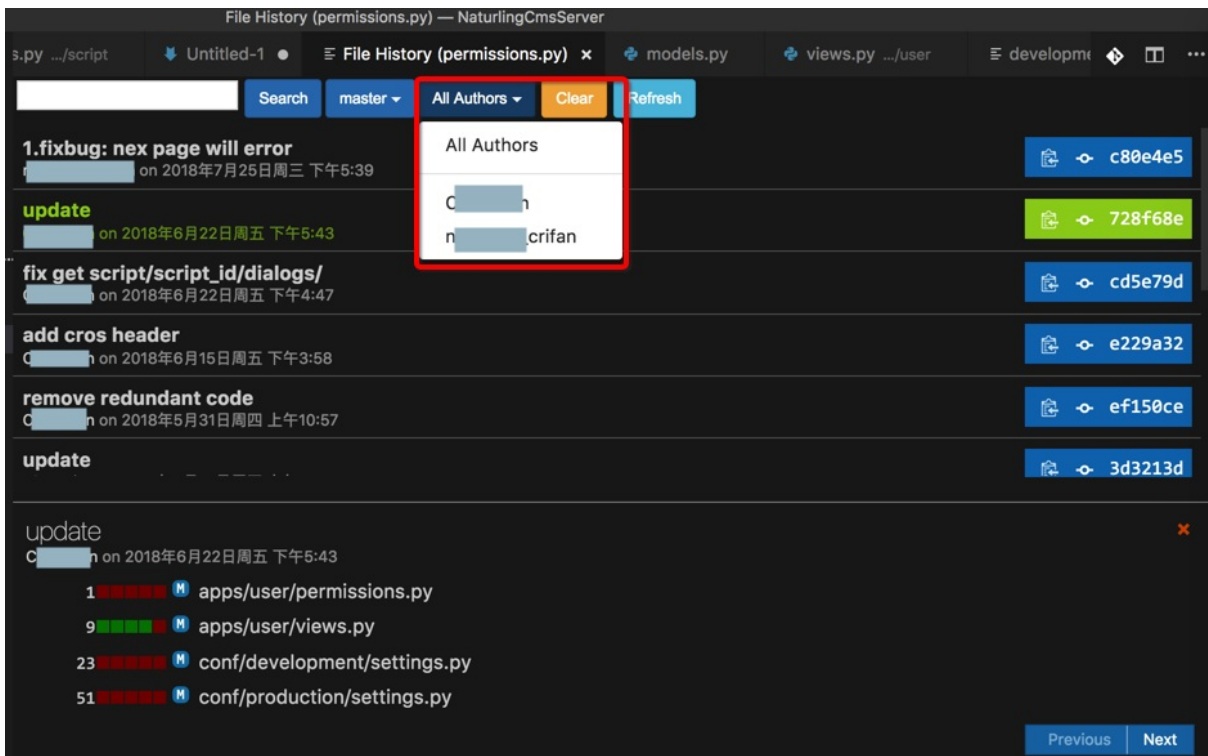


即可去该版本和当前最新文件, 去对比内容差异:

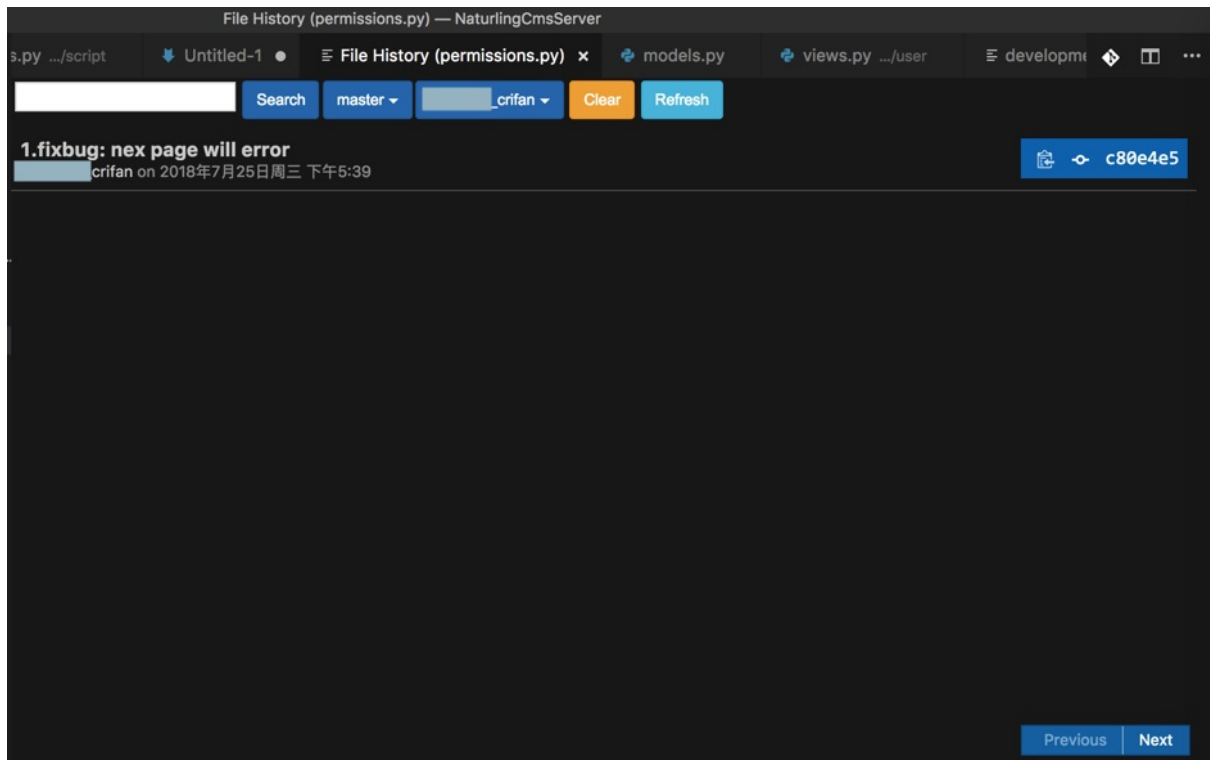


其中左边是最新内容，右边是之前该版本的内容。

另外，如果需要还可以在查看历史版本时，根据不同条件筛选，比如根据作者：



即可只查看某人的提交的代码：



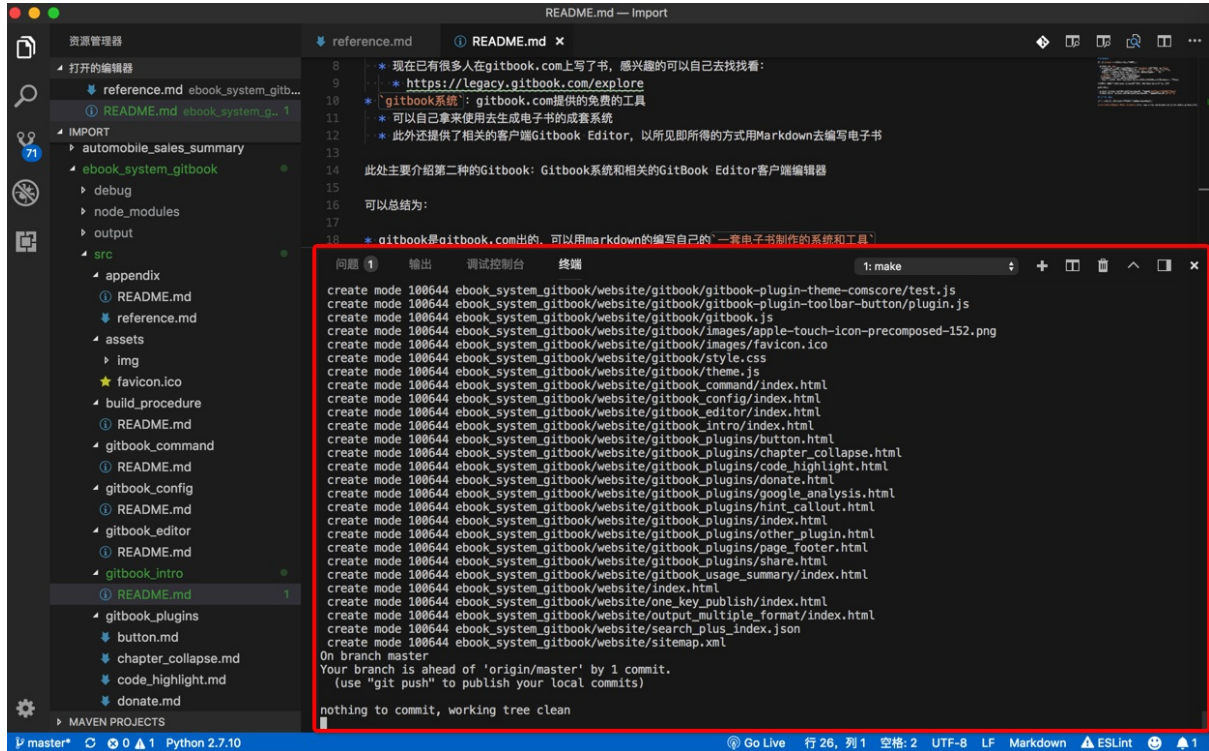
crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-16 21:09:14

集成终端

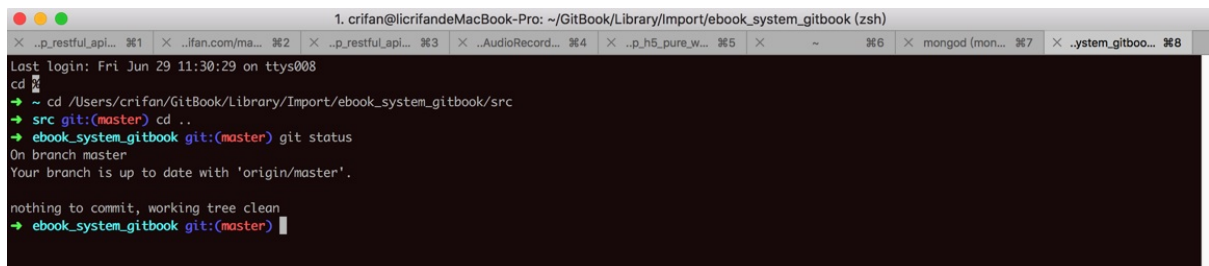
VSCode内置了终端：

查看 -> 终端

即可打开自带集成的终端：

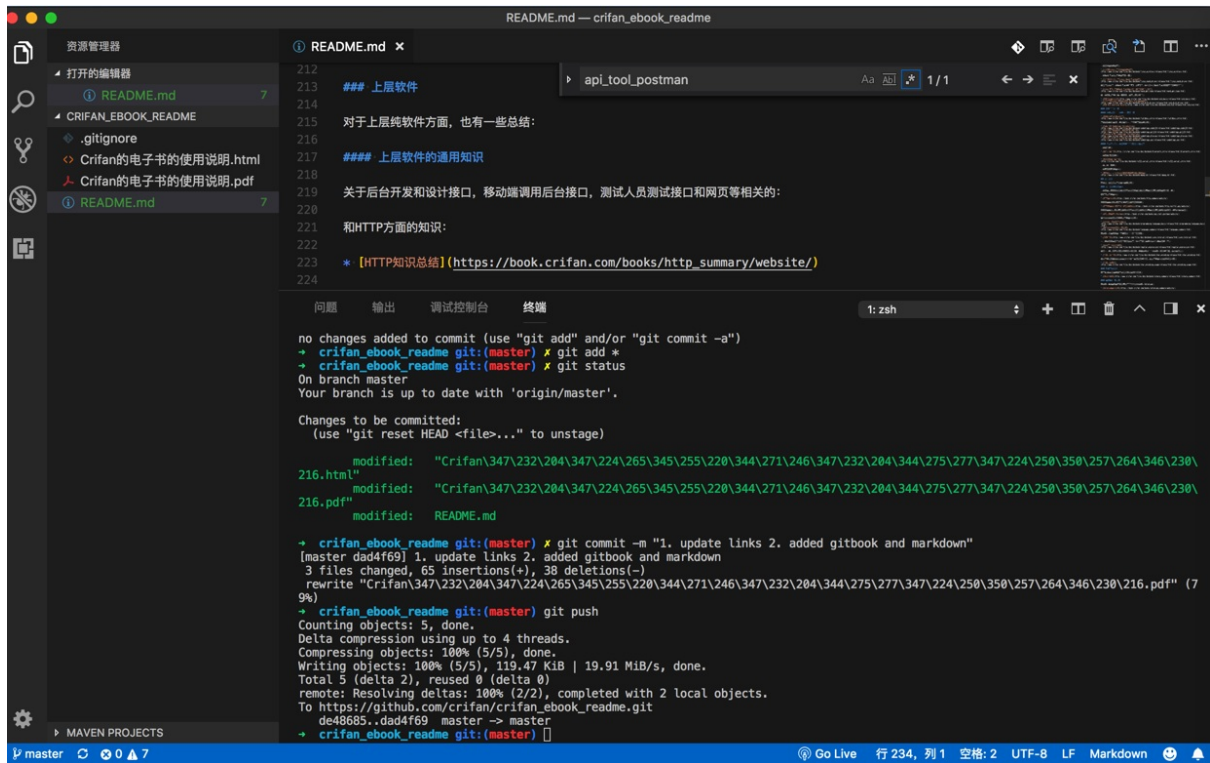


不用再去切换到外部的终端命令行：




直接可以方便的在一个工具界面下，同时处理不同的事情了。

另外一个例子：



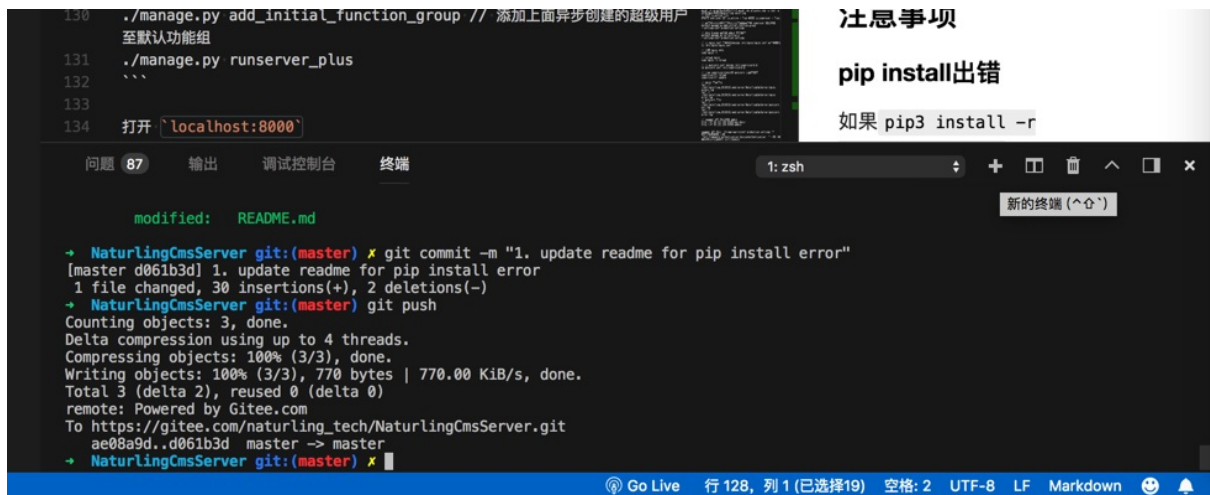
在更新了git中源码后，再去上传，直接在集成终端中进行，很方便。

且支持多个终端

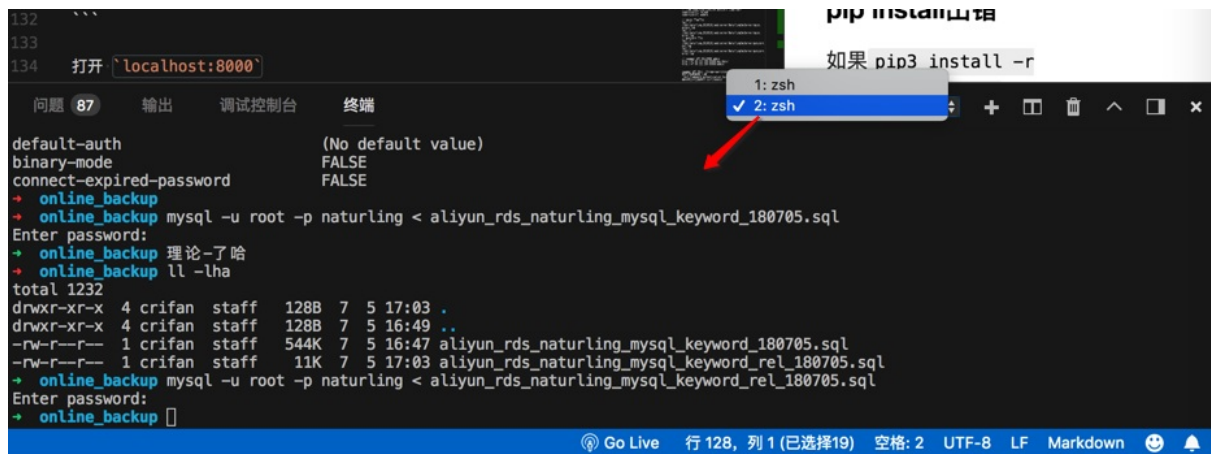
可以点击  加号去新增额外的终端，用于不同的用途。

实际使用场景举例：

比如此处在当前项目所在文件夹下，用git去管理代码：

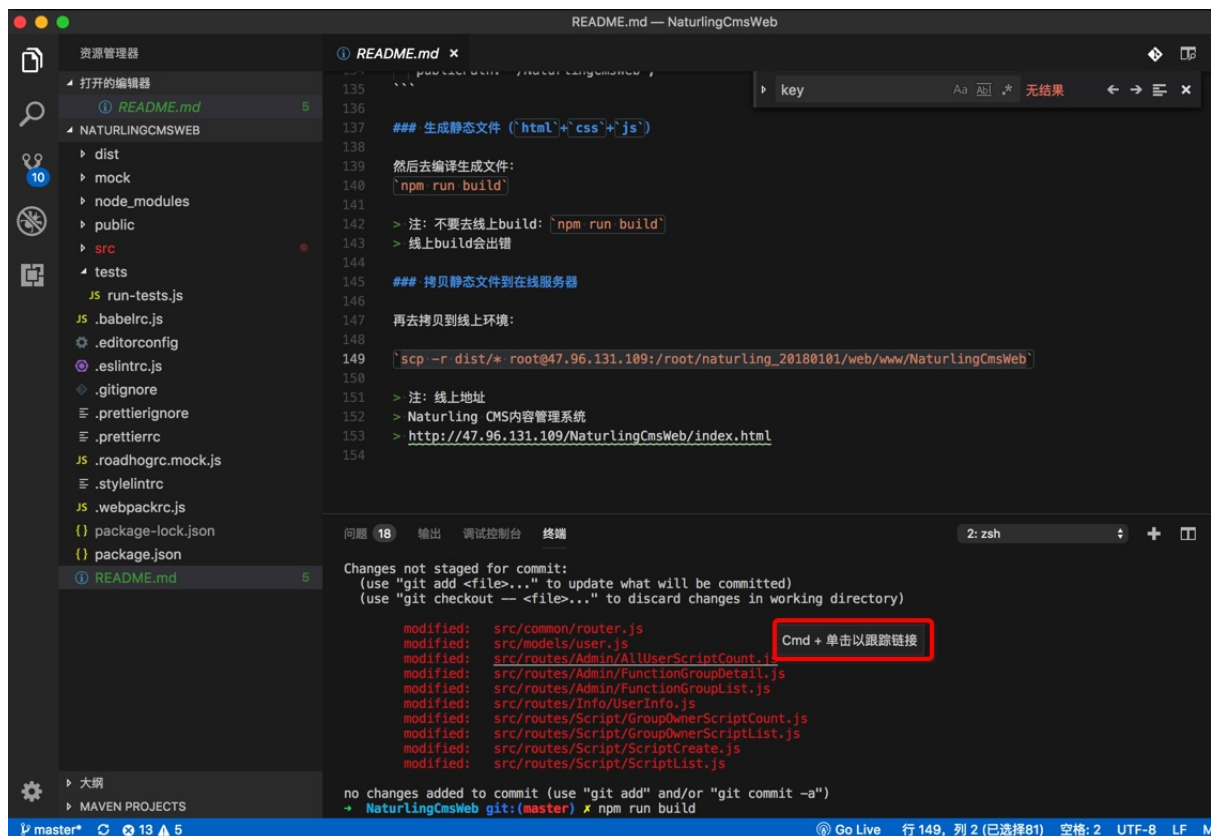


然后需要切换到另外一个目录，去导入mysql，所以点击+去新建终端，处理mysql：



集成的终端中对于git的支持很好

在VSCode的集成的终端中，除了git的颜色看起来很清楚明确，且git的commit后的log中，都可以识别commit哪些文件了：



且Command+鼠标点击可以直接跳转打开对文件：

The screenshot shows the VS Code editor interface. The main editor displays the code for `AllUserScriptCount.js`. The code includes imports for `React`, `dva`, `antd`, `routerRedux`, and `SimpleTable`. It uses `@connect` for state management and `@Form.create` for form handling. The component state is initialized with `formValues: {}`.

The terminal window at the bottom shows the output of `git status`, indicating that changes are not staged for commit. The list of modified files includes:

```

modified: src/common/router.js
modified: src/models/user.js
modified: src/routes/Admin/AllUserScriptCount.js
modified: src/routes/Admin/FunctionGroupDetail.js
modified: src/routes/Admin/FunctionGroupList.js
modified: src/routes/Info/UserInfo.js
modified: src/routes/Script/GroupOwnerScriptCount.js
modified: src/routes/Script/GroupOwnerScriptList.js
modified: src/routes/Script/ScriptCreate.js
modified: src/routes/Script/ScriptList.js

```

At the bottom of the terminal, it shows `no changes added to commit (use "git add" and/or "git commit -a")` and the current directory `NaturlingCmsWeb` on the `git:(master)` branch.

-> 正好省去了我本身要去：找找刚才已经git的commit并push了哪些文件，直接在VSCode的git的log中找到并打开相应文件了，很方便。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2019-03-16 21:09:21

插件

配置插件的参数

下面通过插件Paste Image为例，来解释，如何配置插件的参数。

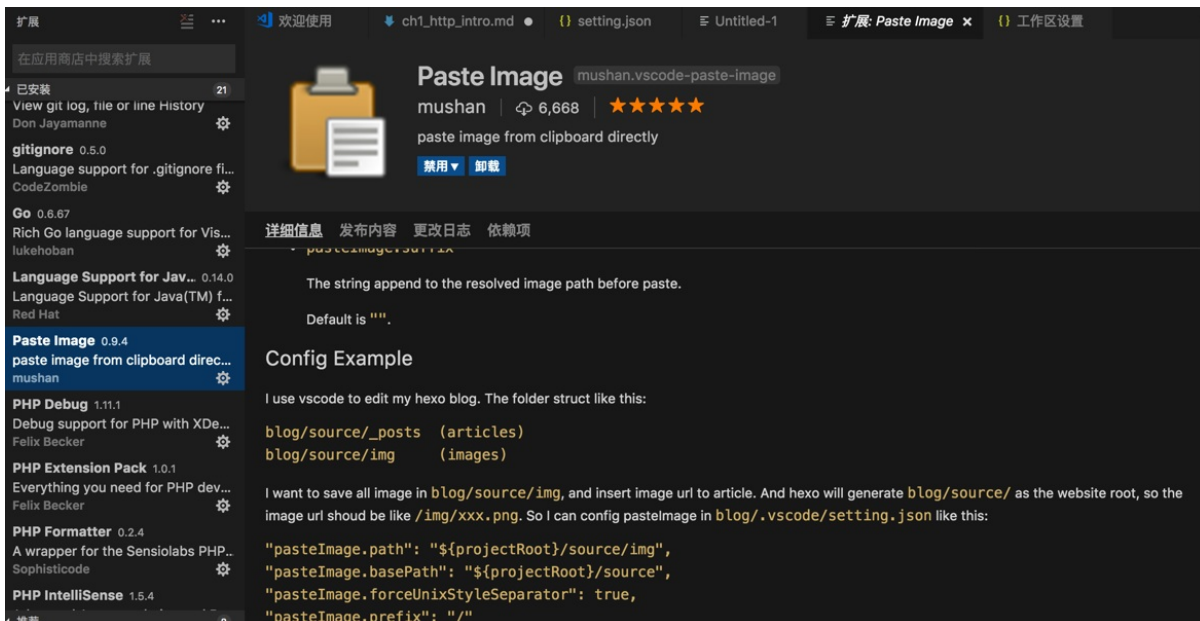
此处需求：希望能设置保存图片到当前项目下的assets/img中

首先要去搞清楚，插件本身支持哪些参数设置，可以从 发布内容 中看到：

名称	描述	默认
pasteImage.path	The destination to save image file.You can use variable <code>\${currentFileDir}</code> and <code>\${projectRoot}</code> . <code>\${currentFileDir}</code> will be replace by the path of directory that contain current editing file. <code>\${projectRoot}</code> will be replace by path of the project opened in vscode.	<code>\${currentFileDir}</code>
pasteImage.basePath	The base path of image url.You can use variable <code>\${currentFileDir}</code> and <code>\${projectRoot}</code> . <code>\${currentFileDir}</code> will be replace by the path of directory that contain current editing file. <code>\${projectRoot}</code> will be replace by path of the project opened in vscode. If you set basePath to empty String, it will insert absolute path to file.	<code>\${currentFileDir}</code>
pasteImage.forceUnixStyleSeparator	Force set the file separator styel to unix style. If set false, separator styel will follow the system style.	true
pasteImage.prefix	The string prepend to the resolved image path before paste.	
pasteImage.suffix	The string append to the resolved image path before paste.	

