

目录

前言	1.1
正则应用概述	1.2
正则应用举例	1.3
JavaScript	1.3.1
Python	1.3.2
BeautifulSoup	1.3.2.1
数据库	1.3.3
MongoDB	1.3.3.1
MongoDB Compass	1.3.3.2
编辑器和IDE	1.3.4
Sublime	1.3.4.1
VSCode	1.3.4.2
附录	1.4
参考资料	1.4.1

正则表达式应用举例

- 最新版本: v1.0
- 更新时间: 20200202

鸣谢

感谢我的老婆陈雪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。

简介

整理正则表达式各个领域的应用，并给出实际使用案例，供参考

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/regex_usage_examples: 正则表达式应用举例](#)

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

- [正则表达式应用举例 book.crifan.com](#)
- [正则表达式应用举例 crifan.github.io](#)

离线下载阅读

- [正则表达式应用举例 PDF](#)
- [正则表达式应用举例 ePUB](#)
- [正则表达式应用举例 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。

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: 2020-02-03 22:04:35

正则应用概述

如 [应用广泛的超强搜索：正则表达式](#) 所说，正则的用途非常广泛，下面来整理各个领域内正则的用途和具体例子。

目的在于：

- 没听过正则的人：了解正则在搜索和替换等方面的功能是多么强大
- 听过正则但不熟悉的人：能大概了解不同领域内，正则大概是怎么使用的
- 需要使用正则的人：借鉴正则的写法，在需要的时候，自己写正则，满足自己的需求

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：2020-02-02 19:42:55

正则应用举例

下面来介绍正则表达式在不同领域的各种应用。

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: 2020-02-02 19:45:17

JavaScript

代码：

```
let cowCodeRegex = new RegExp("/cowActivity/([^\/*]+)", "g");
console.log(`cowCodeRegex=${cowCodeRegex}`);
let foundCowCode = cowCodeRegex.exec(this.state.curUrl); //uapp/cowActivity/12-0985/1
/1522936800000
console.log(`foundCowCode=${foundCowCode}`); //cowActivity/13-6234,13-6234
let cowCode = null;
if (foundCowCode) {
  let inputStr = foundCowCode[0];
  console.log(`inputStr=${inputStr}`); //uapp/cowActivity/11-5953
  cowCode = foundCowCode[1];
  console.log(`cowCode=${cowCode}`); //11-5953
}
```

输出：

```
parsePageType: currentUrl /uapp/cowActivity/12-0985/1/1522936800000 - page 牛只活动量
index.js?5c2a:528 cowCodeRegex /\cowActivity\/([^\/*]+)/g
index.js?5c2a:530 foundCowCode /cowActivity/12-0985,12-0985
index.js?5c2a:534 inputStr /cowActivity/12-0985
index.js?5c2a:536 cowCode 12-0985
index.js?5c2a:540 fullTitle 牛只活动量 12-0985
```

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：2020-02-02 19:44:16

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: 2020-02-02 19:33:40

BeautifulSoup

BeautifulSoup 中的 find 和 findAll 的 name 或 attr 参数，支持正则写法：

代码：

```
h1userSoupList = soup.findAll(name="h1", attrs {"class":re.compile(r" h1user(\s\w+)?")});
```

可以从html：

```
<div class="icon_col">
    <h1 class="h1user">crifan</h1>
    <h1 class="h1user test1">crifan 123</h1>
    <h1 class="h1user test2">crifan 456</h1>
</div>
```

搜到列表：

```
class "h1user"
class "h1user test1"
class "h1user test2"
```

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：2020-02-02 19:51:06

数据库

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: 2020-02-02 19:34:48

MongoDB

相关代码：

```
if args.get('title'):
    filters['title'] = {'$regex': args['title'], '$options': 'i'}

if args.get('unitCode'):
    filters['unit_code'] = {'$regex': args['unitCode'], '$options': 'i'}
```