名称	描述	键盘快捷方式	菜单上下文
extension.pasteImage	Paste Image	⌘ ⇧ V	

以及有些插件的 详细信息 中也有关于参数设置的解释和举例：



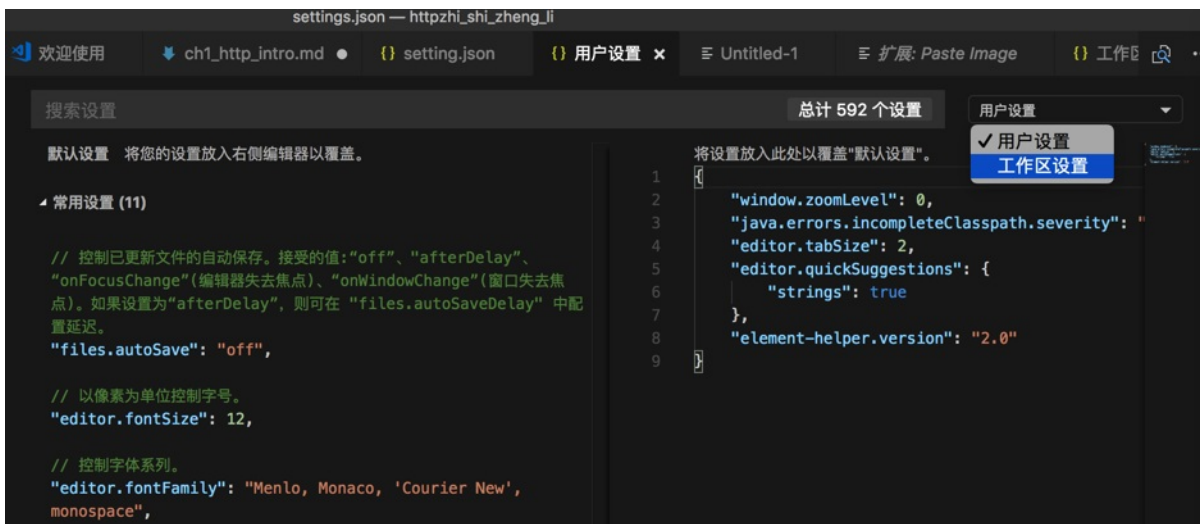
此处能看到与此处截图保存路径相关参数是：`pasteImage.path`，`pasteImage.forceUnixStyleSeparator`，`pasteImage.prefix`

然后去设置插件参数：

Code -> 首选项 -> 设置：



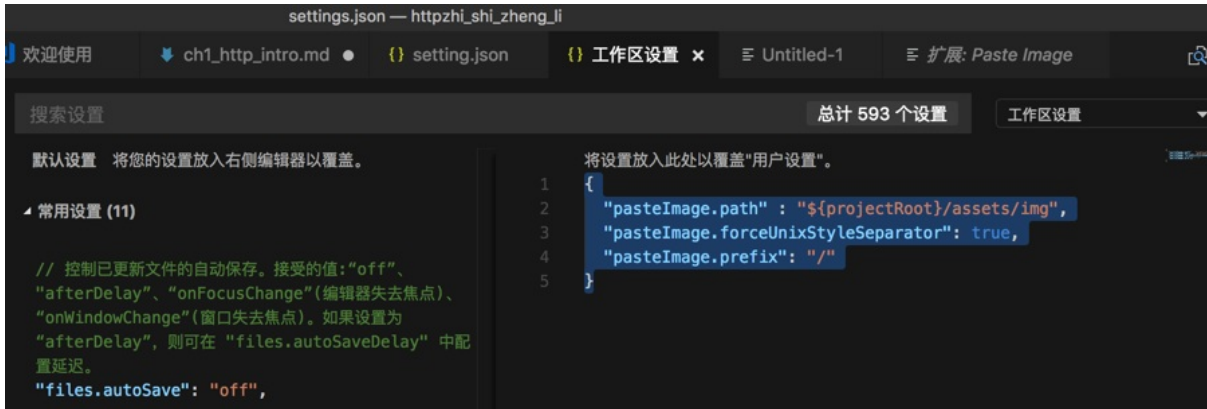
从 用户 切换到 工作区设置：



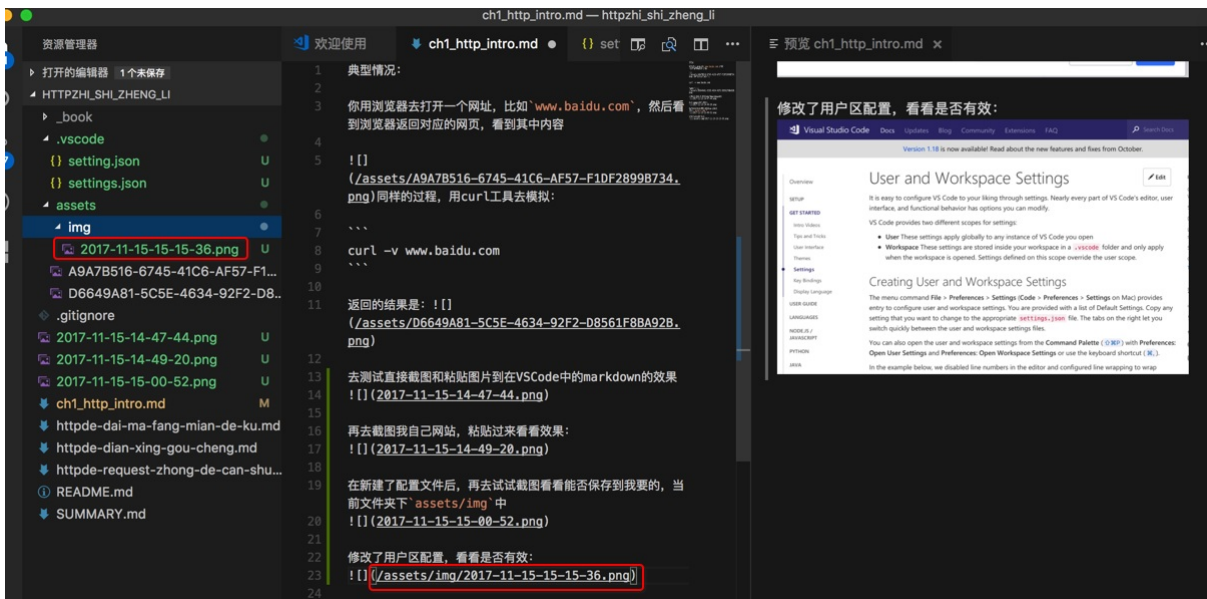
然后把此处相关配置：

```
{  
  "pasteImage.path": "${projectRoot}/assets/img",  
  "pasteImage.forceUnixStyleSeparator": true,  
  "pasteImage.prefix": "/"  
}
```

放进去并保存：



然后再去粘贴图片，然后就生效了，新粘贴的图片就自动保存到希望的目录 assets/img 下了：



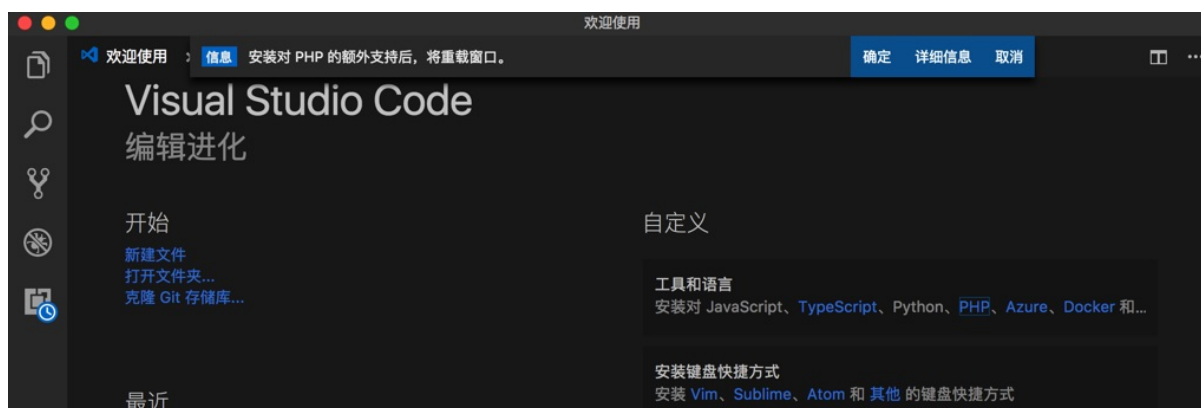
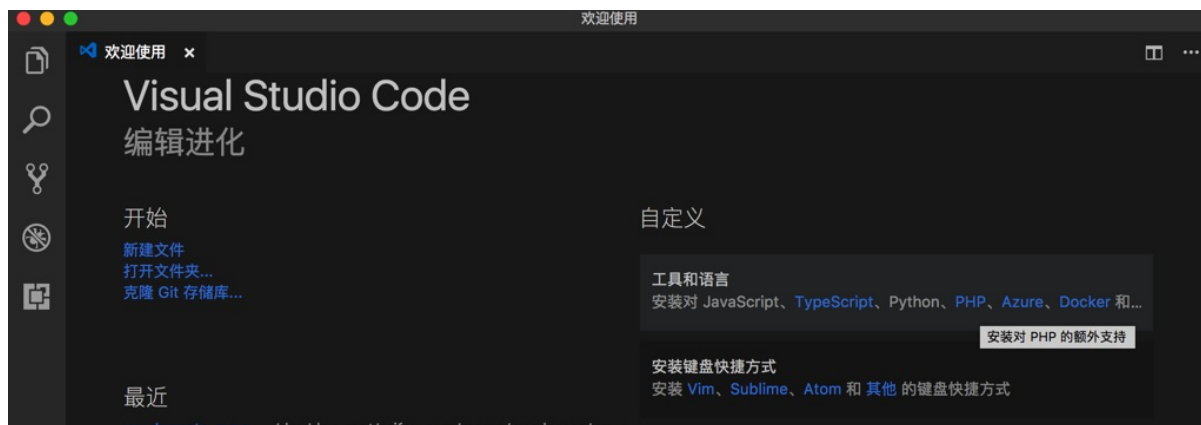
详见：[【已解决】VSCode中如何给插件进行参数配置](#)

插件市场

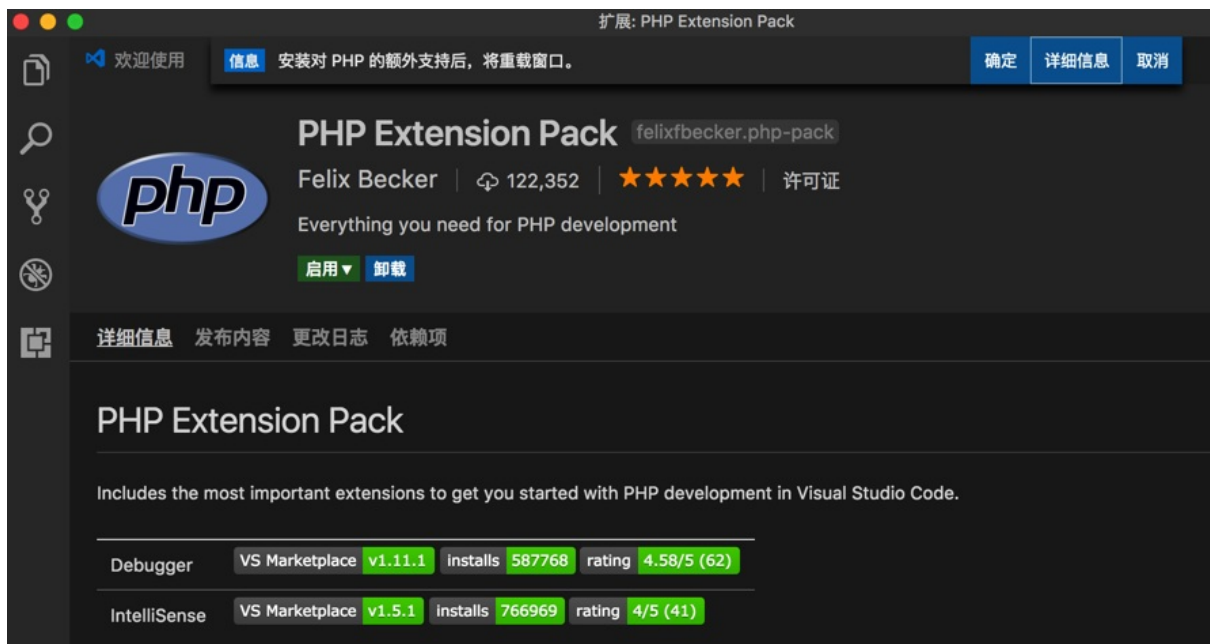
为了支持更多功能和语言，可以通过插件实现。

安装插件很方便

比如此处想要添加PHP语言的支持，可以在这里直接点击PHP：

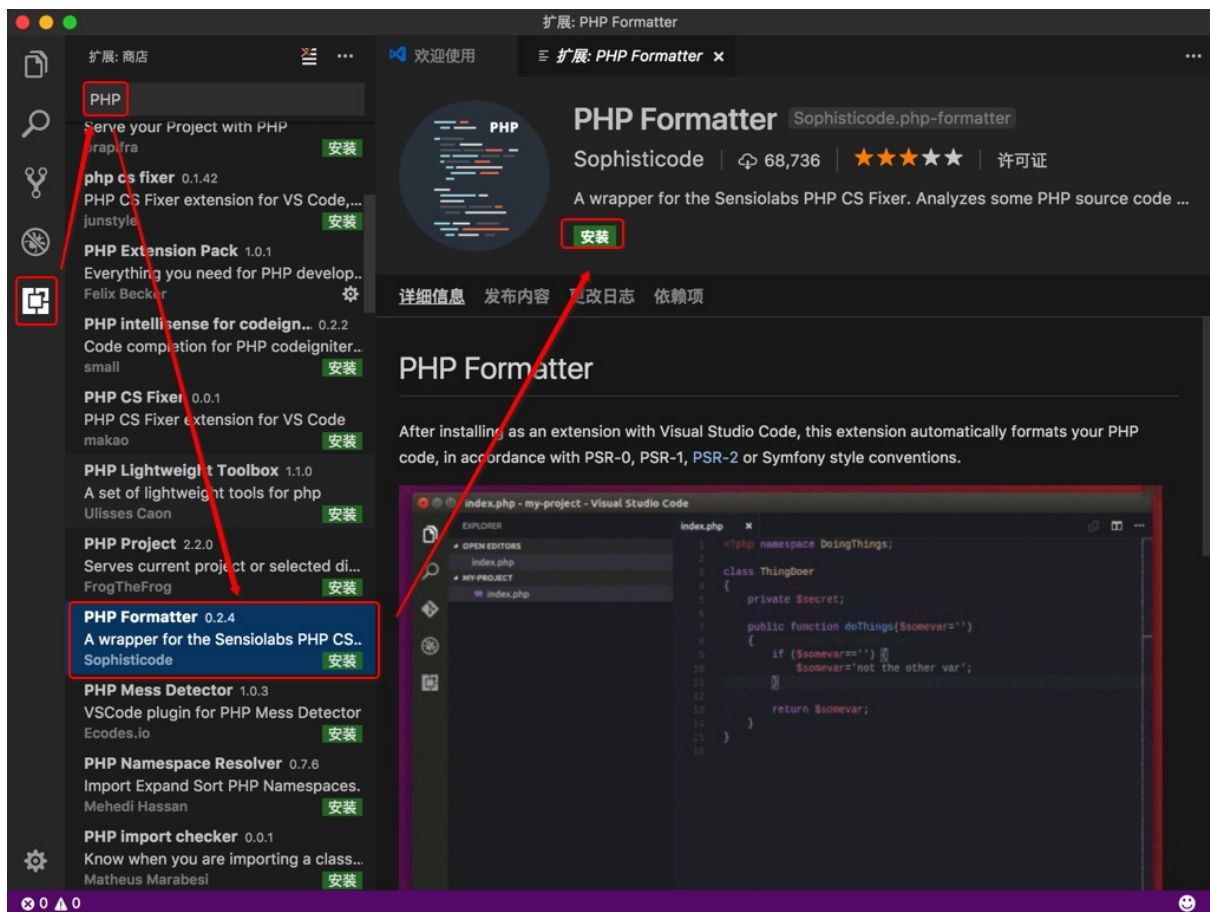


点击详细信息：

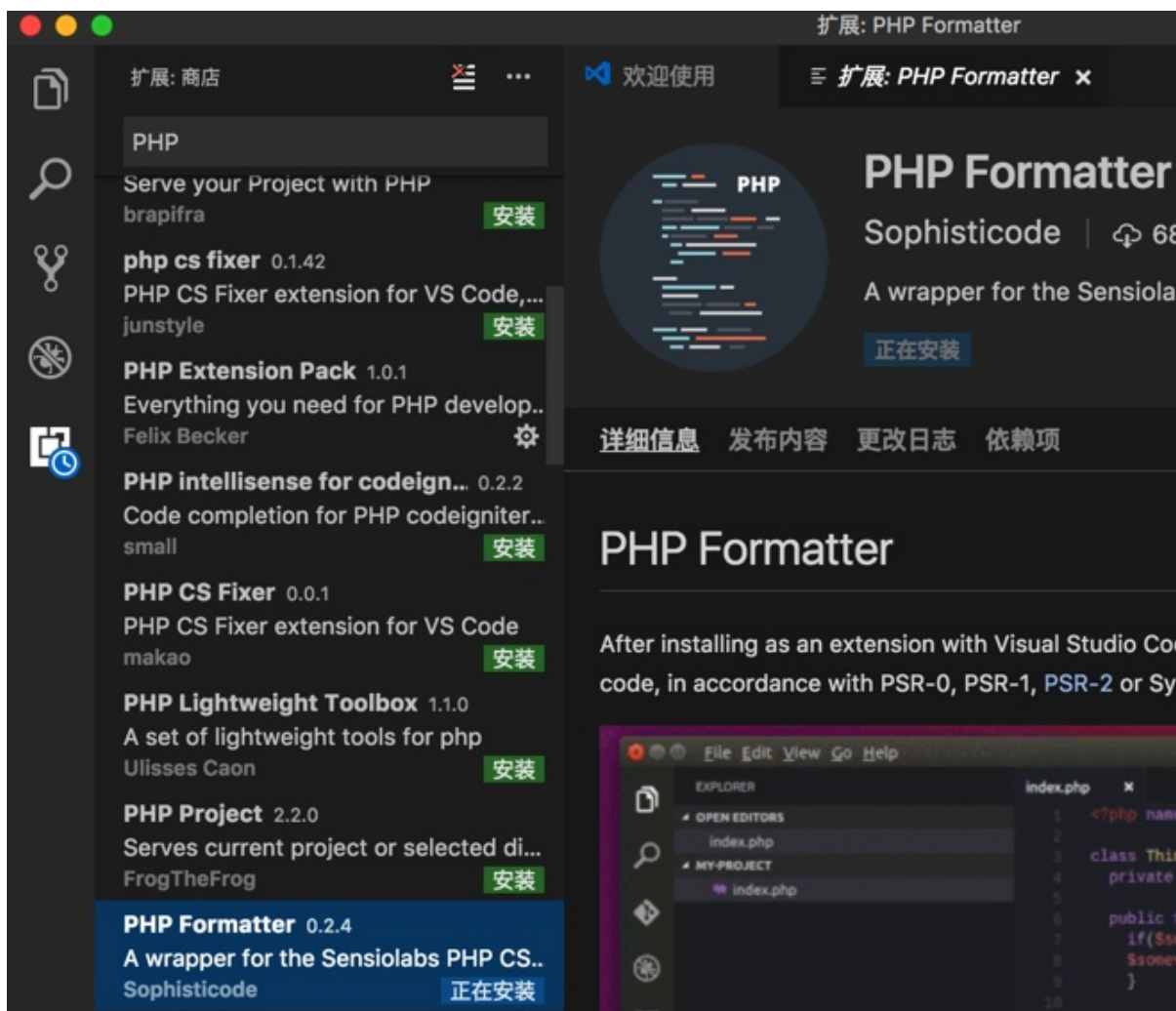


其实更常见的做法是:

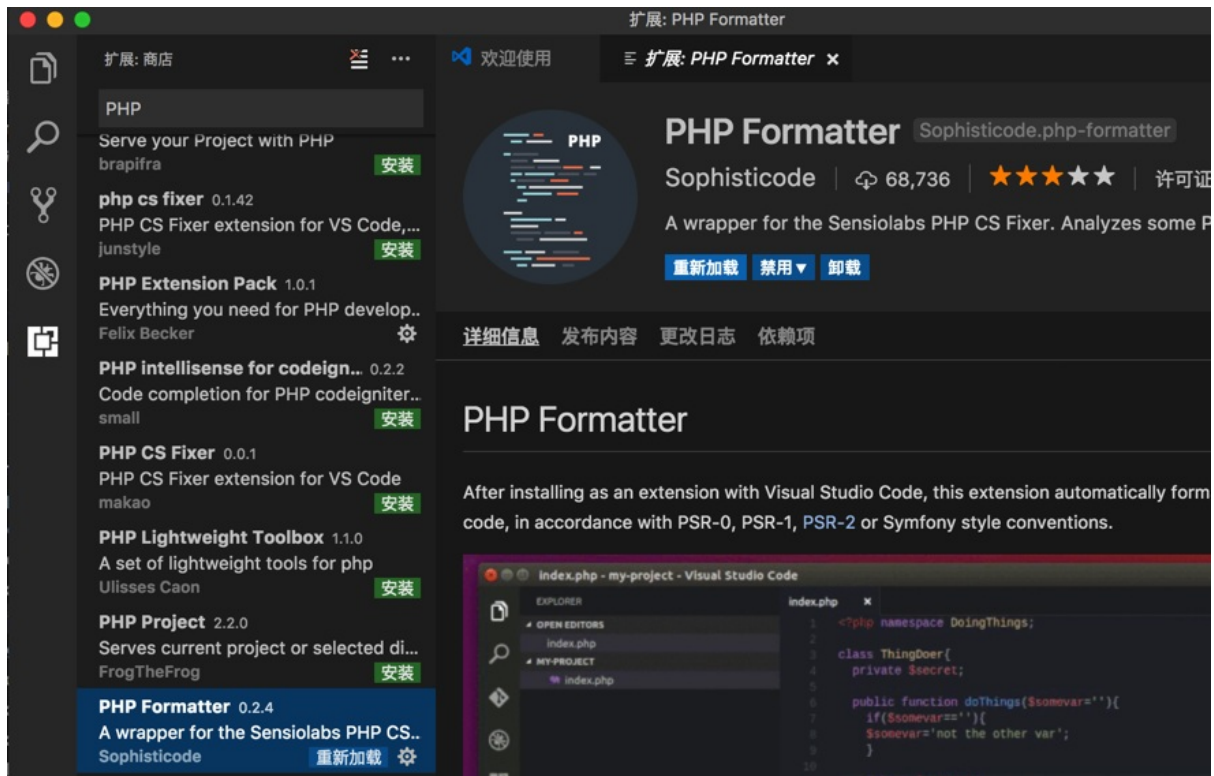
扩展 -> 输入框中输入: PHP -> 点击某个插件的: 安装



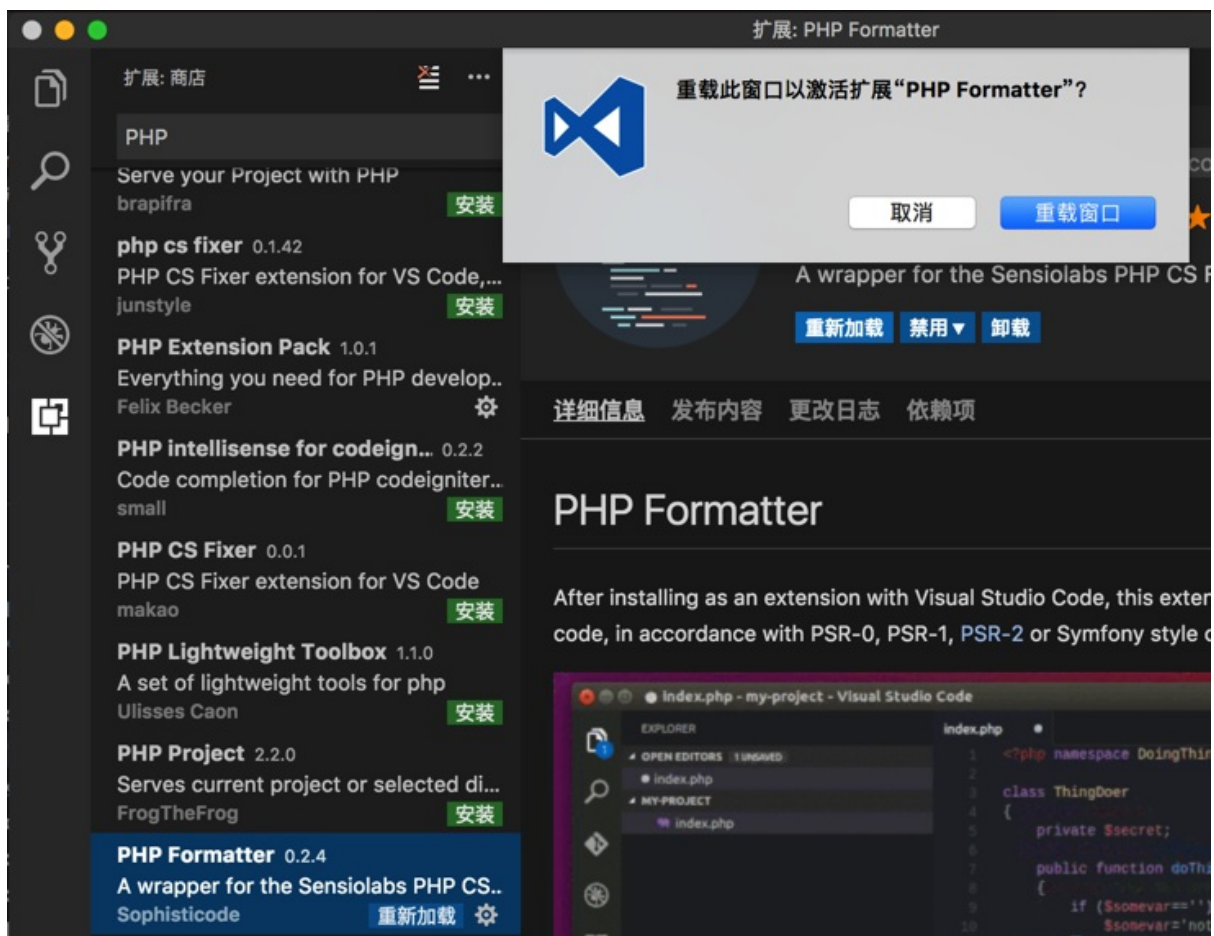
即可安装:



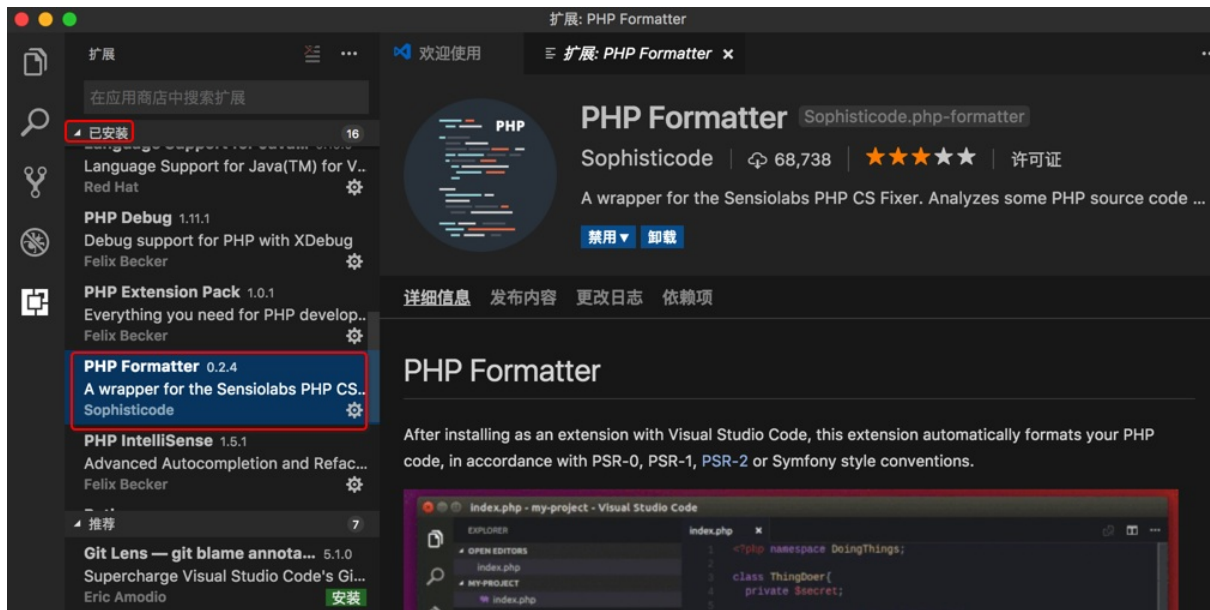
安装后，提示重启：



点击重新加载：



重启VSCode后，即可看到已安装该插件：



注：后来新版VSCode，对于某些插件，安装后，无需重启即可使用。更方便。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-17 23:05:17

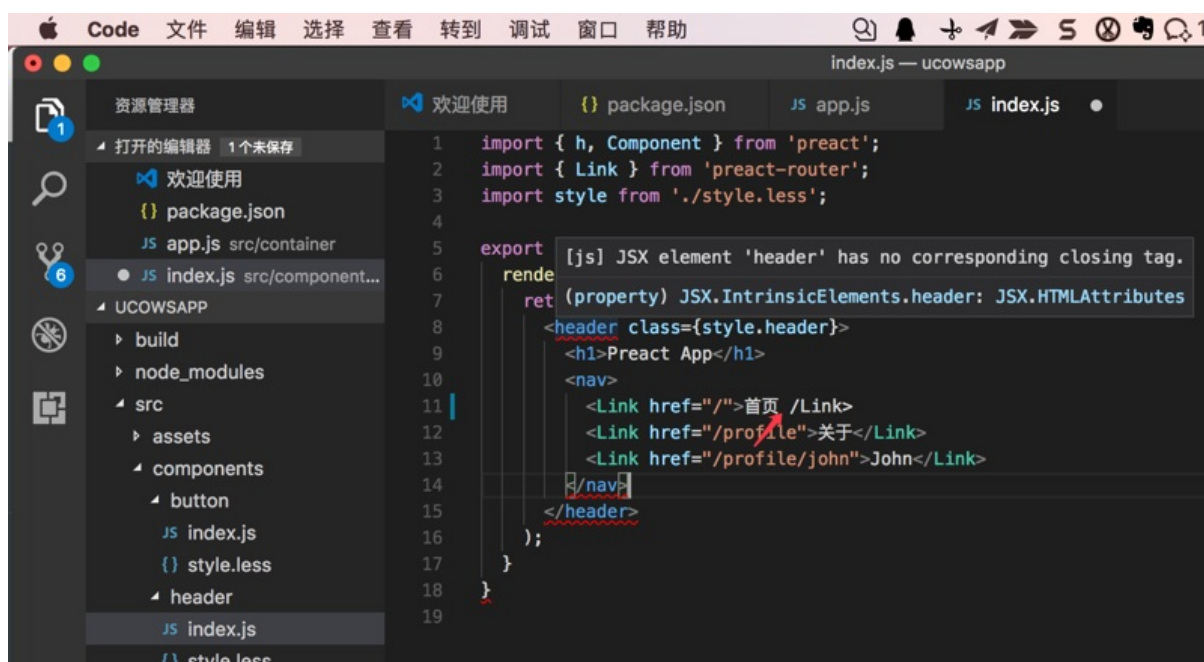
常用插件

其他一些插件

代码语法检测

插件: [Microsoft/vscode-eslint: VSCode extension to integrate eslint into VSCode](#)

作用: 代码语法出错时可以检测出并提示你:



详见: [【记录】给VSCode装插件: eslint](#)

editorconfig

插件地址: [editorconfig/editorconfig-vscode: EditorConfig extension for Visual Studio Code](#)

作用: 通过 `.editorconfig` 文件覆盖重写VSCode相关配置, 而无法独立的 `.vscode` 文件夹 (及其中的一堆配置文件)

详见: [【记录】给VSCode装插件: editorconfig](#)

安装插件心得

Could not install packages due to an EnvironmentError Erno 13 Permission denied

原因: 安装Python相关库时没权限

办法: `python -m pip install -U pylint --user`

详见: [【已解决】VSCode提示安装lint出错: Could not install packages due to an EnvironmentError Erno 13 Permission denied](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-17

12:21:28

Markdown Preview Enhanced

TODO:

- 预览的效果好
- 快速生成文件
 - html
 - pdf
 - 图片：png/jpg等

插件地址：[Markdown Preview Enhanced - Visual Studio Marketplace](#)

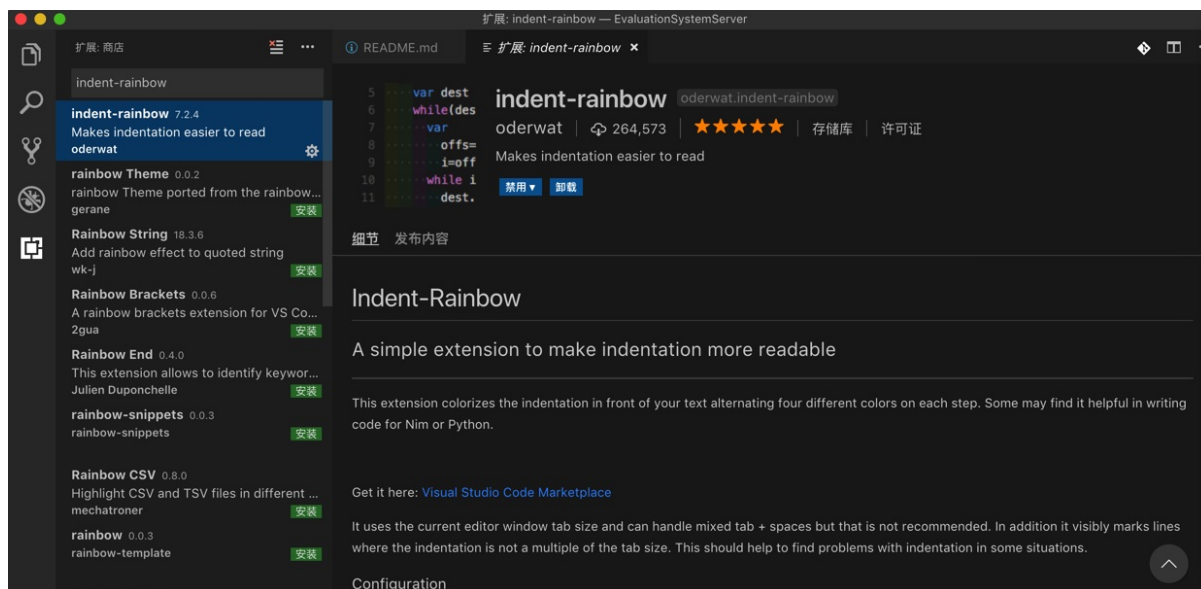
crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook 最后更新: 2019-03-17 23:15:19

indent-rainbow

插件地址: [indent-rainbow - Visual Studio Marketplace](#)

作用: 高亮 (每行代码之前的) 缩进

安装 indent-rainbow :



效果:

- 对于 Python 代码效果: 更方便清晰的看到代码的缩进

o

- 对于 Html 效果: 方便看清是哪个元素

-
- 对于缩进层次较多的：效果尤其明显

◦

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-16 10:31:02

Bracket Pair Colorizer

TODO:

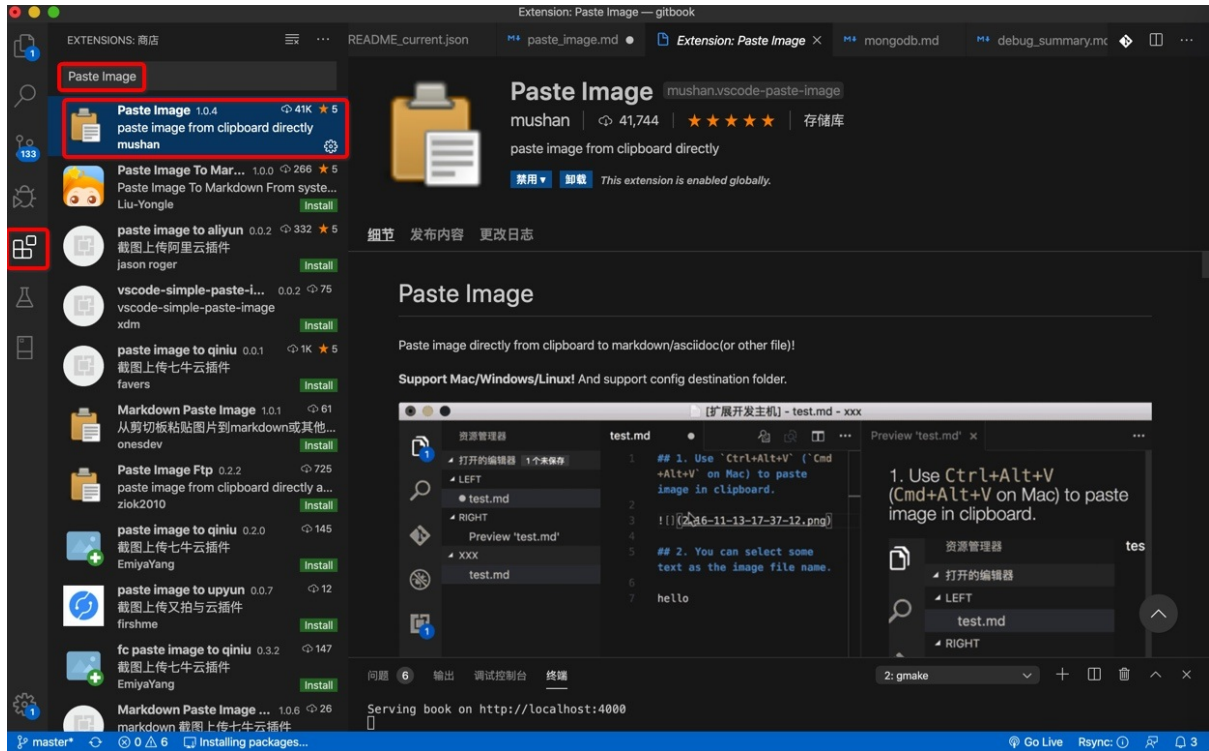
- 列出来括号高亮的效果，尤其是多组嵌套括号

插件地址: [Bracket Pair Colorizer - Visual Studio Marketplace](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook 最后更新: 2019-03-17 23:15:03

Paste Image

先去 插件 中安装插件： Paste Image



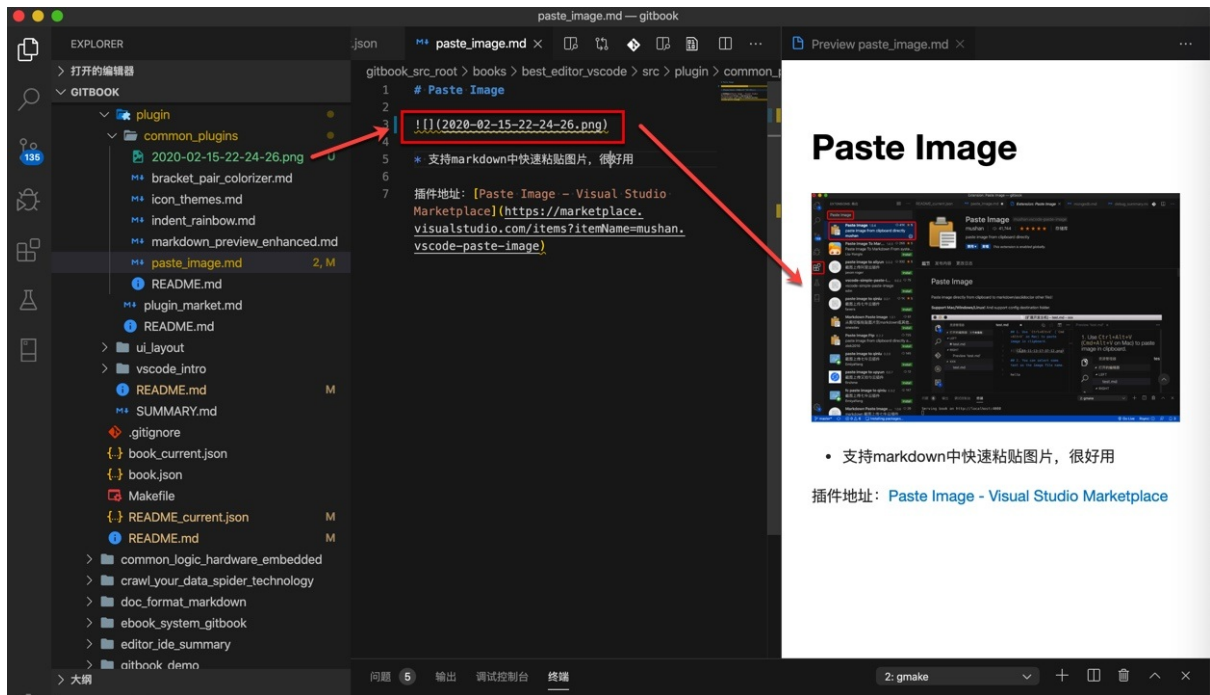
然后确保当前系统剪贴板中有图片，比如复制了图片或者截了图

然后 Mac 中用快捷键 `Command+Alt+V`，稍等一下，即可粘贴出图片到当前目录，且在当前 Markdown 中插入图片的内容了：

```

```

效果如图：



如此，方便的在当前Markdown同目录下插入图片，效果很不错。

附上：

插件地址：[Paste Image - Visual Studio Marketplace](https://marketplace.visualstudio.com/items?itemName=mushan.vscode-paste-image)

另外，后来遇到过快捷键冲突的，可以参考这个去解决：

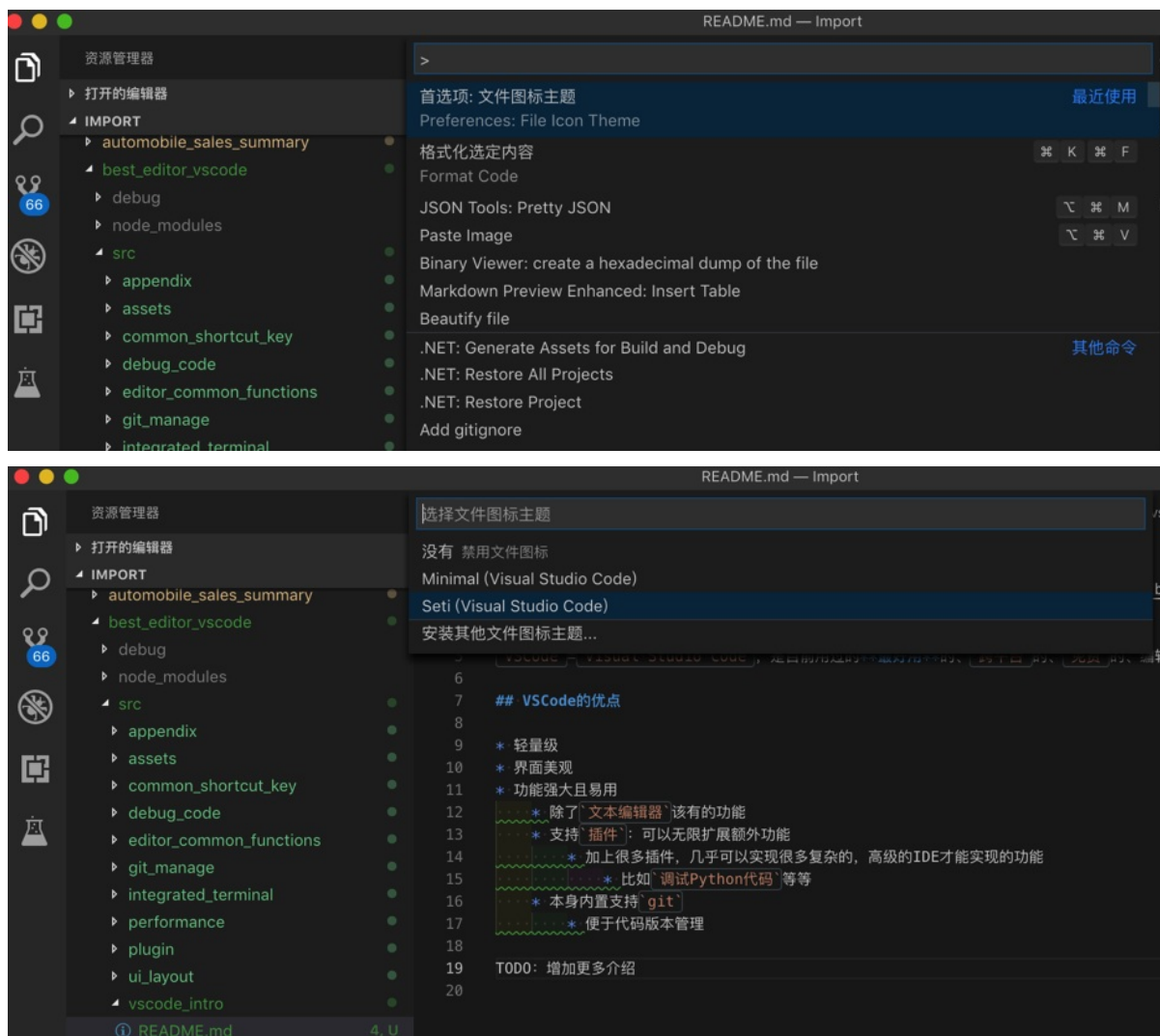
[【已解决】VSCode中插件Paste Image的粘贴图片快捷键Command+Alt+V失效 – 在路上](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新： 2020-02-15 22:46:11