其中的 \$regex 表示搜索时，用正则搜索

-» 此处含义是：去搜索 title 和 unitCode 两个字段，且 i 表示不区分大小写

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：2020-02-02 19:47:17

MongoDB Compass

MongoDB Compass 中想要用正则搜索字段：

```
grading
  lexile: "AD450L"
```

写法是：

```
{"grading.lexile": {$regex: "AD.*"})
```

或： regex加上行首和行尾判断：

```
{"grading.lexile": {$regex: "^AD.*$")}
```

或 regex用引号引起来

```
{"grading.lexile": {"$regex": "AD.*"})
```

详见：

【整理Book】主流文档型数据库：MongoDB

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：2020-02-02 19:49:39

编辑器和IDE

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: 2020-02-02 19:35:53

Sublime

提取youtube网页返回的html中包含的子页面的url地址

比如，用：

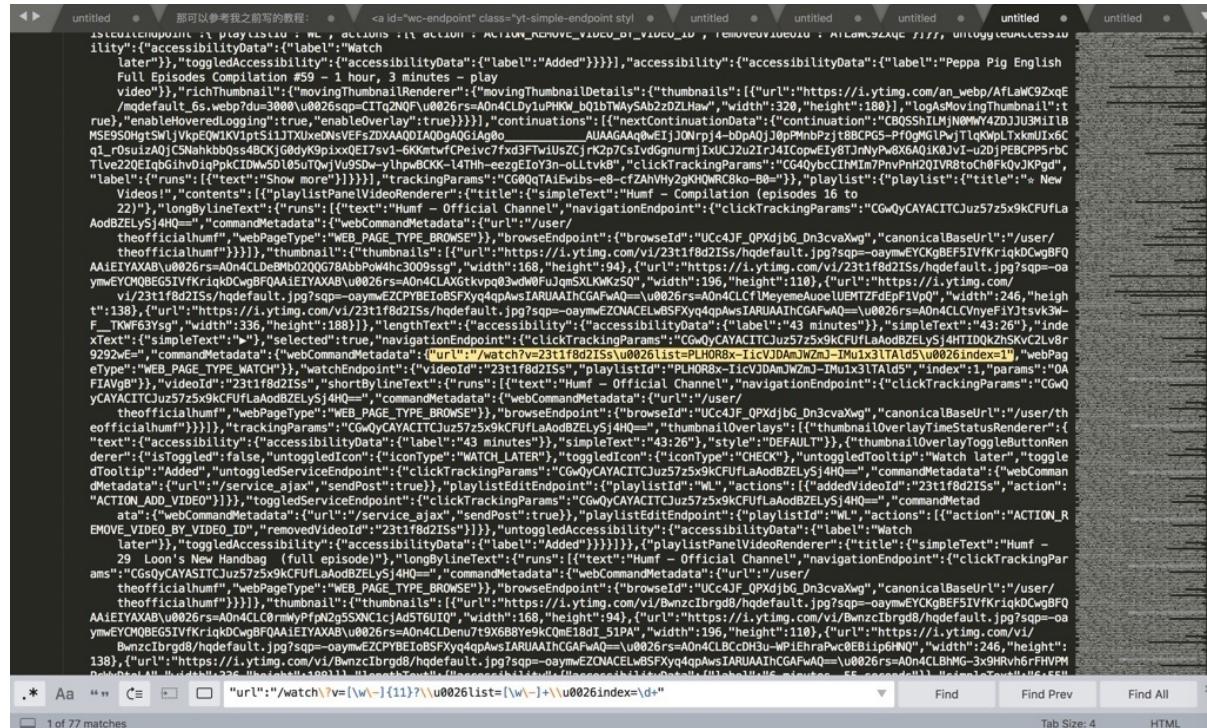
"url": "/watch\\?v=[\\w\\-]{11}\\u0026list=[\\w\\-]+\\u0026index=\\d+"

可以从一堆的js中的script的值中：

```
{"webCommandMetadata": {"url": "/watch?v=23t1f8d2ISs\u0026list=PLHOR8x-IicVJDAmJWZmJ-IMu1x31TAld5\u0026index=1", "webPageType": "
```

```
{"webCommandMetadata": {"url": "/watch?v=au7Nkr-5MA8\u0026list=PLHOR8x-IicVJDAmJWZmJ-IMu1x31TAld5\u0026index=14"}, "webPageType": "
```