文件图标主题

VSCode 支持给文件的图标设置不同的主题。

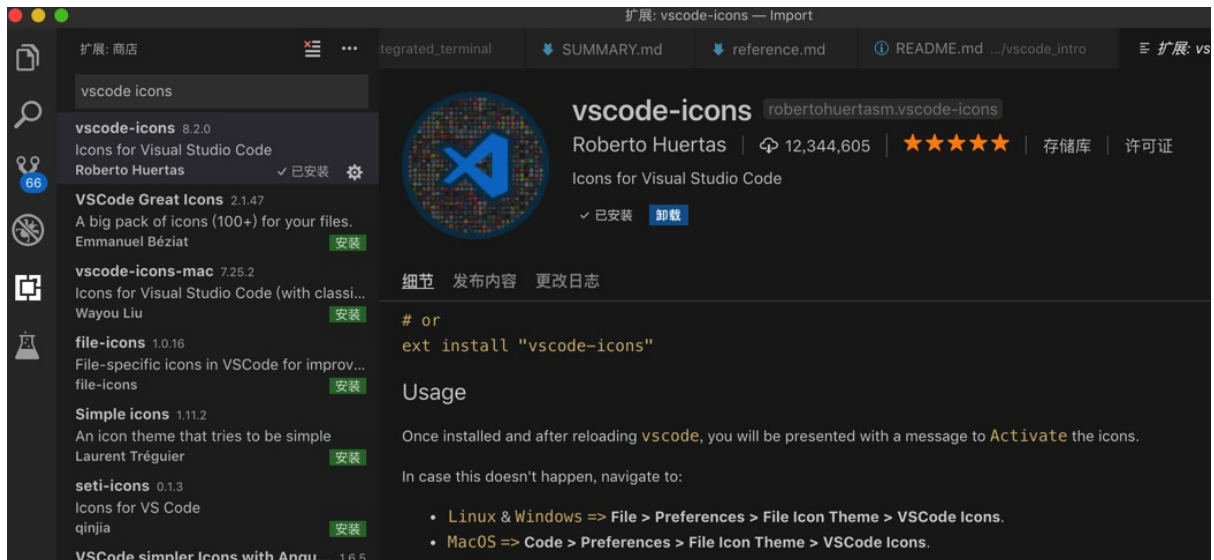
默认是 Seti 主题，效果是：



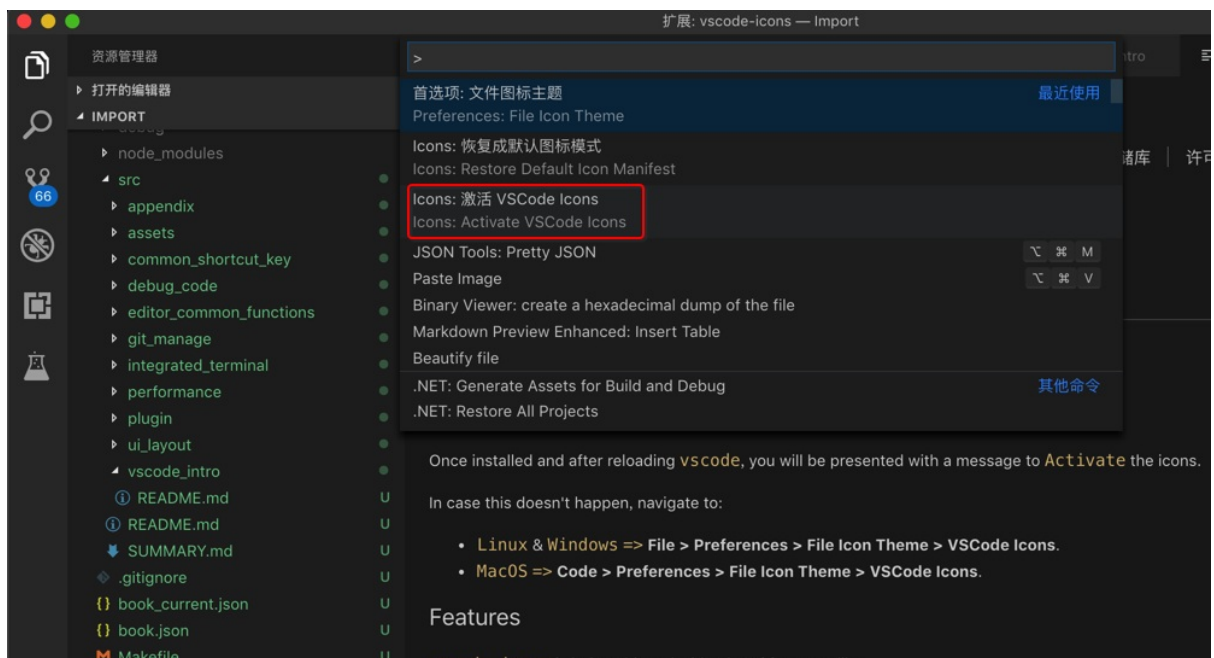
有其他新的图标主题，比如：

[vscode-icons](#)

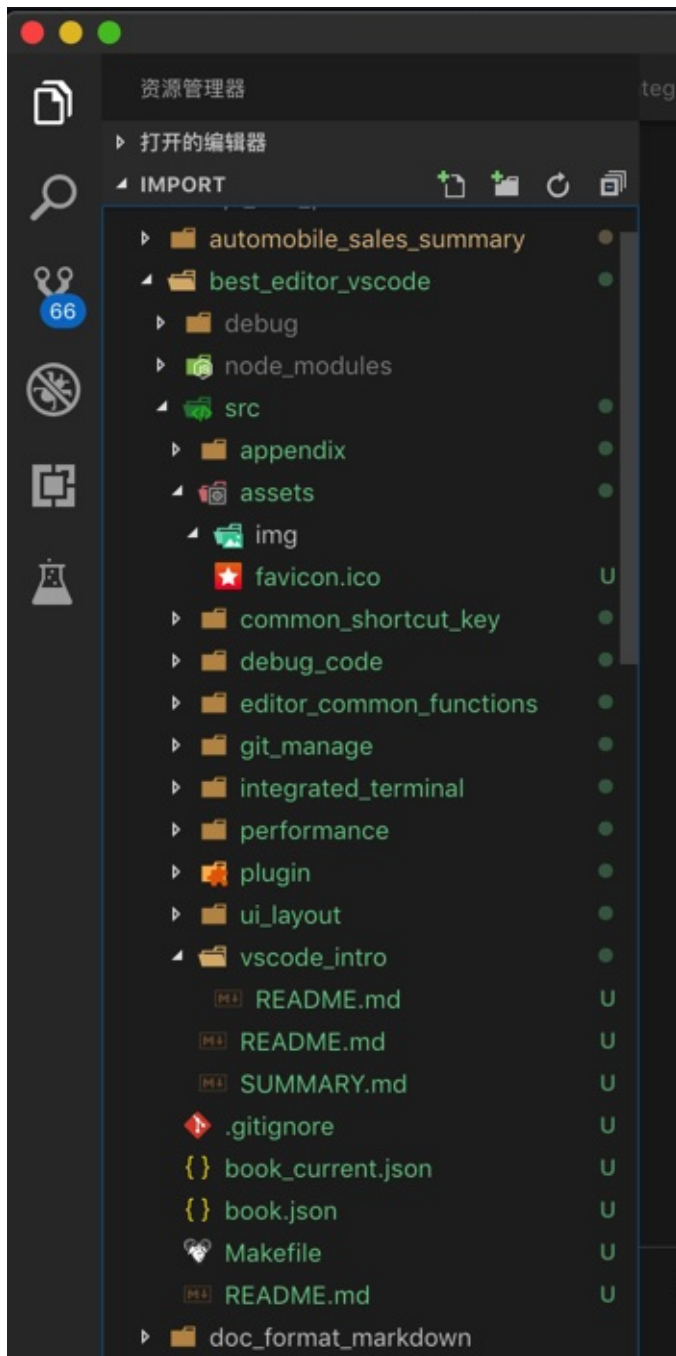
去安装：



之后再激活:

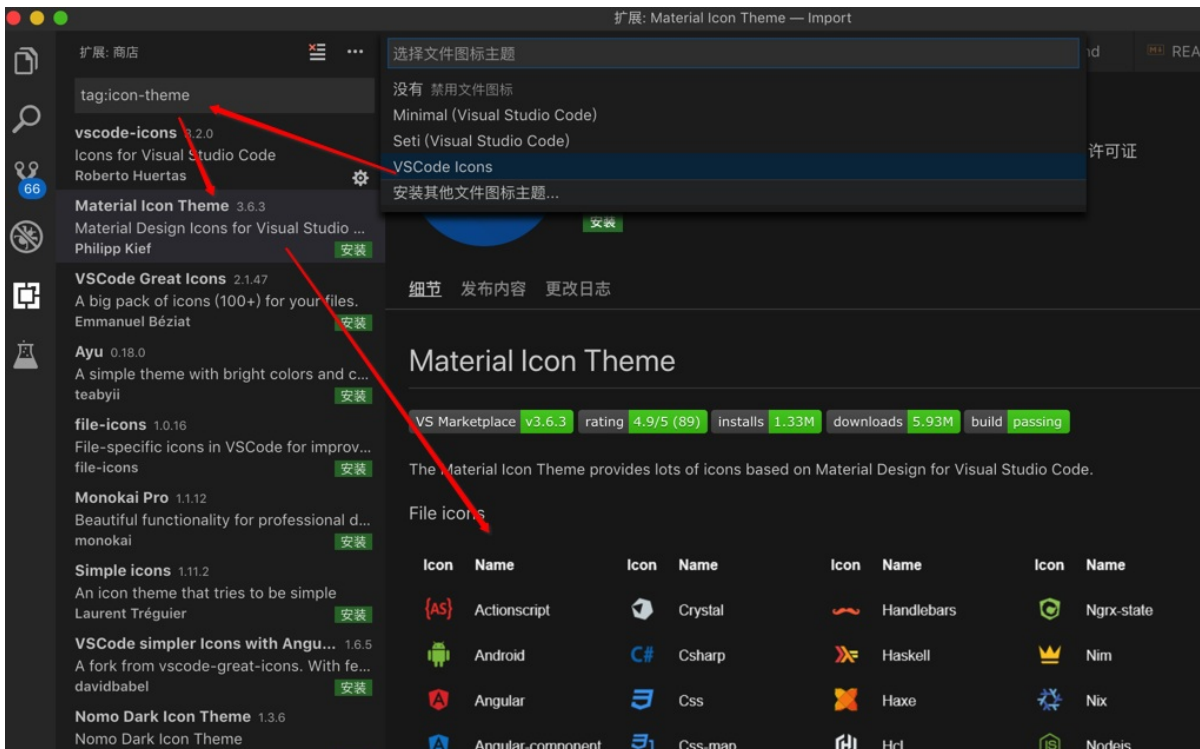


后的效果:



很不错，图标很丰富，识别度比较高。

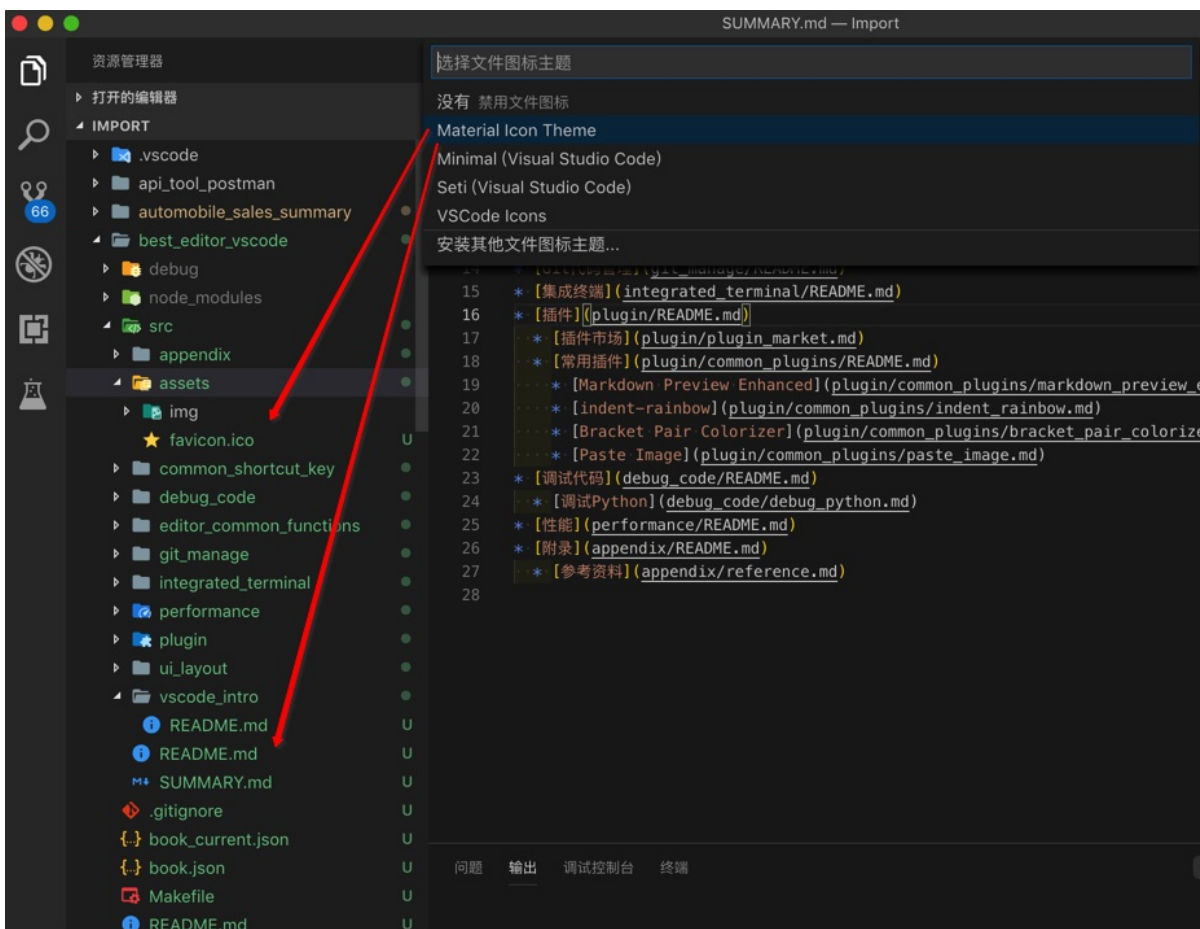
另外，可以再去根据提示去安装其他版本图标的主题：



然后去安装另外一个：

Material Icon Theme

后，去换成该主题：



感觉比前面的 `vscode-icons` 更好看，更现代化些。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook 最后更新: 2019-03-16 09:35:48

调试代码

VSCode可以通过内置功能和安装插件，实现支持调试多种代码。

比如下面要介绍的Python。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2020-02-16 15:01:18

调试Python

VSCode支持调试Python代码。

核心逻辑：

- 首次初始化要做的事
 - 确保已安装 Python 插件
 - 选择Python版本
 - （一次性）设置好调试配置
- 之后每次可以直接
 - 启动调试

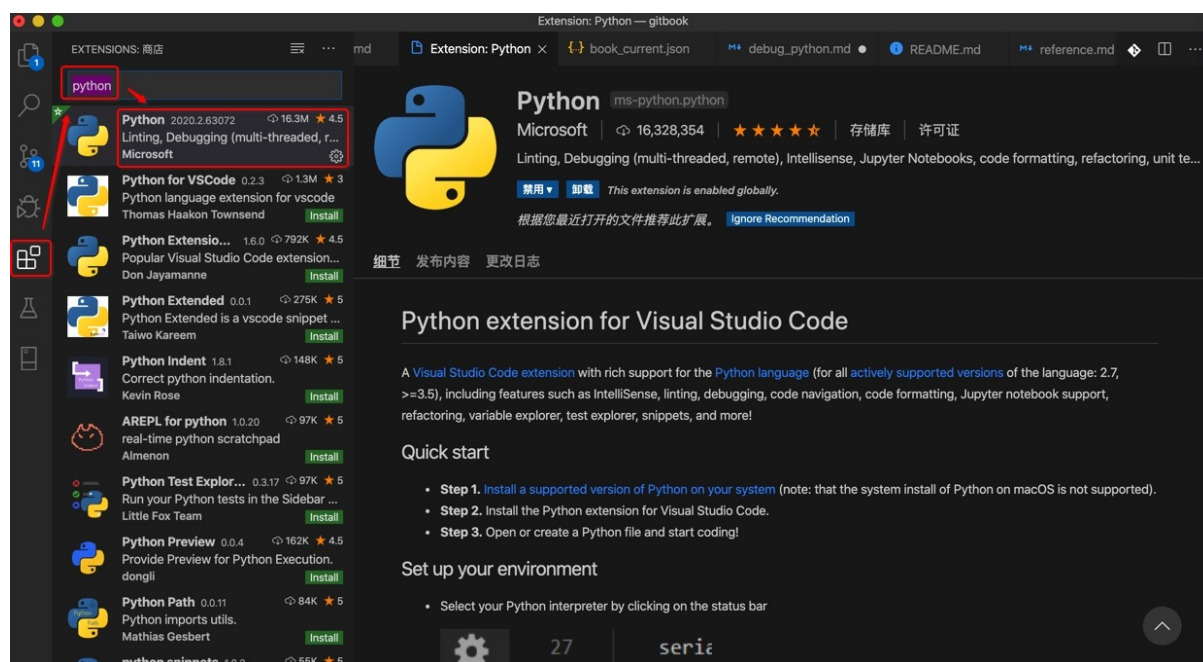
crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新： 2020-02-16 17:20:30

Mac中用VSCode调试Python

下面以Mac中为例，详细解释如何用VSCode去调试Python：

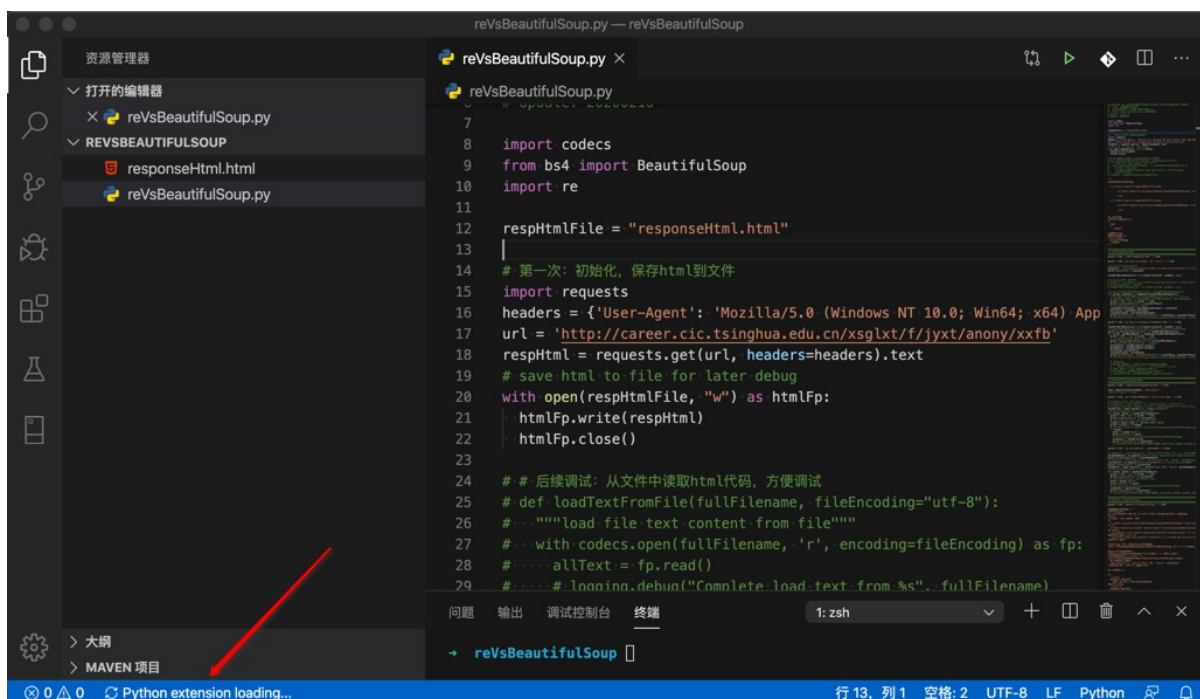
安装Python插件

在插件中输入 Python ，然后点击第一个匹配出来的插件中的 Install 安装即可，安装后：

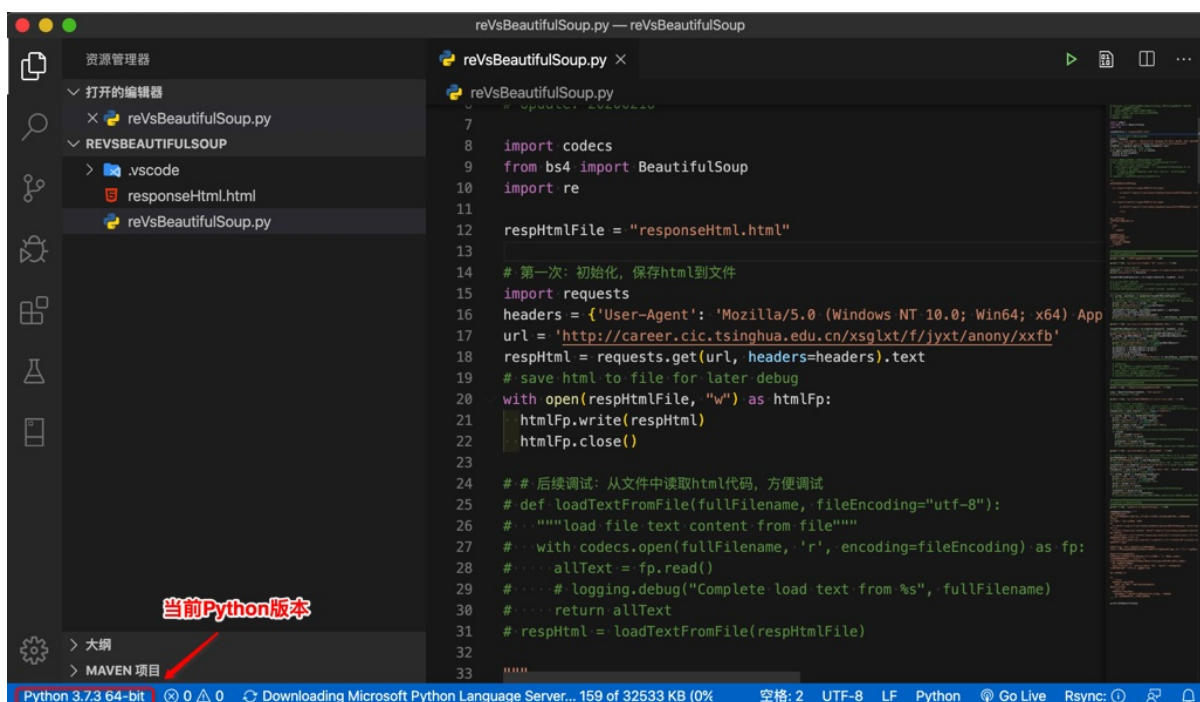


选择Python版本

每次用VSCode打开了Python项目后，左下角会出现： Python extension loading



表示正在加载Python插件，加载完毕后，左下角就会显示出某个版本的Python：



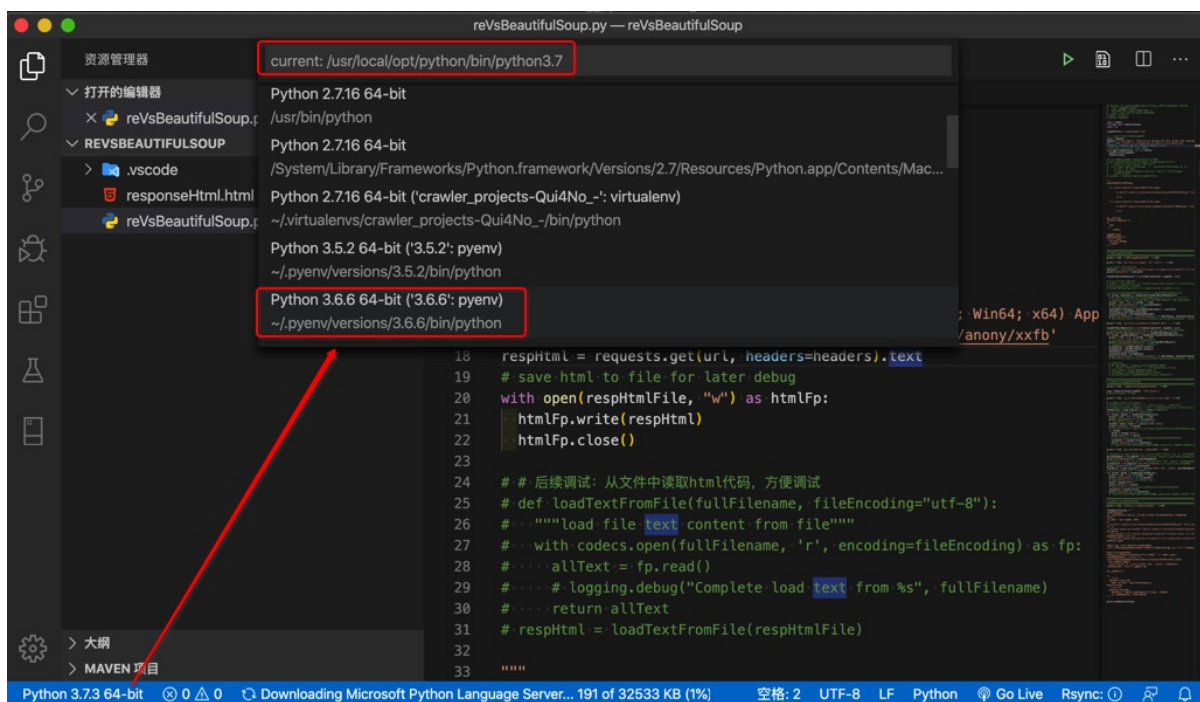
其含义是：当前项目所用的Python版本，此处是 Python 3.7.364-bit