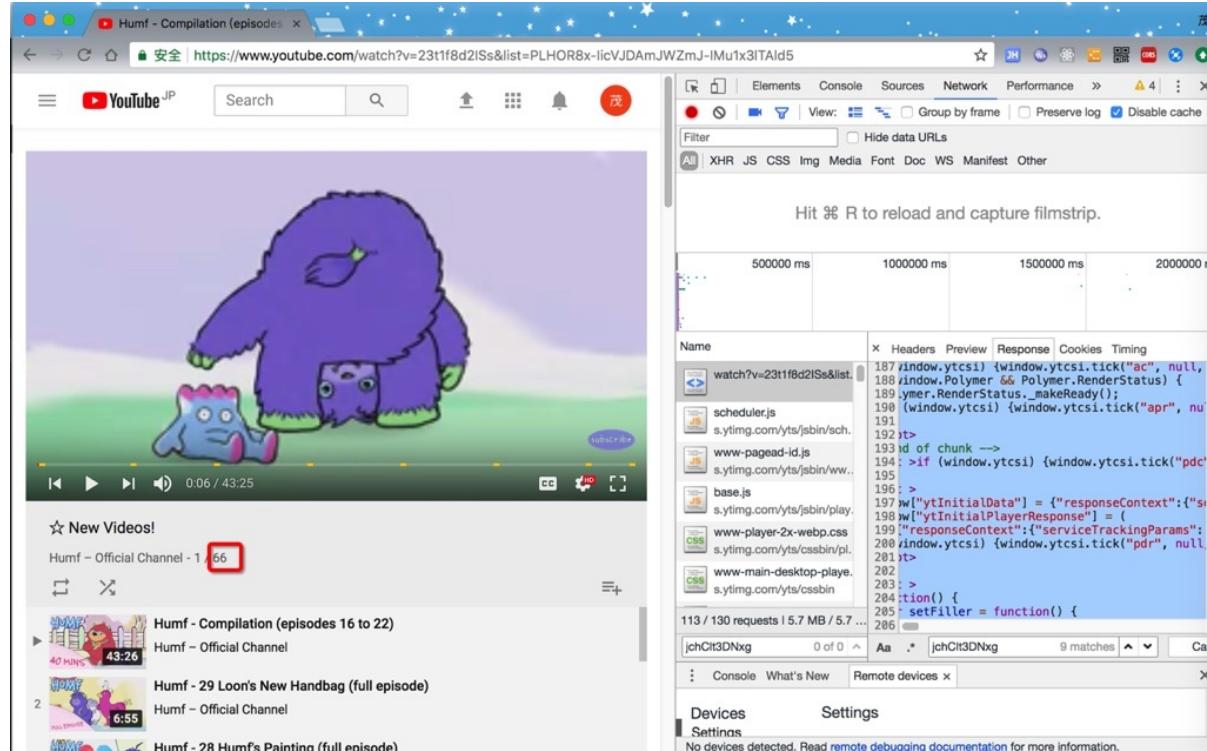
搜索出所要的内容，此处有77个符合需要的内容： 1 / 77



点击Find，继续向下找，比如找到第14个： 14/77

另外，进一步的举例：

此处，对应着页面上的其实只是希望找到66个地址就可以了：



但是此处找到77个，多出11个，则是由于：

此外的is的变量的值中，包含了不需要的额外的11个

所以此时，由于没法方便的从字符串中区别开来，不好去掉另外那11个，则只能：

想办法拿到js的变量值，然后通过转换为json，然后再去获取json对象中的值，即可准确的得到所需要的值。

所以此处，又可以接着通过正则去先得到js的变量的值：

用正则：

```
window\["ytInitialData"\]\s* \s*
```



```
; \s+window\["ytInitialPlayerResponse"\]
```