意味着，后续调试Python代码，就是用的这个Python版本。

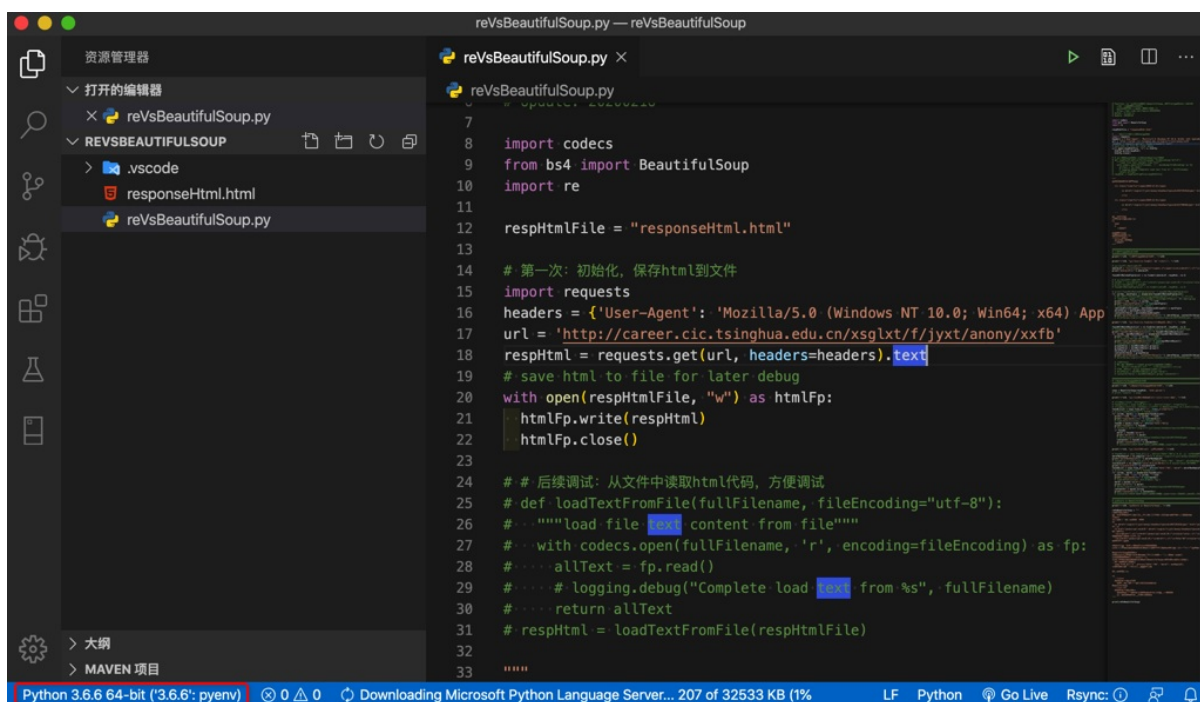
如果这个版本不是你所希望的，则需要：

选择Python版本 = 切换Python版本 = 设置Python解析器 = 设置Python虚拟环境（如果你用的是某个Python虚拟环境）

假如我此处，之前系统中有安装过另外的版本的 Python 3.6.6 64-bit，想要使用该版本，则点击左下角，会弹出列表，选择对应的版本



然后做下角即可显示你切换后的 Python 3.6.6 64-bit :



至此已切换成你要的Python版本了。

启动调试和调试配置

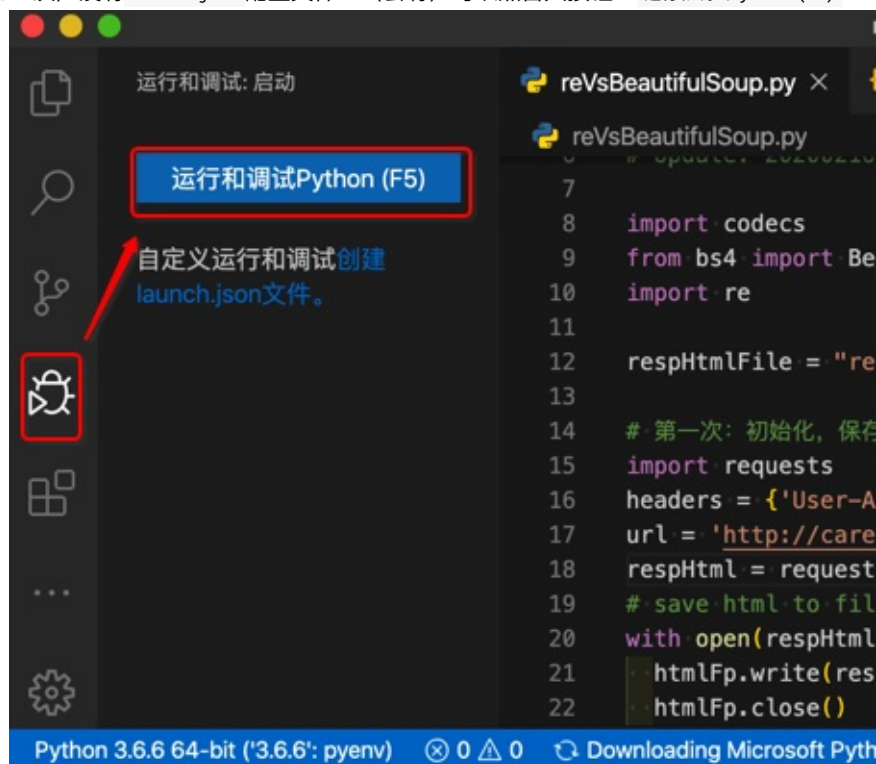
由于关系密切，此处把 启动调试 和 调试配置 放在一起解释。

启动调试

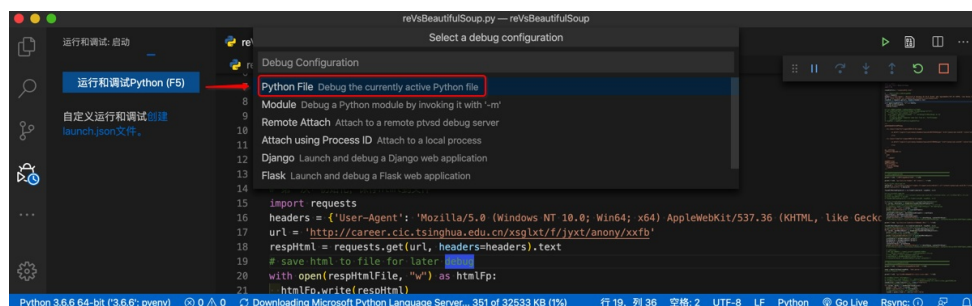
启动调试的有多种方式，效果都是一样的：

- 快捷键: F5
- 菜单: 调试 -> 启动调试 F5

-
- 通过 运行和调试 的tab页去调试
 - (第一次) 没有 launch.json 配置文件: (会有) 可以点击大按钮: 运行和调试Python (F5)



- 注: 且点击后, 每次都要选择调试的类型
 - 一般最常用的就是: Python File



- 表示去调试当前的Python文件
- 已有 launch.json 配置文件：有个绿色启动按钮和调试类型选项
 - 也就可以点击绿色调试按钮去调试了
 - 下面就是要去详细解释这部分内容

调试配置

开始调试之前，需要设置和添加好调试的配置。

对于调试的配置，或者说调试的类型的使用，一般有2种：

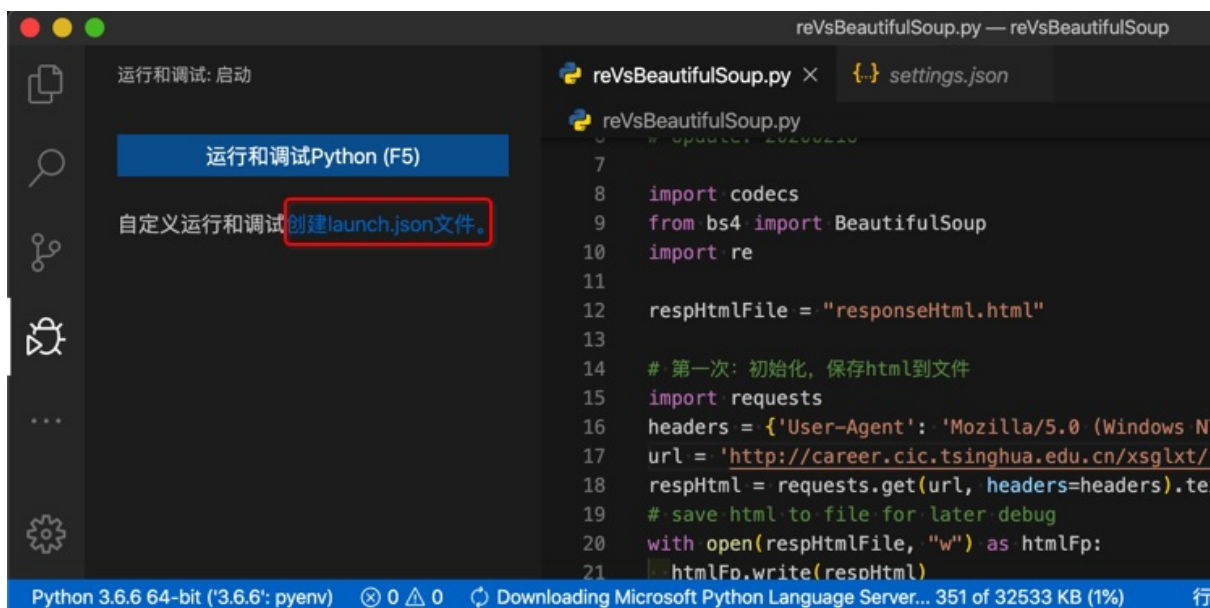
- 常用：一劳永逸型
 - 即：设置一次，每次都无需再设置
 - 在项目根目录中创建 `./vscode/launch.json` 配置文件
 - 在其中加上对应的配置
- 不太常用：每次都选择型
 - 即：不创建 `launch.json` 配置文件，每次启动都选择一次调试类型
 - 也就是上面已解释过的：
 - 点击 运行和调试Python (F5)
 - 选择调试类型
 - -> 很麻烦，不太推荐

所以此处主要去解释：

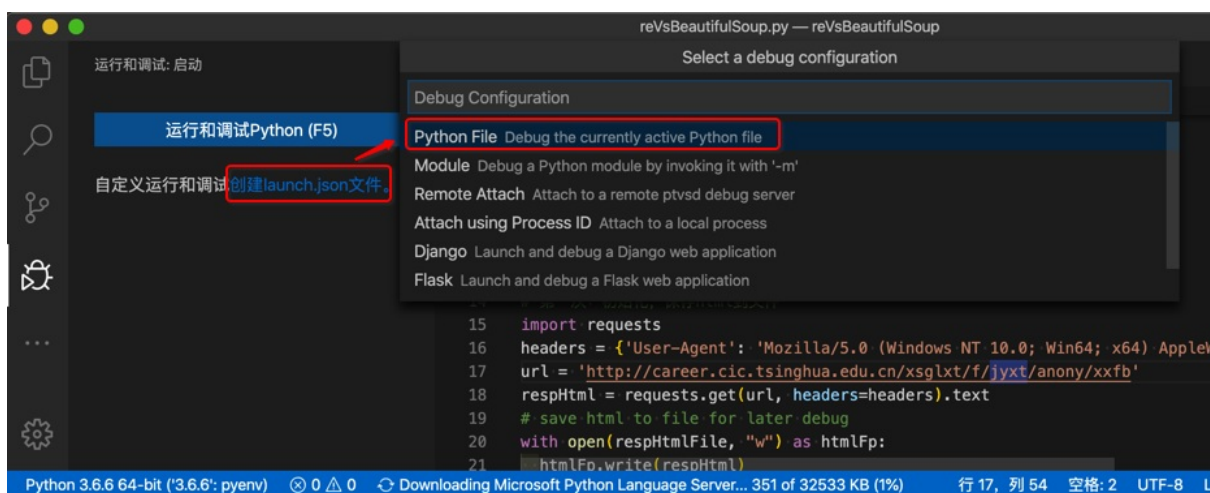
一劳永逸型：创建launch.json配置

自己手动在当前项目根目录下创建 `./vscode/launch.json` 配置文件

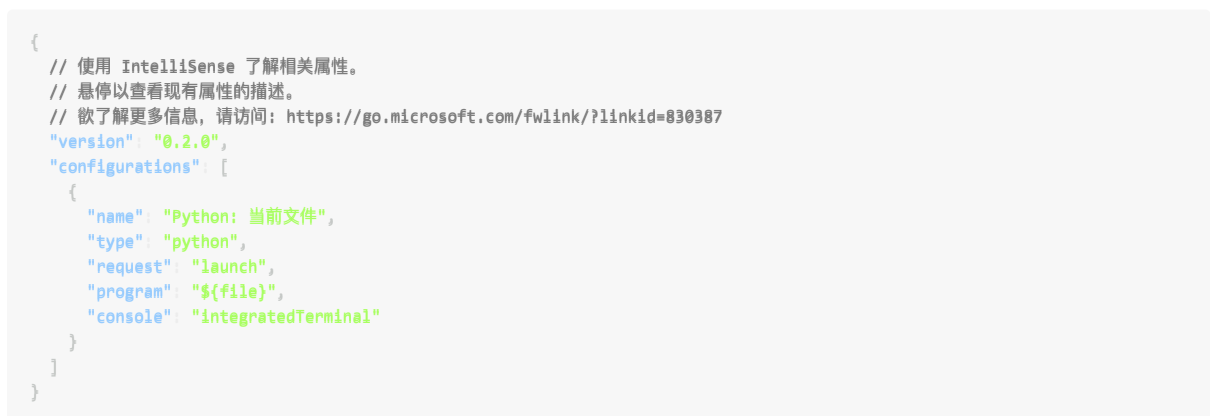
或者点击 创建launch.json文件：



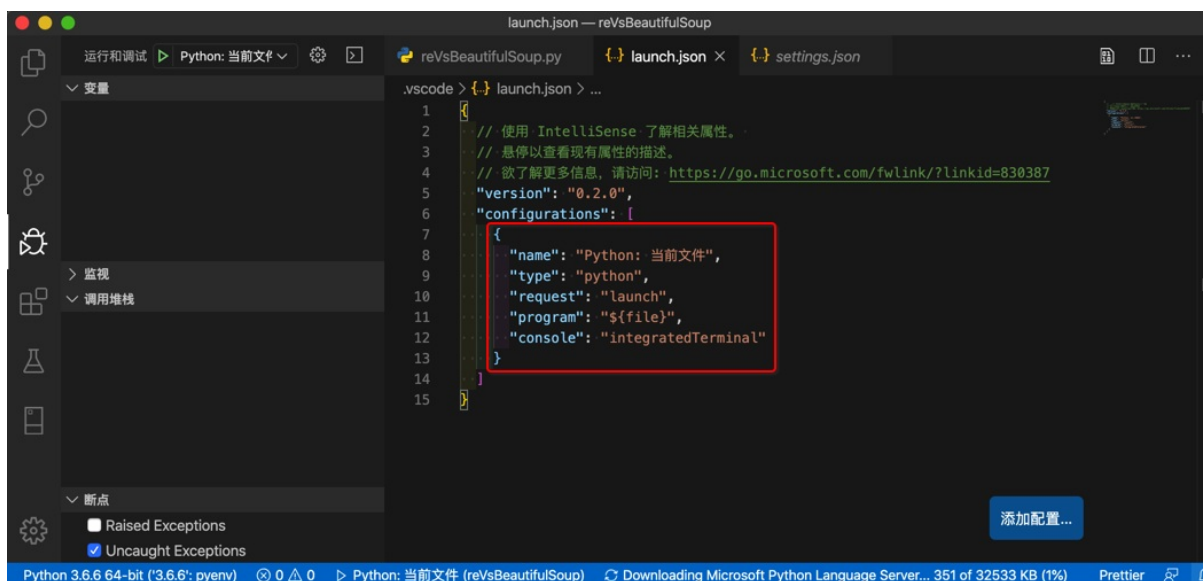
然后选择: Python File



会自动创建调试的配置文件 `./vscode/launch.json` , 并根据你前面所选择的调试类型 `Python File` 而生成对应的调试配置:



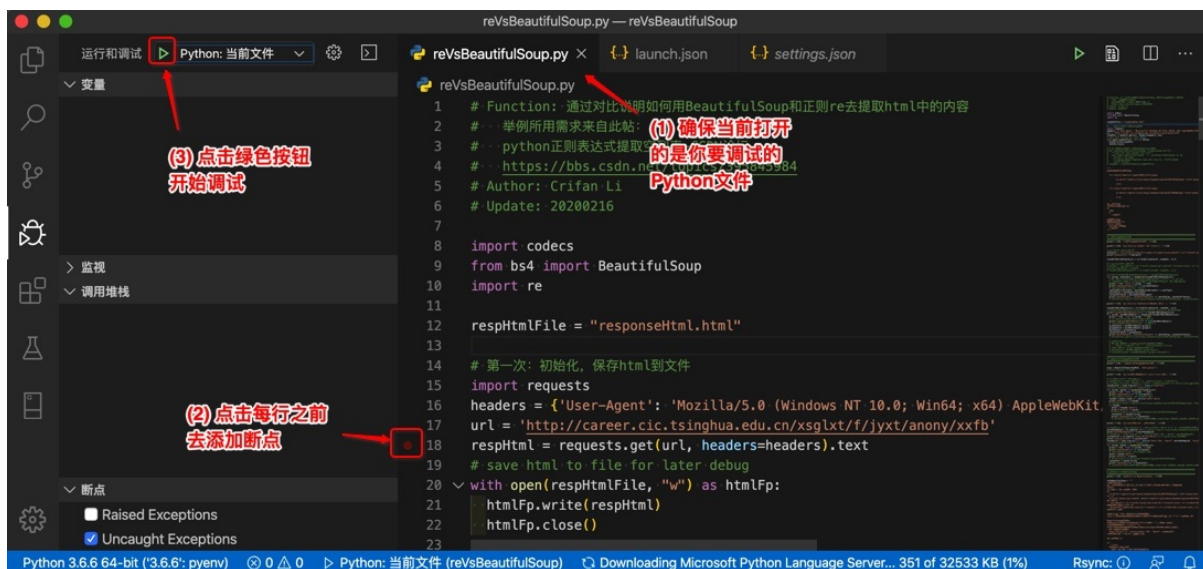
效果是:



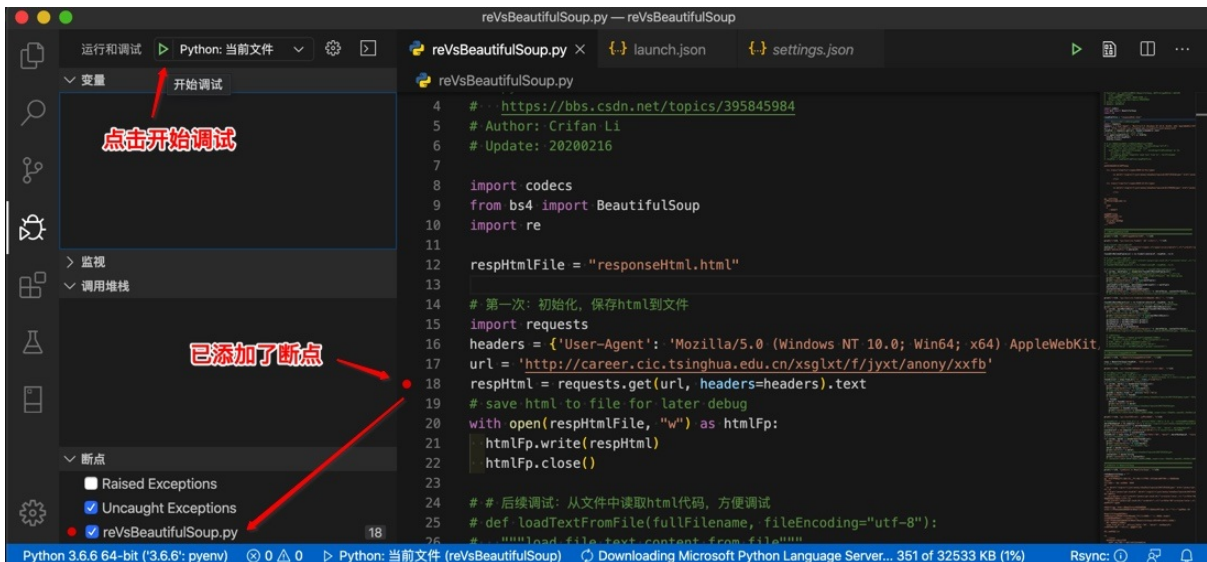
如此，就是创建完毕调试配置文件了。

之后就是去调试了：

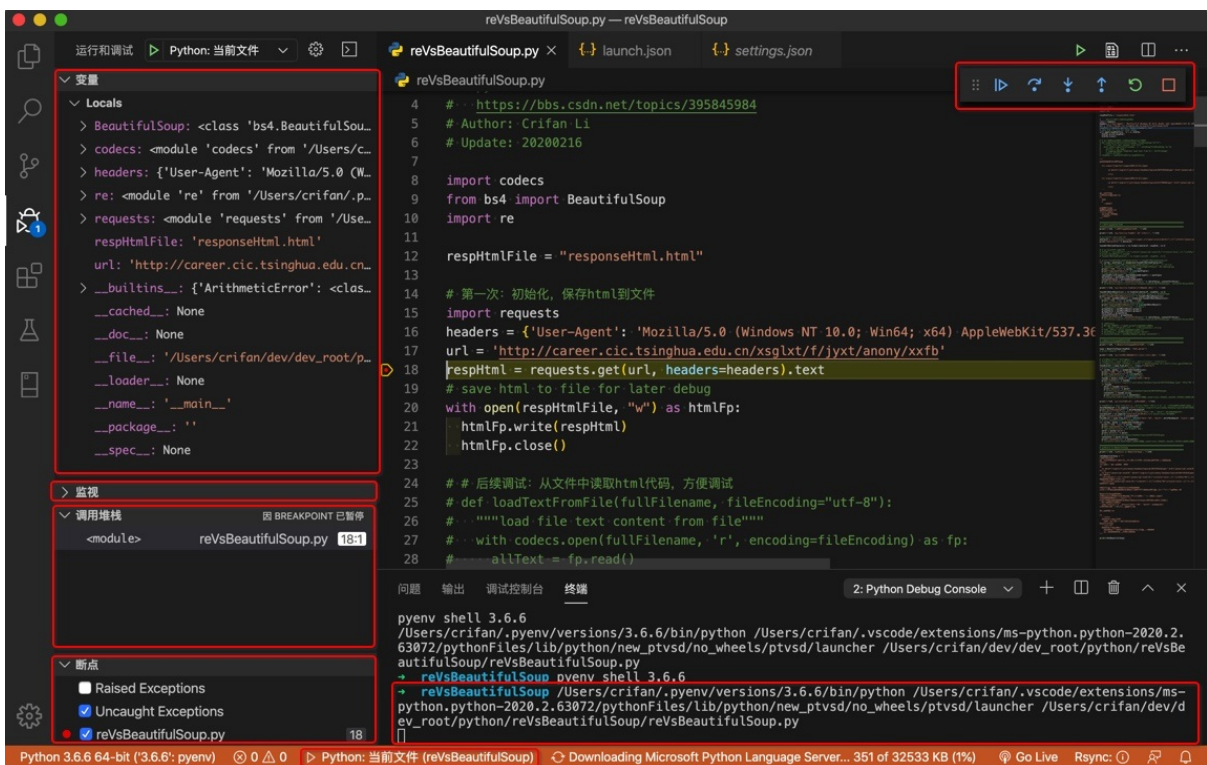
1. 确保VSCode当前打开的是你要调试的Python文件
2. 点击每行之前去添加断点
3. 点击绿色的调试按钮▶去开始调试