从：

```
<script>
  window["ytInitialData"] = {"responseContext": "xqmOCnbELAge VPNj1N1SqHurYg"};
  window["ytInitialPlayerResponse"] = (
```

中，搜索到所要的内容：

而首尾的正则之间的内容，就是需要找的js的变量的值，是个json

对应着写个完整的正则：

```
window\[\"ytInitialData\"\]\s*\s*(.+?);s+window\[\"ytInitialPlayerResponse\"\]
```

就可以匹配到这段完整的内容了：

```
187 <script>
188   if (window.ytcsi) {window.ytcsi.tick("ac", null, '')};
189   if (window.Polymer && Polymer.RenderStatus) {
190     Polymer.RenderStatus._makeReady();
191     if (window.ytcsi) {window.ytcsi.tick("apr", null, '')};
192   }
193 </script>
194 <!-- end of chunk -->
195 <script> if (window.ytcsi) {window.ytcsi.tick("pdc", null, '')};</script>
196
197 <script>
198 window["ytInitialData"] = {"responseContext": {"serviceTrackingParams": [{"service": "GFEEDBACK", "params": [{"key": "has_unlimited_entitlement", "value": "False"}, {"key": "has_unlimited_ncc_free_trial", "value": "False"}, {"key": "e", "value": "23788904,23788906,23708910,23710476,23712994,23713711,23715480,23716420,23716663,23716986,23717456,23719525,23721750,23721898,23722182,23723618,23724477,23724688,23725111,23726426,23726508,23727146,23727420,23728274,3300116,3300132,3310161,331321,3314088,9405989,942596,9413754,9441384,9452652,9471239,9474357,9475643,9485008,9486759}, {"key": "logged_in", "value": "True"}, {"key": "watch", "value": "True"}], "service": "GUIDER_HELP", "params": [{"key": "context", "value": "yt_web_kevlar_watch"}, {"key": "creator_channel_id", "value": "UCJcmJNk6Bx0BjHSY1ldcgv"}, {"key": "logged_in", "value": "True"}, {"key": "service", "value": "CSN"}, {"key": "params": [{"key": "getWatchNext_rid", "value": "0xf5f3b782b1d410"}, {"key": "c", "value": "WEB"}, {"key": "cver", "value": "2.20180227"}, {"key": "yt_li", "value": "1"}, {"key": "service", "value": "ECATCHER"}, {"key": "params": [{"key": "client.name", "value": "WEB"}, {"key": "client.version", "value": "2.20180227"}, {"key": "inertube.build.changeList", "value": "187164786"}, {"key": "inertube.build.experiments.sourceVersion", "value": "187191768"}, {"key": "inertube.build.timestamp", "value": "1519740832"}, {"key": "inertube.build.variants.checksum", "value": "9360fd205a26124e140a1e3be1ca"}, {"key": "inertube.run.job", "value": "ytfe-jser-replica-only,ytfe"}]}, {"key": "webResponseContextExtensionData", "value": "ytConfigData": {"Csn": "8eMwtuA08ewh0oFq7IDA", "visitorData": "CgthnChbdy2ndJR0q3Dw3D", "sessionIndex": 0, "rootVisualElementType": 3832}, "feedbackDialog": {"polymerOptOutFeedbackDialogRenderer": {"title": "runs": [{"text": "We're sorry to see you go!"}], "subtitle": "runs": [{"text": "Please tell us why. Your feedback helps us improve YouTube. Remember, you can always return to the new design by going to \"[text\": \"youtube.com/new\", \"navigationEndpoint\": {\"commandMetadata\": {\"url\": \"https://www.youtube.com/new\"}}, \"urlEndpoint\": {\"url\": \"https://www.youtube.com/new\"}}]"}, {"options": [{"polymerOptOutFeedbackOptionRenderer": {"optionKey": "missing", "description": "runs": [{"text": "Something is missing!"}]}, "responsePlaceholder": {"runs": [{"text": "Tell us more!"]}], "polymerOptOutFeedbackOptionRenderer": {"optionKey": "broken", "description": "runs": [{"text": "Something is broken!"}]}, "responsePlaceholder": {"runs": [{"text": "Tell us more!"]}], "polymerOptOutFeedbackOptionRenderer": {"optionKey": "harder", "description": "runs": [{"text": "Something is harder to use!"}]}, "responsePlaceholder": {"runs": [{"text": "Tell us more!"]}], "polymerOptOutFeedbackOptionRenderer": {"optionKey": "dislike", "description": "runs": [{"text": "I don't like the new design!"}]}, "responsePlaceholder": {"runs": [{"text": "Tell us more!"]}], "polymerOptOutFeedbackOptionRenderer": {"optionKey": "unlisted", "description": "runs": [{"text": "My reason isn't listed!"}]}]}]
```

后续就可以通过解析json去精确获取所要的url的值了。

比如第12个url：

```
11262
11263
11264
11265
11266
11267
11268
11269
11270
11271
11272
11273
11274
11275
11276
11277
11278
11279
11280
11281
11282
11283
11284
11285
    "accessibility": {
        "accessibilityData": {
            "label": "6 minutes, 56 seconds"
        }
    },
    "simpleText": "6:56"
},
"indexText": {
    "simpleText": "12"
},
"selected": false,
"navigationEndpoint": {
    "clickTrackingParams": "CGEQuCAYCyITCJu57z5x9kCFUfLaAoBZELySj4HTIDQkZhSKvC2Llv8r9292wE",
    "commandMetadata": {
        "webCommandMetadata": {
            "url": "/watch?v=jchClt3DNg&list=PLHOR8x-IicVJDAmJWZmJ-IMu1x3lTAld5&index=12",
            "webPageType": "WEB_PAGE_TYPE_WATCH"
        }
    }
},
"watchEndpoint": {
    "videoId": "jchClt3DNg",
    "playlistId": "PLHOR8x-IicVJDAmJWZmJ-IMu1x3lTAld5",
    "index": 12,
    "params": "OAFIAVwM"
}
.* Aa “ “ C≡ ⌂ /watch?v=jchClt3DNg&list=PLHOR8x-IicVJDAmJWZmJ-IMu1x3lTAld5&index=12 Find Find Prev Find All
```

由此实现了：

根据自己的实际的（业务）需求，通过充分利用正则表达式，获取想要的符合特定某一规则的内容。

html中提取出浙江省的每个市到Xmind中

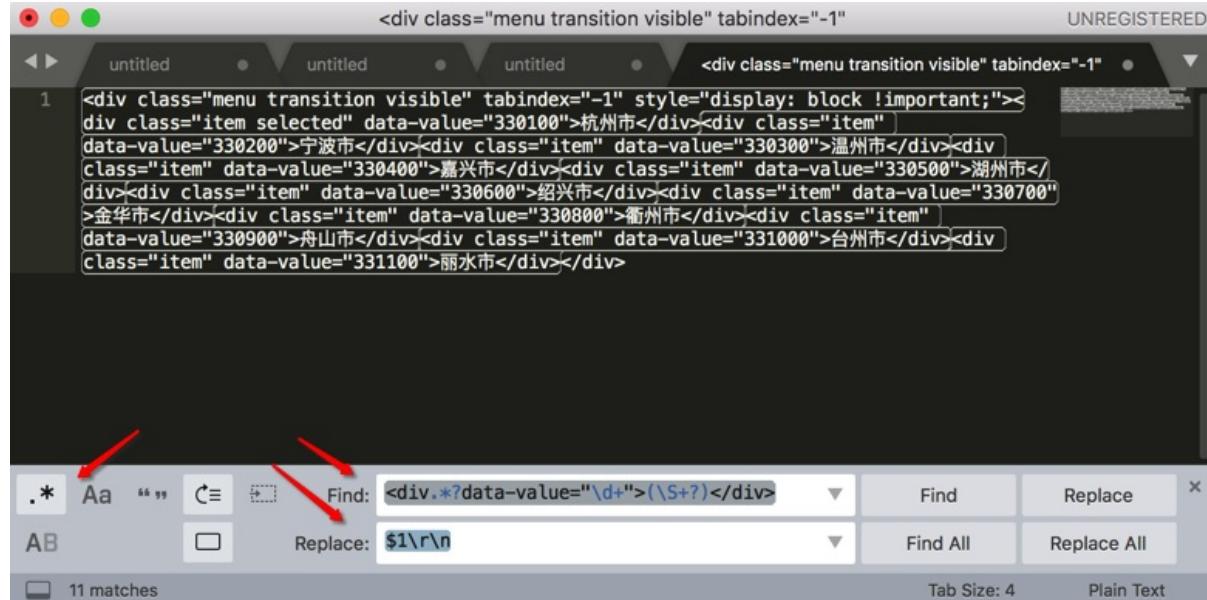
```
<div class="menu transition visible" tabindex="-1" style="display: block !important;"><div class="item selected" data-value="330100">杭州市</div><div class="item" data-value="330200">宁波市</div><div class="item" data-value="330300">温州市</div><div class="item" data-value="330400">嘉兴市</div><div class="item" data-value="330500">湖州市</div><div class="item" data-value="330600">绍兴市</div><div class="item" data-value="330700">金华市</div><div class="item" data-value="330800">衢州市</div><div class="item" data-value="330900">舟山市</div><div class="item" data-value="331000">台州市</div><div class="item" data-value="331100">丽水市</div></div>
```