添加了断点后，可以看到左下角出现了断点的文件和行号，再去点击开始按钮▶去调试：

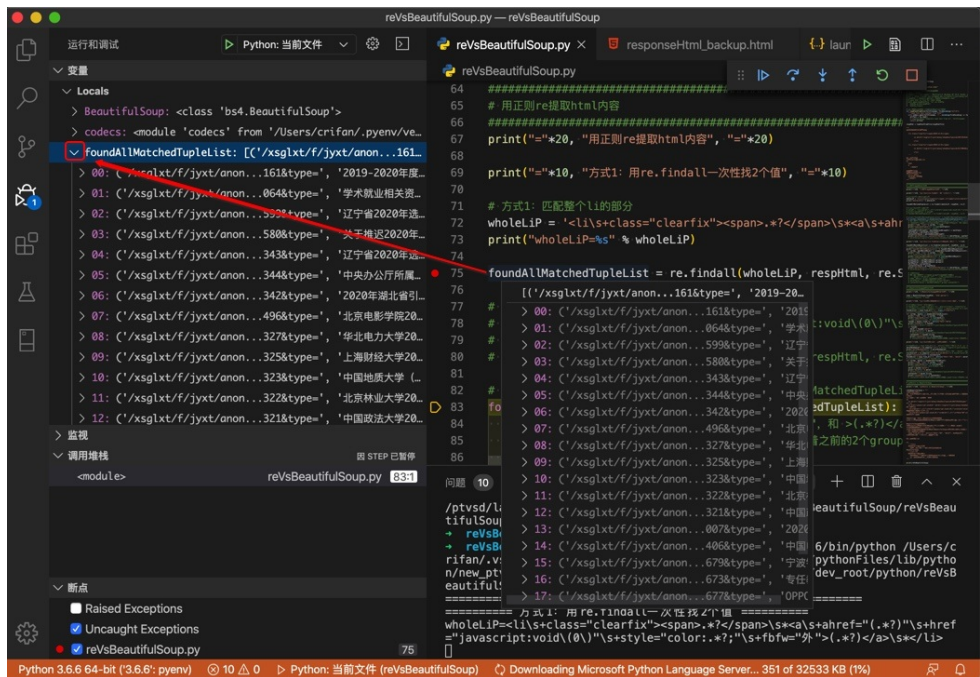


然后就可以看到，正在调试的界面和效果了：



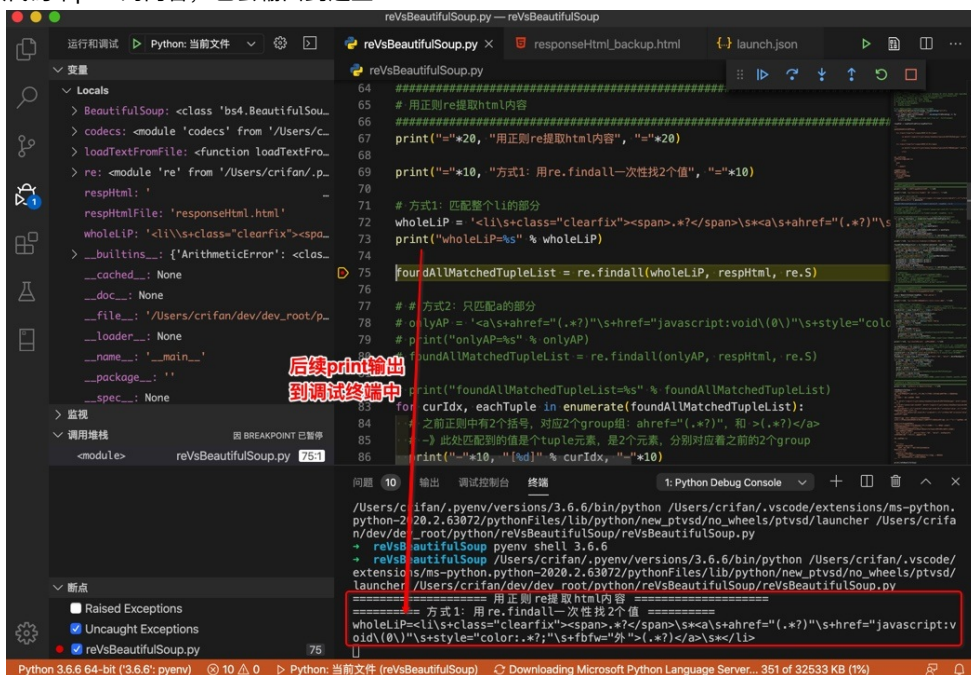
调试界面中的几个区域和对应功能，分别解释一下：

- 左边
 - 变量
 - 显示当前的各种变量值
 - 对于有属性的变量，可以点击绿色按钮，查看具体的值



- 监视Evaluation
 - 可以输入变量表达式，查看结果
- 调用堆栈
 - 查看文件调用的先后顺序
 - 在代码报错时，尤其有用
 - 知道出错时，文件调用的先后顺序，才容易找出错误位置和原因
- 断点
 - 之前自己添加的所有的断点
 - 2个选项：属于高级功能
 - Rased Exceptions=抛出的异常
 - 调试期间代码发生异常exception时，代码执行就会暂停
 - 就像打了断点的效果一样
 - 即使你代码中用try except去捕获了异常，发生了异常，也会暂停
 - Uncaught Exceptions=未捕获的异常
 - 调试期间，代码中没有捕获到的异常，即没有try except对应异常，则此时就会暂停
- 右上角
 - 调试工具条
 - 继续(F5)
 - 单步跳过(F10)
 - 单步调试(F11)
 - =单步进入
 - 单步跳出(Shift+F11)
 - 重启(Shift+Command+F5)
 - 停止(Shift+F5)
- 右下角
 - 调试输出
 - 最开始会输出，启动调试时所用的命令
 - 此处是
 - /Users/crifan/.pyenv/versions/3.6.6/bin/python /Users/crifan/.vscode/extensions/ms-python.python-2020.2.63072/pythonFiles/lib/python/new_ptvsd/no_wheels/ptvsd/launcher /Users/crifan/dev/dev_root/python/reVsBeautifulSoup/reVsBeautifulSoup.py
 - /Users/crifan/.pyenv/versions/3.6.6/bin/python：是当前所用的Python

- /Users/crifan/.vscode/extensions/ms-python.python-2020.2.63072/pythonFiles/lib/python/new_ptvsd/no_wheels/ptvsd/launcher 是当前所用的启动器
- /Users/crifan/dev/dev_root/python/reVsBeautifulSoup/reVsBeautifulSoup.py : 当前被调用的, 被调试的Python文件
- 后续代码中print的内容, 也会输出到这里



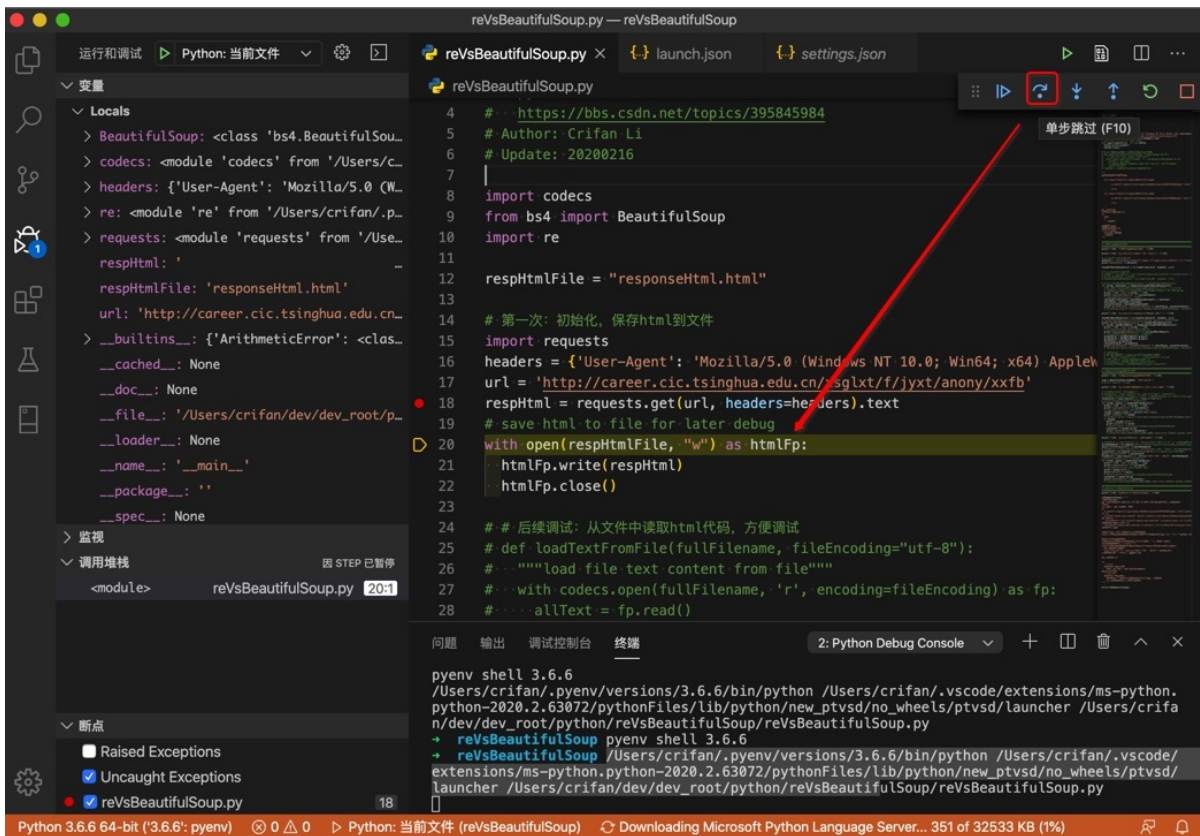
然后继续正常调试

开始调试

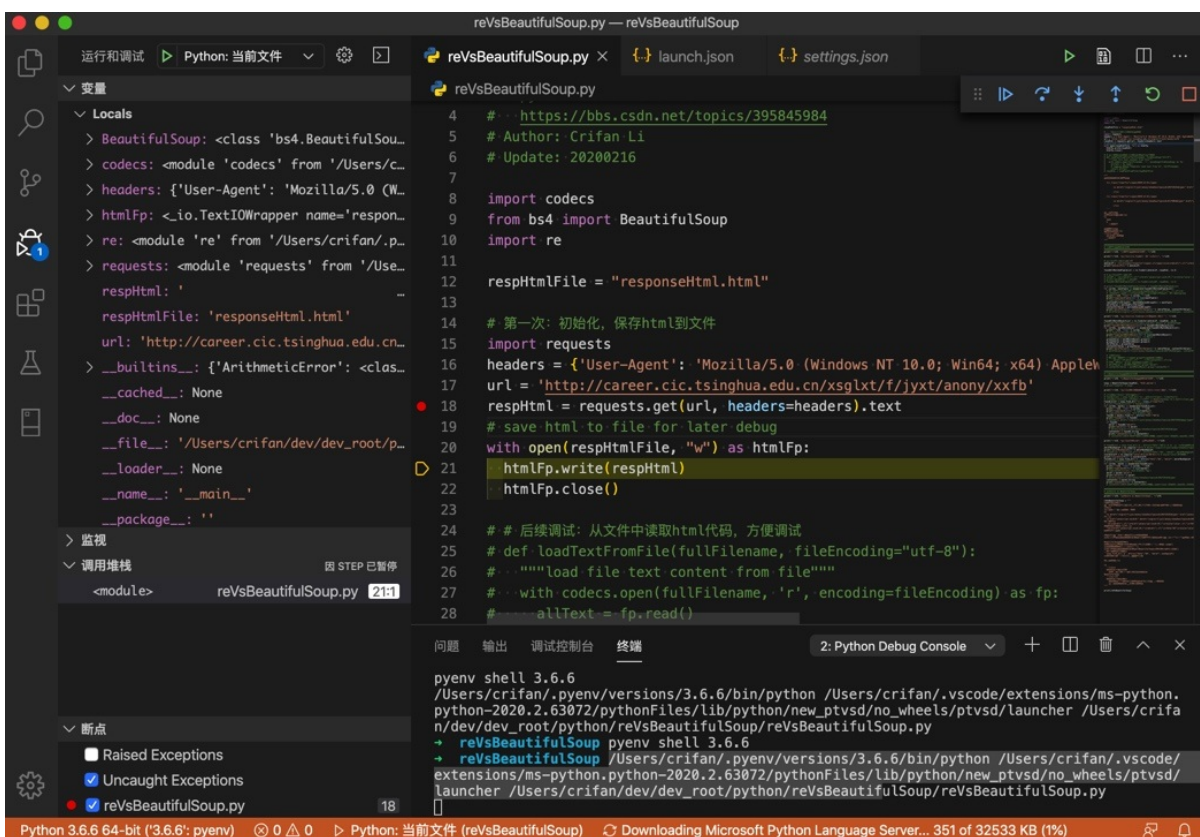
开始了调试之后, 就可以继续正常调试了。

单步跳过(F10)

继续按 F10 或点击调试工具栏中的按钮, 即可运行当前代码, 然后进入下一行并暂停了:

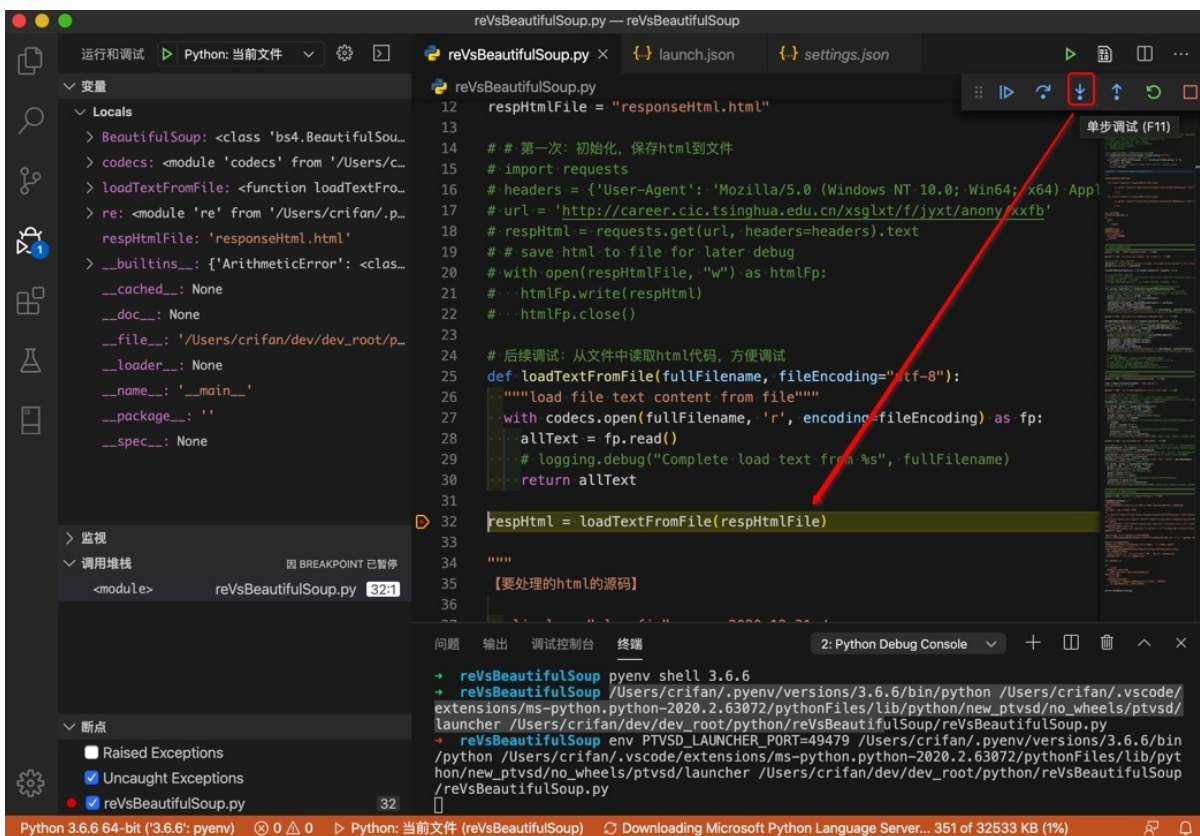


如此, 继续一直 F10 单步调试即可:

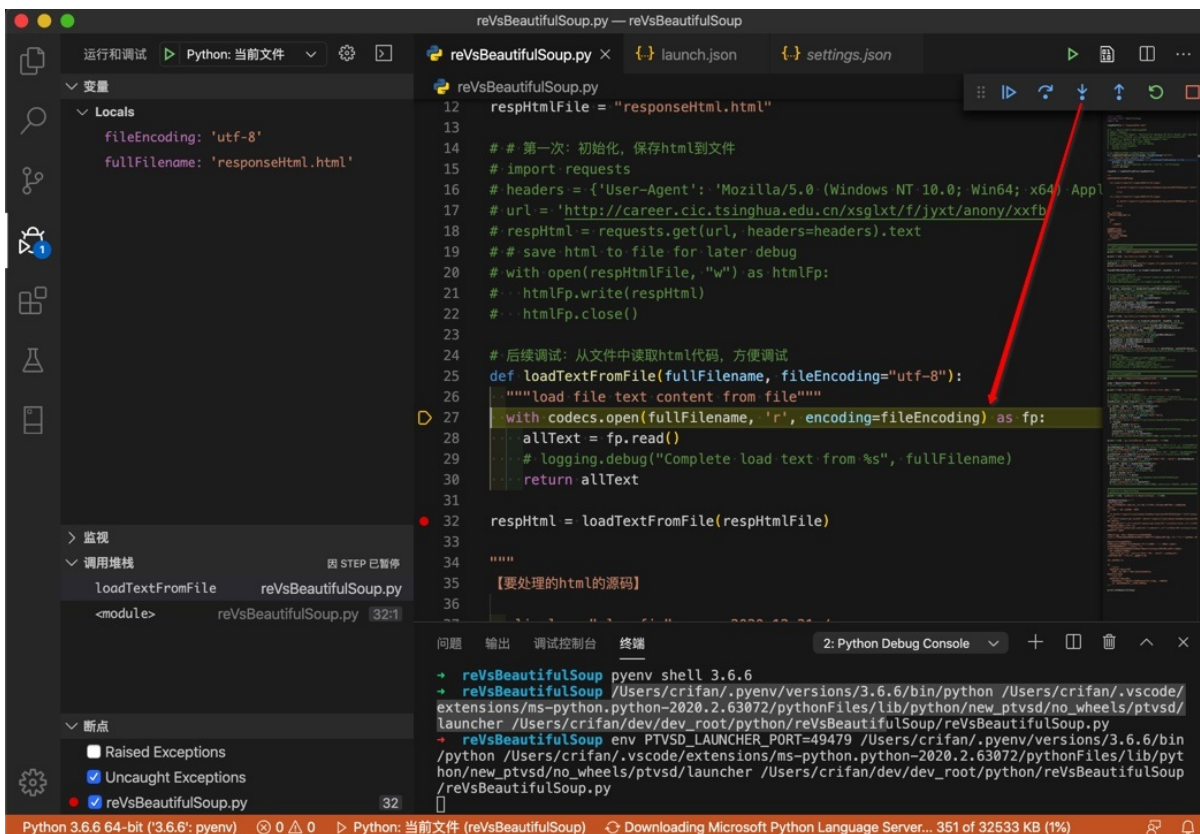


单步调试(F11)

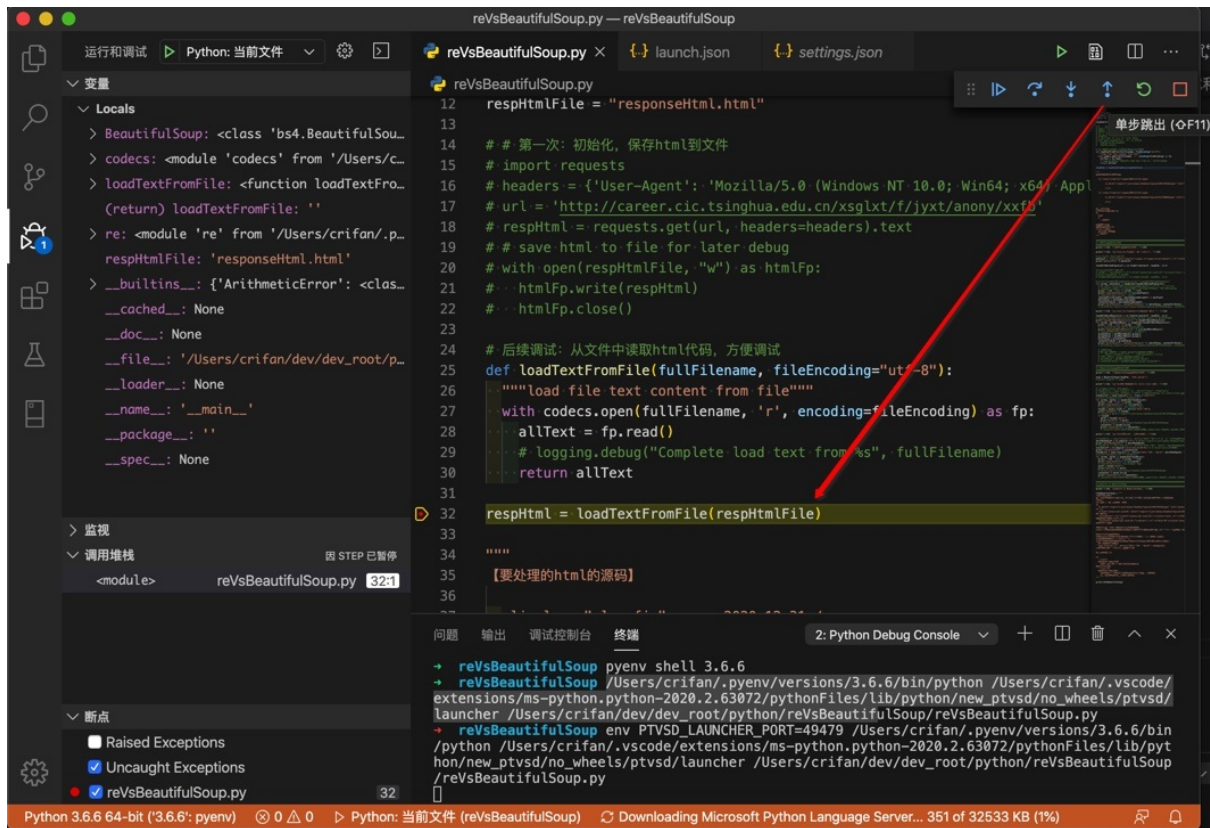
而对于如果遇到函数调用：



则可以用 F11 单步进入该函数中去调试：



而对应的 Shift+F11 是去跳出当前执行的范围，此处即跳出当前函数，即让当前函数返回：

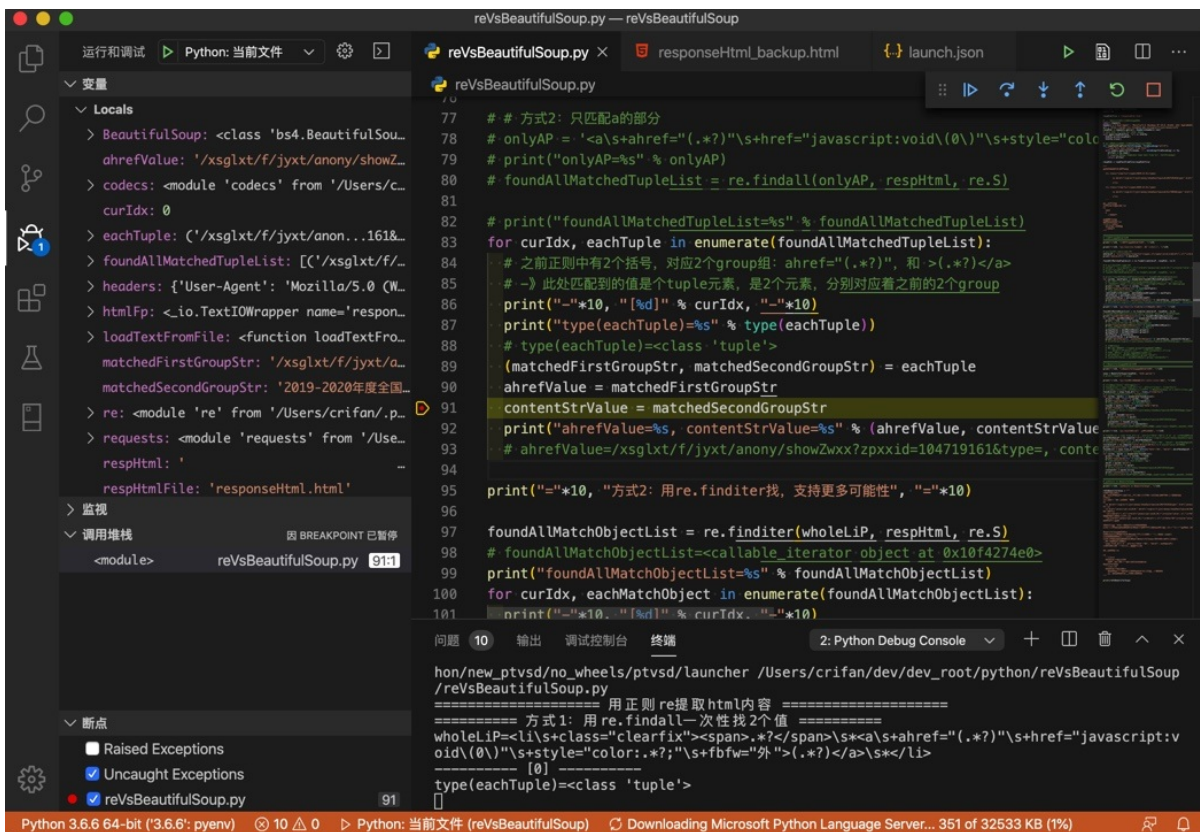


之后即可继续调试。

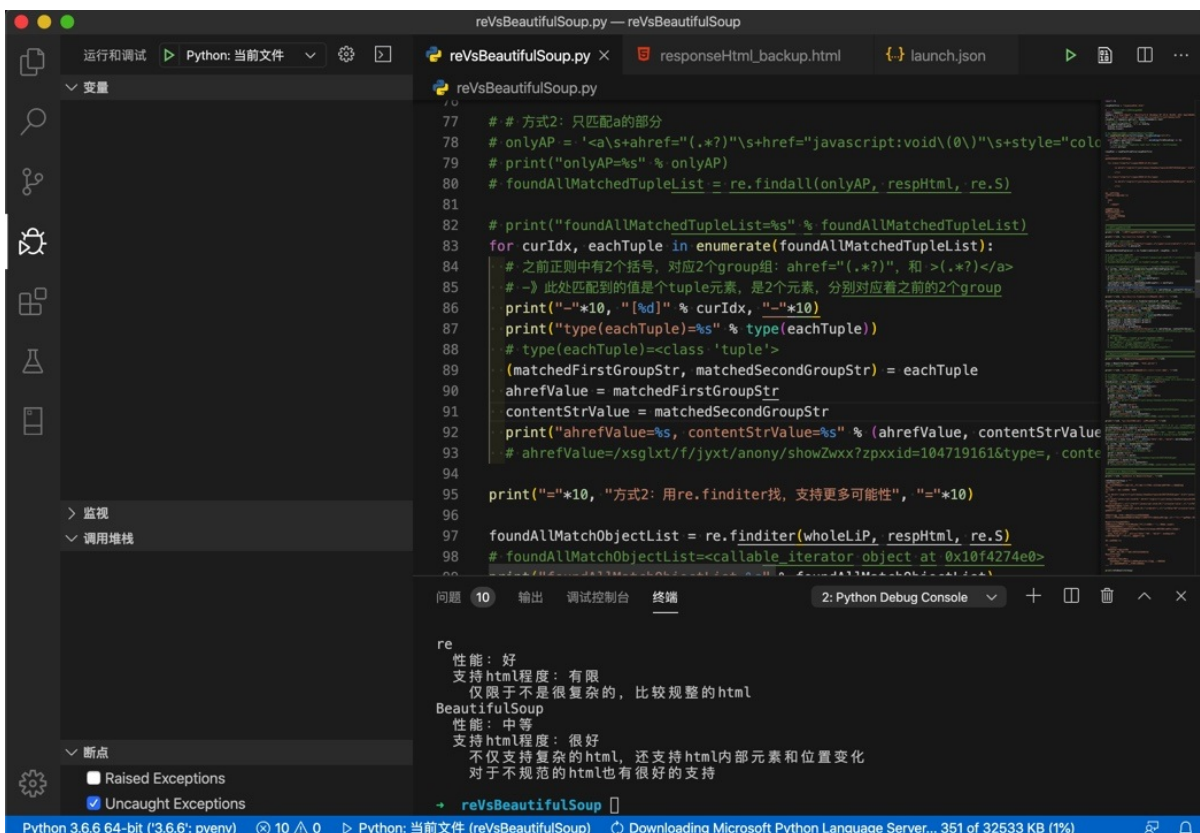
F5=继续调试

如果调试发现前面代码没问题，那么就可以让其继续运行

就可以F5继续调试，直到新的断点才暂停：



如果后续没有断点，则程序运行完毕，自动停止运行：



重启(Shift+Command+F5)

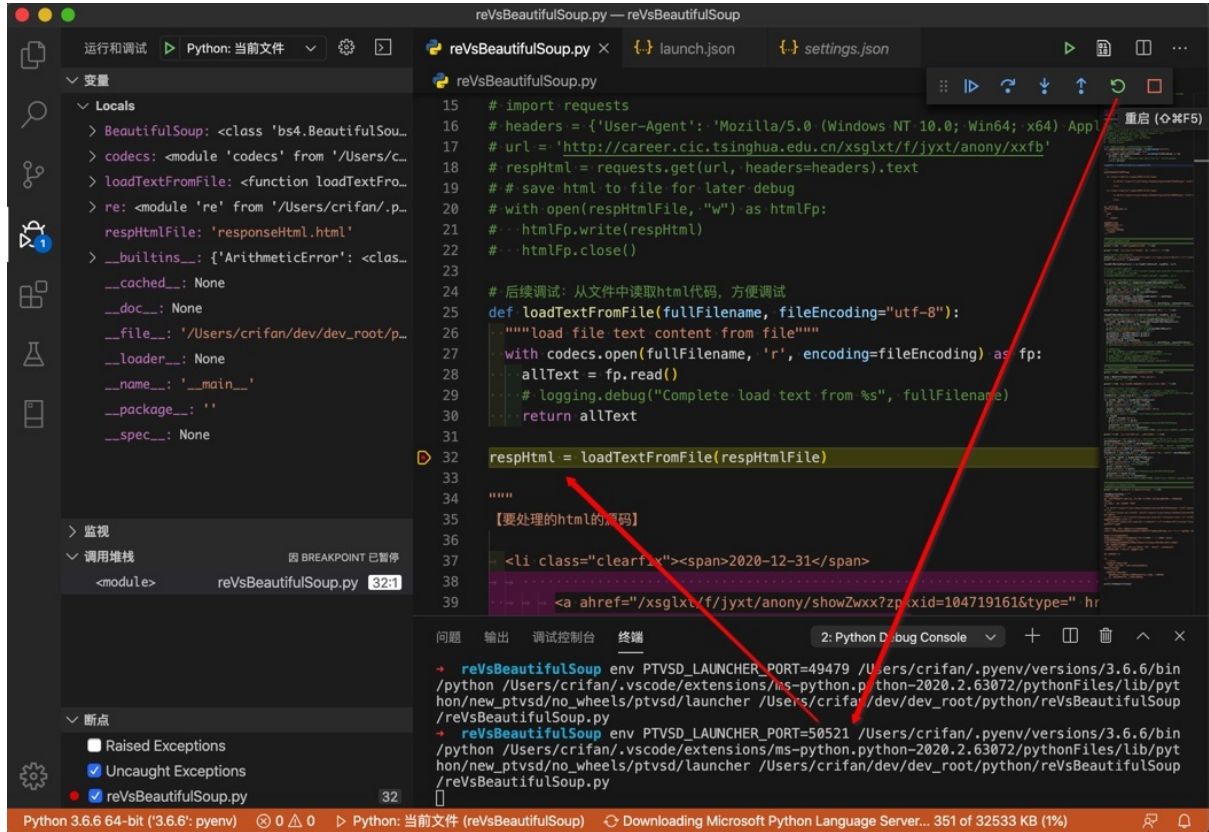
对于重启调试，一般能用到的场合是：

调试期间，错过了想要查看的内容。

或者是调试期间，发现代码有问题，顺带在调试期间，就把代码改好了，然后就可以去

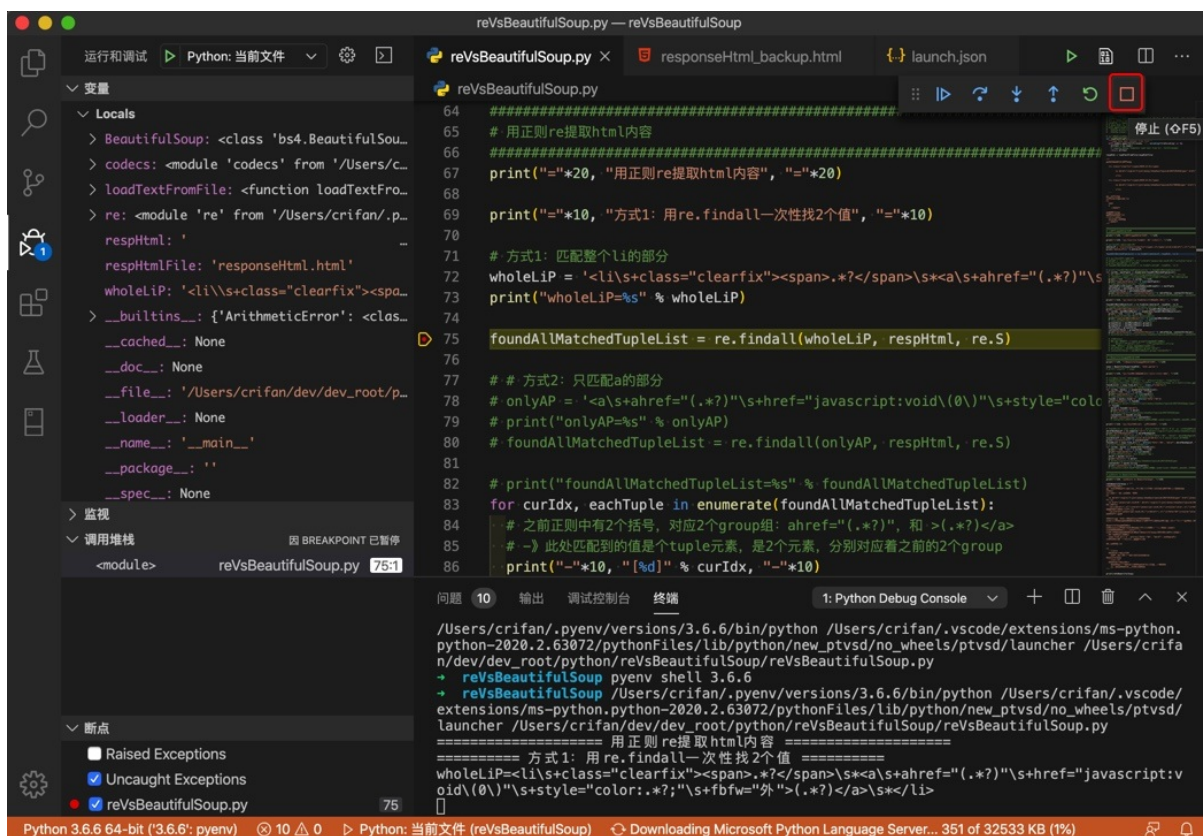
点击 重启，会让调试重新启动，并运行到（你所设置的断点）的位置

此处可以从终端的log看出，是重启启动了一次调试：



停止(Shift+F5)

如果不想要调试了，则可以点击停止：



即可停止调试。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2020-02-16 17:33:16

Windows中用VSCode调试Python

其实和前面已经详细解释的[Mac中调试Python](#)的步骤没区别。

但是为了别人更加方便参考，所以也专门整理如下：

- [【已解决】 windows中安装VSCode](#)
- [【记录】 Windows中新建VSCode的Python项目](#)
- [【记录】 Windows中用VSCode去调试Python项目代码文件](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2020-02-16 17:22:32

经验心得

VSCode调试Python期间，有些需要注意的事情和一些心得，整理如下。

注意事项

不要轻易根据提示切换Python版本

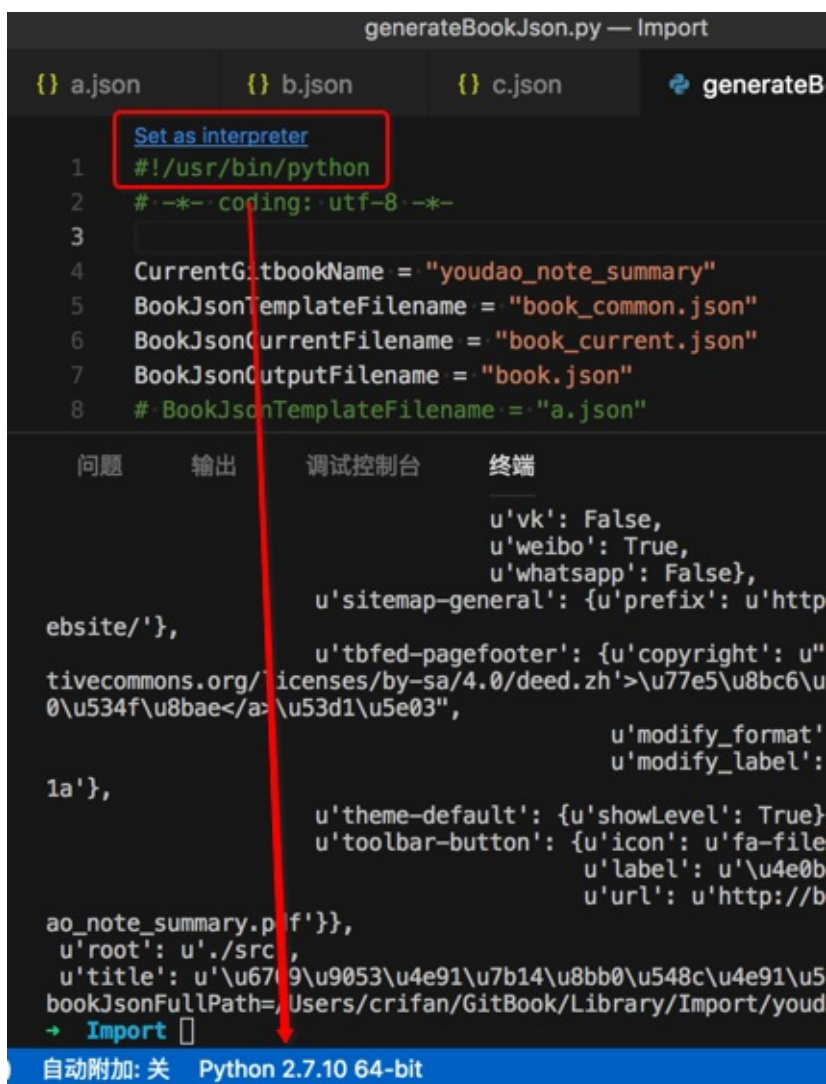
之前遇到python文件中顶部有常见的写法：

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
```

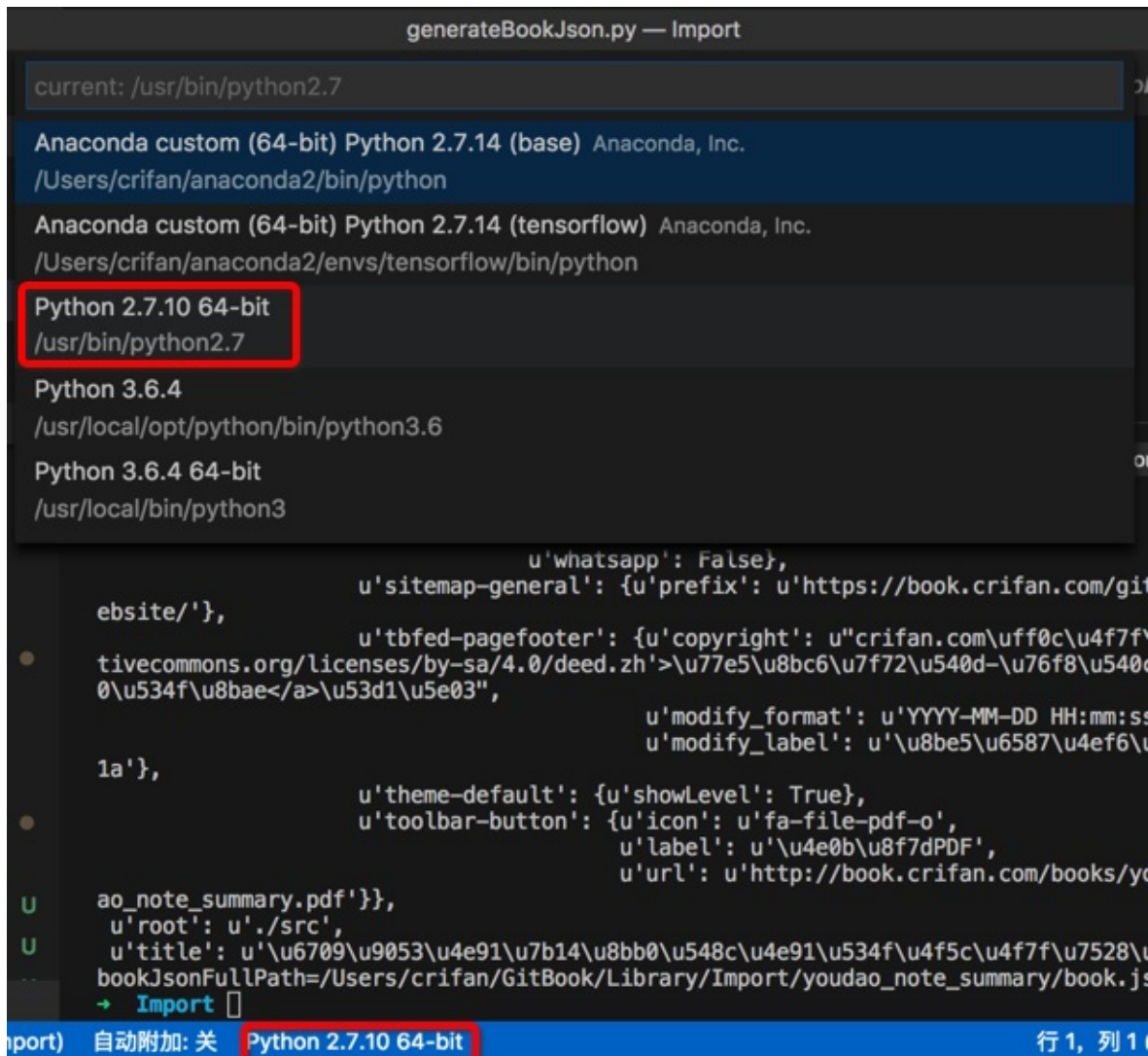
其中第一行是建议的Python解释器的位置

对此，VSCode的Python插件自动检测处理，然后给出提示 `Set as interpreter`

你如果点击了：



则就可以从当前的解析器：



换成代码中设置的解析器了：


```

238 # topicMatch = re.search("(?P<topicLevel1>[\w\s]+)\s*([-|~|~]\s*(?P<topicLevel2>[\w\s]+))?", topic)
239 # sports - skiing
240 # topicMatch = re.search("(?P<topicLevel1>[\w\s]+)\s*([-|~|~]\s*(?P<topicLevel2>[\w\s]+))?", topic)
241 # Sports - tae-kwon-do
242 topicMatch = re.search("(?P<topicLevel1>[\w\s]+)\s*([-|~|~]\s*(?P<topicLevel2>[\w\s]+))?", topic)
243 if topicMatch:
244     topicLevel1 = topicMatch.group("topicLevel1")
245     topicLevel1 = topicLevel1.strip()
246     isExisted, realTopicLevel1 = checkTopic(topicLevel1)
247     if isExisted:
248         curScriptDict["topic"] = realTopicLevel1
249
250     topicLevel2 = topicMatch.group("topicLevel2")
251     if topicLevel2:
252         topicLevel2 = topicLevel2.strip()
253         isExisted, realTopicLevel2 = checkTopic(topicLevel2, isChild=True, parentTopicName=realTopicLevel1)
254         if isExisted:
255             curScriptDict["second_level_topic"] = realTopicLevel2
256         else:
257             logging.error("Can not find real child topic %s from parent %s", topicLevel2, realTopicLevel1)
258     else:
259         logging.warning("Not found topic level 2 from topic string: %s" % topic)
260 else:
261     logging.error("Can not find level1 topic: %s", topicLevel1)

```

只能再：去掉之前断点，重新打断点：

```

240 # topicMatch = re.search("(?P<topicLevel1>[\w\s]+)\s*([-|~|~]\s*(?P<topicLevel2>[\w\s]+))?", topic)
241 # Sports - tae-kwon-do
242 topicMatch = re.search("(?P<topicLevel1>[\w\s]+)\s*([-|~|~]\s*(?P<topicLevel2>[\w\s]+))?", topic)
243 if topicMatch:
244     topicLevel1 = topicMatch.group("topicLevel1")
245     topicLevel1 = topicLevel1.strip()
246     isExisted, realTopicLevel1 = checkTopic(topicLevel1)
247     if isExisted:
248         curScriptDict["topic"] = realTopicLevel1
249
250     topicLevel2 = topicMatch.group("topicLevel2")
251     if topicLevel2:
252         topicLevel2 = topicLevel2.strip()
253         isExisted, realTopicLevel2 = checkTopic(topicLevel2, isChild=True, parentTopicName=realTopicLevel1)
254         if isExisted:
255             curScriptDict["second_level_topic"] = realTopicLevel2
256         else:
257             logging.error("Can not find real child topic %s from parent %s", topicLevel2, realTopicLevel1)
258     else:
259         logging.warning("Not found topic level 2 from topic string: %s" % topic)
260 else:
261     logging.error("Can not find level1 topic: %s", topicLevel1)
262 else:
263     logging.error("!!! Can not recognize topic format for: %s" % topic)
264
265 content = eachScriptMatch.group("content")

```

所以还是有点麻烦的

-> 而PyCharm就可以很好的支持：当代码改动（不多）的时候，可以自动保持原有的断点的位置。

-> 不过后来也发现，此问题只是偶尔发生的。有时候代码改动后，断点还正常的。

心得

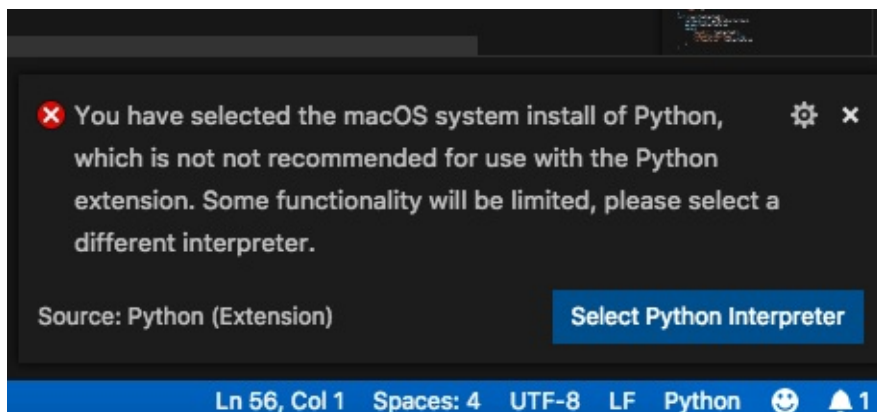
安装了虚拟环境后提示你切换到对应版本

对于某个Python项目，在：

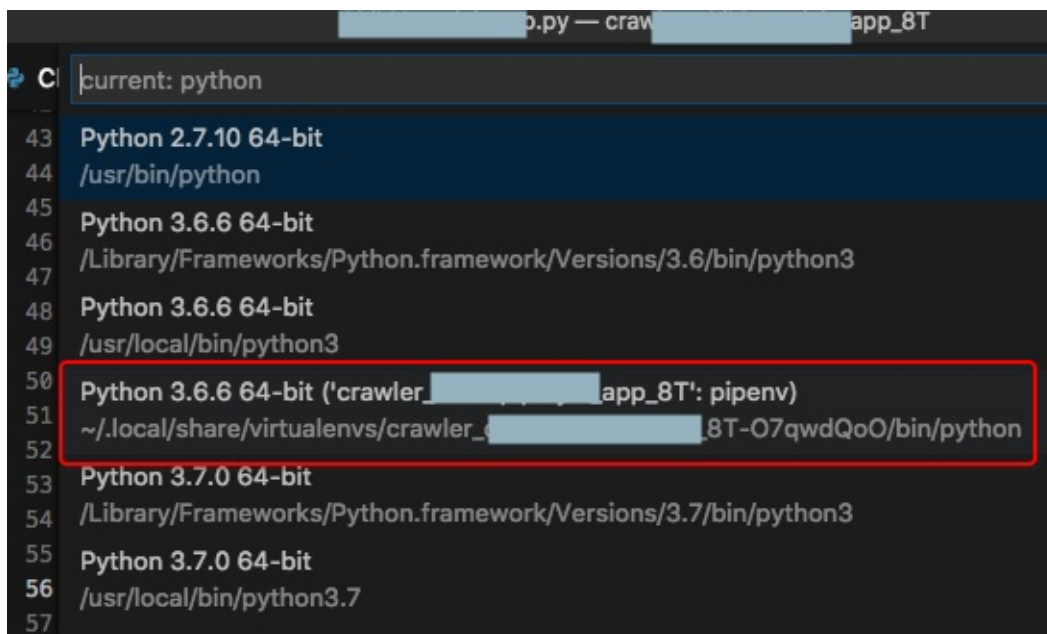
```
pipenv install
```

安装了虚拟环境后，重新用VSCode打开该项目，会提示你：

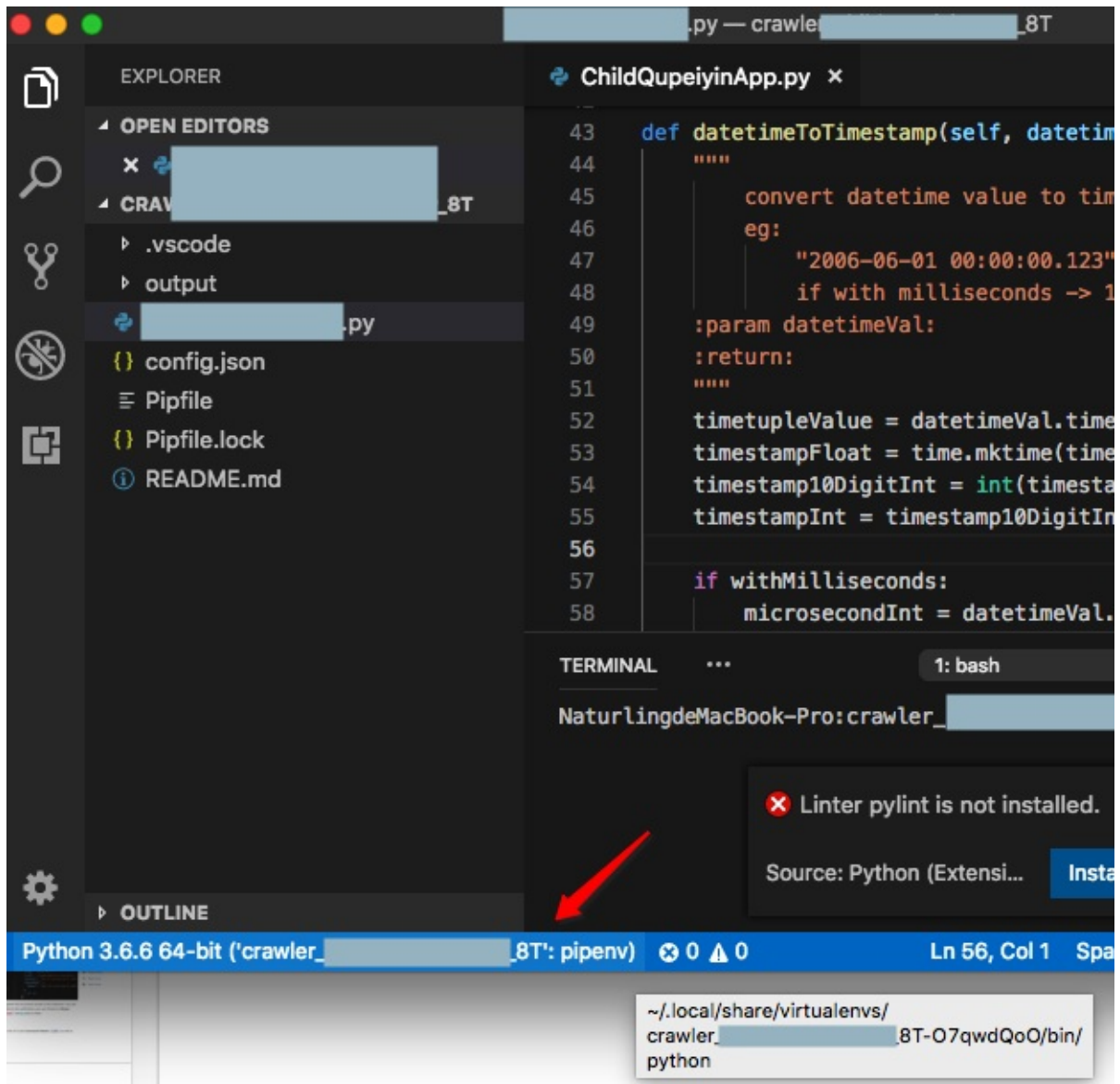
You have selected the macOS system install of Python, which is not recommended for use with the Python extension. Some functionality will be limited, please select a different interpreter.



然后点击 Select Python Interpreter，去选择刚装好的虚拟环境中的python：



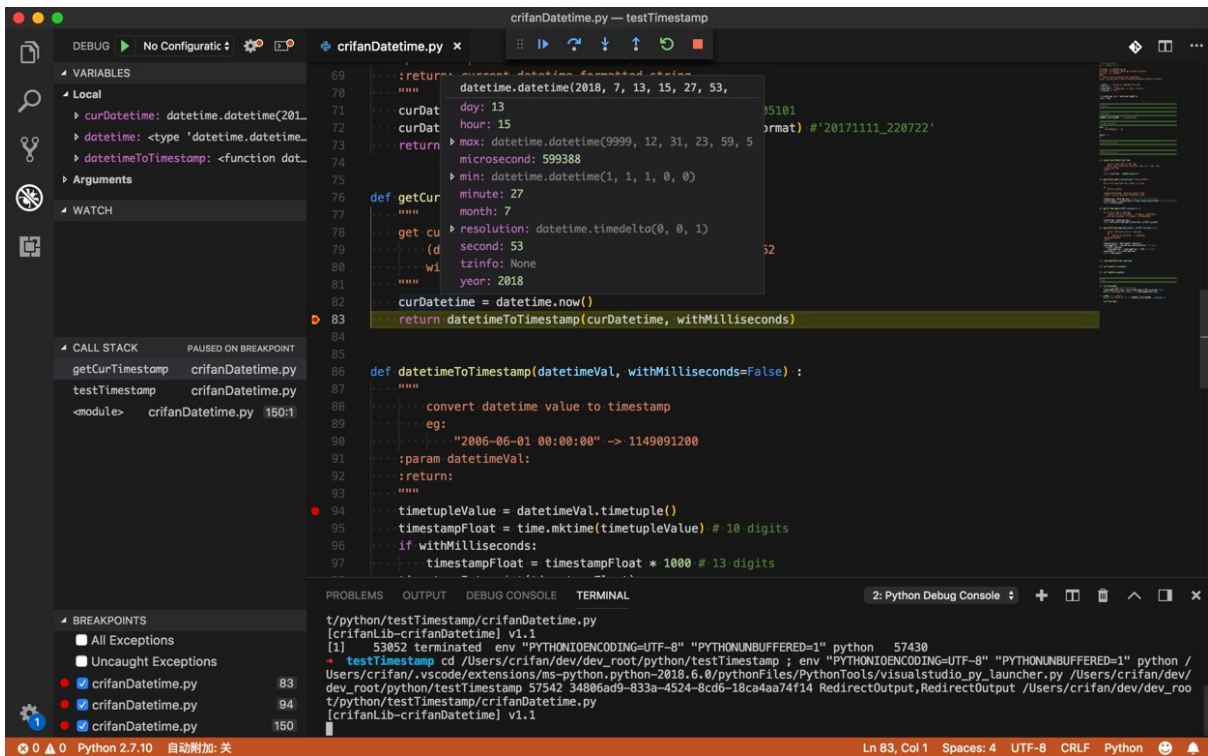
选择后，左下角就可以显示出当前所选Python了：



说明VSCode的Python插件还是很智能的，提示你新切换Python版本到你所安装的虚拟环境的版本。

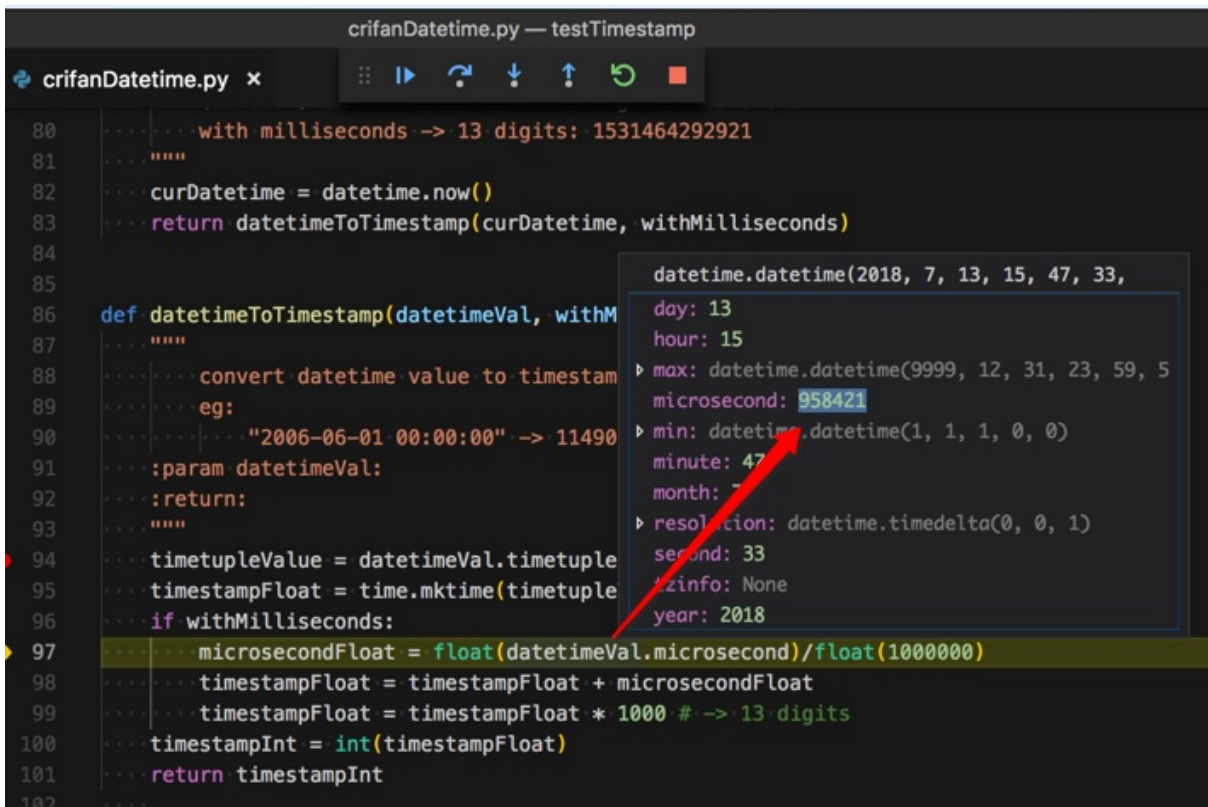
鼠标移动上去可以查看变量值

后来又多次使用VSCode去调试Python：



鼠标移动到变量（类）的属性上，支持（直接）显示变量的属性的值：

比如datetime



的microsecond：

```

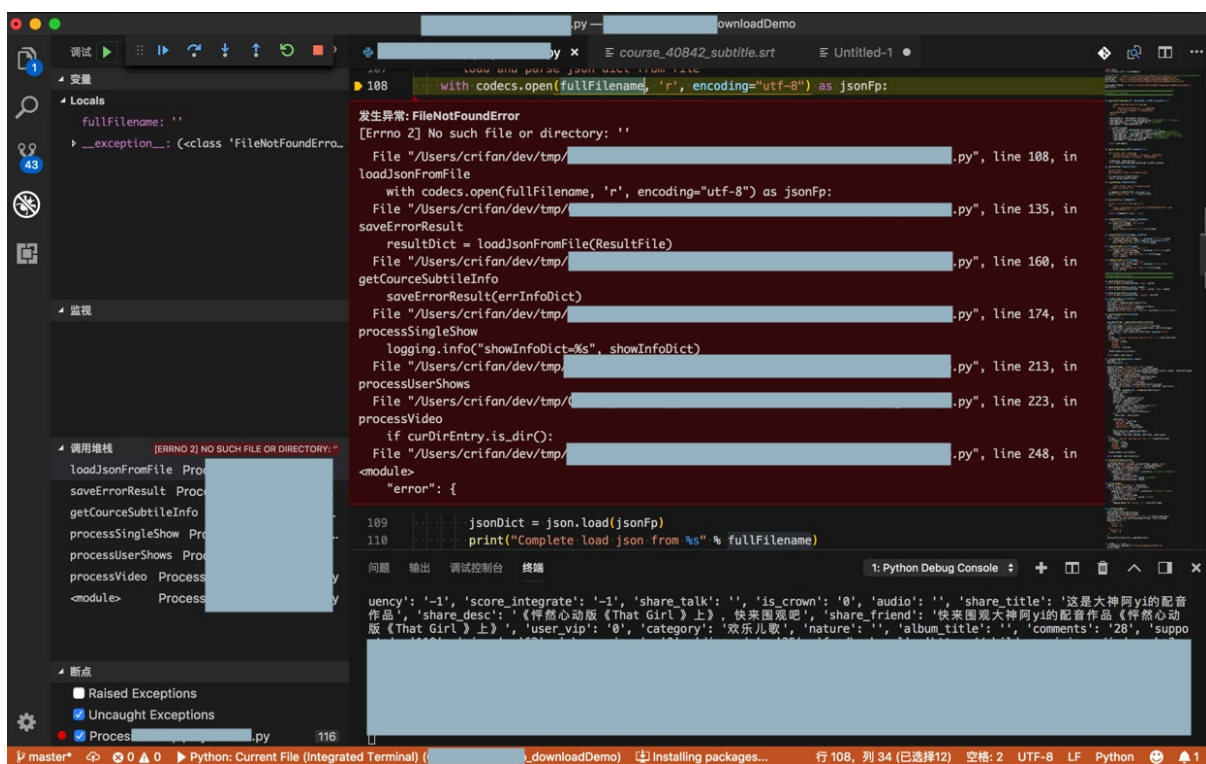
92     ... return:
93     ... """
94     ... timetupleValue = datetimeVal.timetuple()
95     ... timestampFloat = time.mktime(timetupleValue) # -> 10 digits
96     ... if withMilliseconds: 958421
97     ...     microsecondFloat = float(datetimeVal.microsecond)/float(1000000)
98     ...     timestampFloat = timestampFloat + microsecondFloat
99     ...     timestampFloat = timestampFloat * 1000 # -> 13 digits
100    ... timestampInt = int(timestampFloat)
101    ... return timestampInt

```

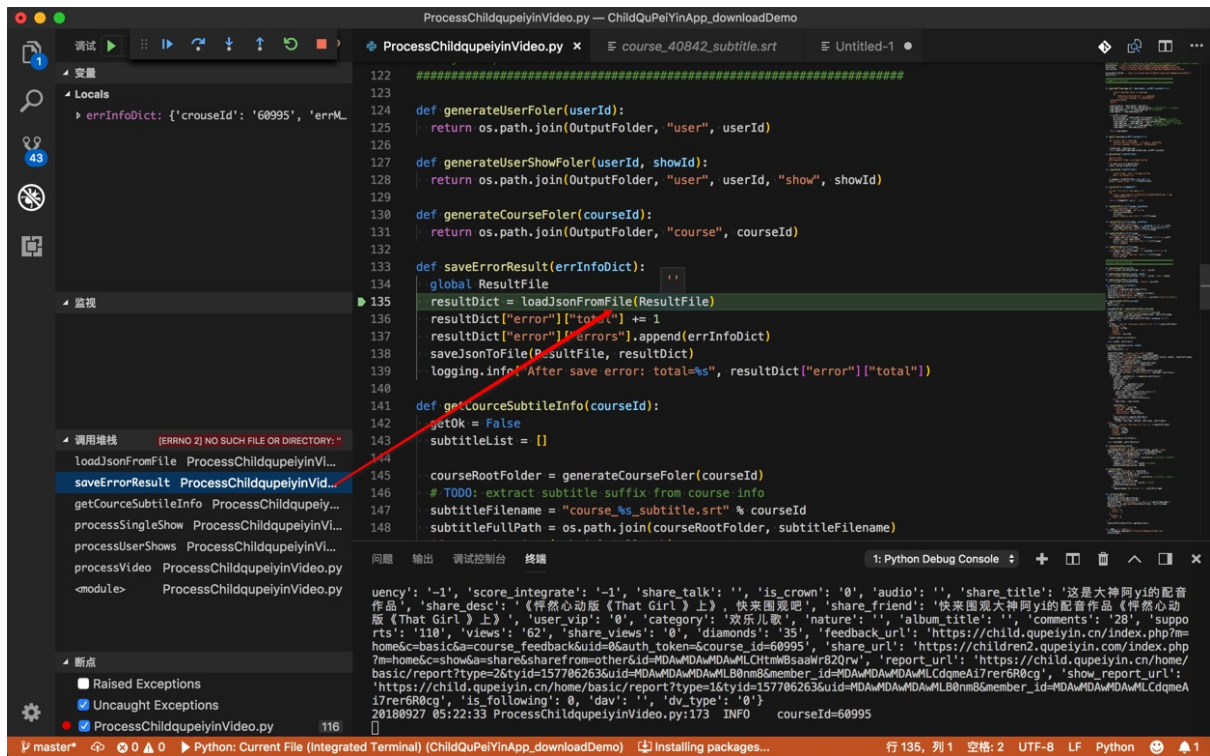
而这类功能，之前只有比较高级的IDE，比如 Visual Studio，PyCharm 等才支持。

支持异常信息的显示

当调试代码时发生异常，则可以方便快速的显示异常堆栈错误信息：



且可以点击左下角的 调用堆栈，调转到对应代码位置：



crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2020-02-16 15:43:12

性能

VSCode打开大文件速度也不错。

最近遇到30多M和700多MB的（csv）文本文件：

名称	修改日期	大小	种类
invalidSentences_20180628_5000个无效单词.csv	2018年6月28日 18:15	2.5 MB	CSV 文稿
invalidSentences_20180629_171152.csv	2018年6月29日 17:23	18.9 MB	CSV 文稿
invalidSentences_20180629_174251.csv	2018年6月29日 17:42	7 KB	CSV 文稿
invalidSentences_20180629_174309.csv	2018年6月29日 18:15	197.9 MB	CSV 文稿
invalidSentences_20180702_091757.csv	今天 12:10	762 MB	CSV 文稿
invalidWords_20180628_5000个无效单词.csv	2018年6月28日 18:15	45 KB	CSV 文稿
invalidWords_20180702_091757—去除无效的.csv	今天 13:36	39.6 MB	CSV 文稿
invalidWords_20180702_091757.csv	今天 13:36	39.6 MB	CSV 文稿
mongoDialogToMysqlQa.log	今天 13:36	1.57 GB	日志文件
mongoDialogToMysqlQa.py	2018年6月29日 17:42	16 KB	Python Source
README.html	今天 14:19	24 KB	HTML 文稿
README.md	今天 14:19	5 KB	Markdo...ument
vocabulary.txt	2018年5月31日 11:26	10 KB	纯文本文稿

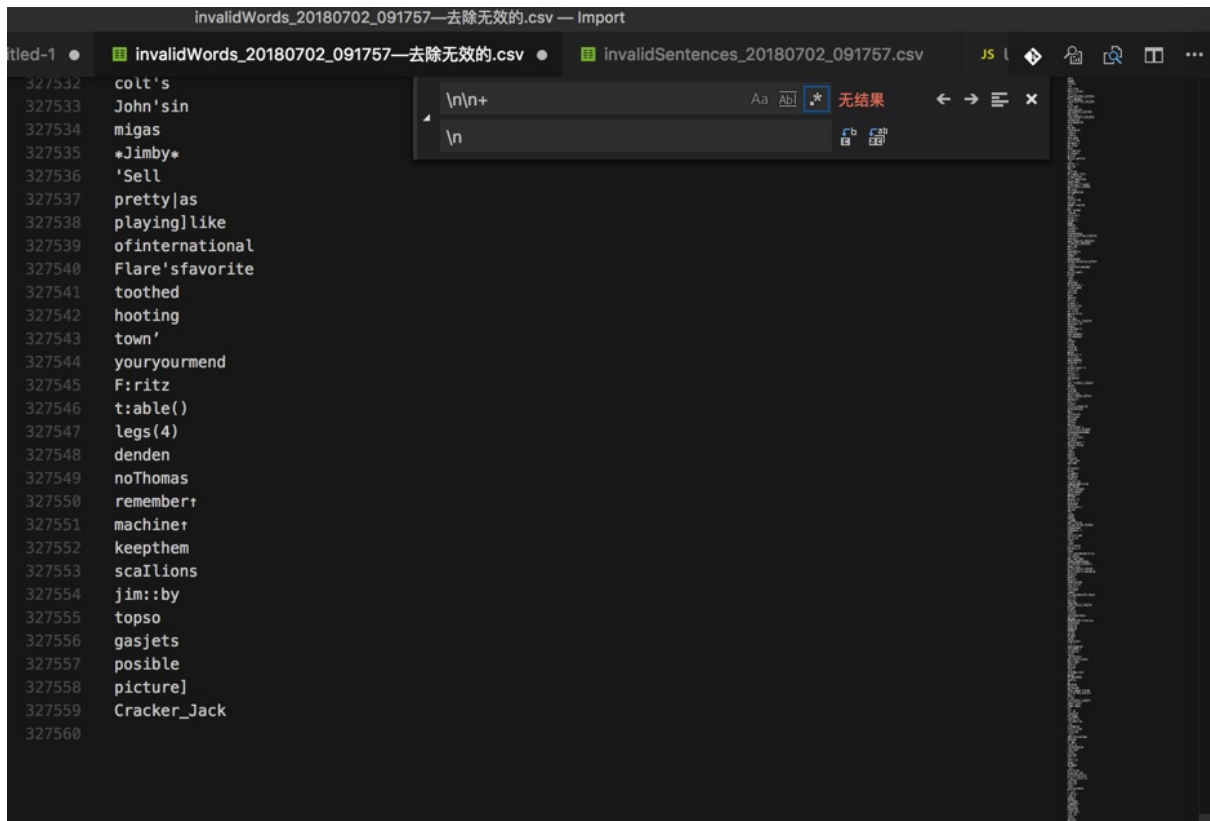
用VSCode打开，速度和反应也还是不错的：

（不过700多MB打开时，也是等了几十秒，然后卡死了一小会才打开）

```

invalidSentences_20180702_091757.csv — Import
title-1 • invalidWords_20180702_091757—去除无效的.csv • invalidSentences_20180702_091757.csv x JS |
8188977 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188978 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188979 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188980 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188981 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188982 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188983 5b0fa0e8d8d4e87251298e0e,T00_MANY_INVALID_WORD,I'm taking my favourite hat to wear in Indonesia,"tak
8188984 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,Present simple and present continuous,continuous
8188985 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Gran, what are you doing? GRAN","GRAN,doing,are"
8188986 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,I'm knitting? Why don't you say 'I knit'? That's the p
8188987 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,Because there are two present tenses in English and wh
8188988 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Well, I'm sitting on the sofa, I'm talking to you and
8188989 5b0fa0e8d8d4e87251298e0f,T00_SHORT,Good!,
8188990 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Well, for things we do every day, every week or every
8188991 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Well, for things we do every day, every week or every
8188992 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Well done! Oh, I remember when I was a girl ... Hey,
8188993 5b0fa0e8d8d4e87251298e0f,T00_MANY_INVALID_WORD,"Well done! Oh, I remember when I was a girl ... Hey,
8188994 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,Modals - 'must' and 'mustn't',"must",-,"mustn't"
8188995 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,"Oh Kitty, what's the matter?","what's
8188996 5b0fa0e8d8d4e87251298e0d,T00_SHORT,Nothing!,
8188997 5b0fa0e8d8d4e87251298e0d,T00_SHORT,Kitty!,
8188998 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,I got into trouble at school today because I took Hero
8188999 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,Because it's a rule. Can you think of any other school
8189000 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,Because it's a rule. Can you think of any other school
8189001 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,mustn't do?,mustn't
8189002 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,"Hmm. I mustn't eat in class, I mustn't shout or be ch
8189003 5b0fa0e8d8d4e87251298e0d,T00_MANY_INVALID_WORD,And what about things you must do?,things
8189004 5b0fa0e8d8d4e87251298e0d,T00_LONG,"We must wear our school uniform, we must do our homework and we mu
8189005

```



crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2019-03-16 21:16:46

附录

下面列出相关参考资料。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook 最后更新: 2019-03-02 22:22:58

参考资料

- [这就是我想要的 VSCode 插件! - 知乎](#)
- [\[VSCode插件推荐\] Bracket Pair Colorizer: 为代码中的括号添上一抹亮色 - 知乎](#)
- [【已解决】VSCode调试Python出错: 调试适配器进程意外终止Cannot read property 'style' of undefined](#)
- [30个极大提高开发效率的VSCode插件 - 知乎](#)
- [Debugger for Chrome - Visual Studio Marketplace](#)
- [【已解决】VSCode中对选中内容切换大小写的快捷键](#)
- [Visual Studio Code October 2018](#)
- [Backreferences and lookahead in search](#)
- [Multiline search](#)
- [Visual Studio Code 的正则匹配好用吗? - 知乎](#)
- [【已解决】后台导入csv文件后解析成功但导入失败](#)
- [【记录】VSCode试试其他主题和字体和插件](#)
- [【已解决】VSCode中搜索不到实际上是存在的字符串](#)
- [【已解决】VSCode中插件Paste Image的粘贴图片快捷键Command+Alt+V失效 - 在路上](#)
- [【已解决】VSCode中调试Python代码](#)
- [【记录】用VSCode开发和调试Python](#)
- [让你人生不苦短的编程语言: Python](#)
- [【已解决】windows中安装VSCode](#)
- [【记录】Windows中新建VSCode的Python项目](#)
- [【记录】Windows中用VSCode去调试Python项目代码文件](#)
-

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2020-02-16 17:17:14