希望提取出每个市

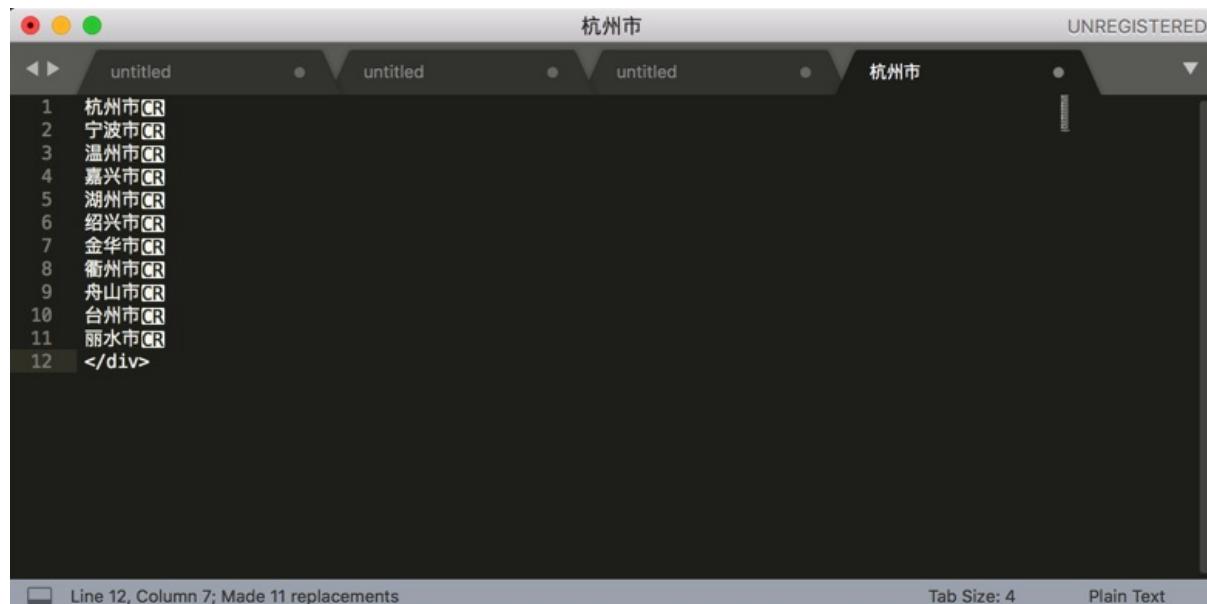
正则写法：

- 查找Find: <div.*?data-value="\d+>(\S+?)</div>
- 替换Replace: \$1\r\n

点击 Replace All :



替换成：



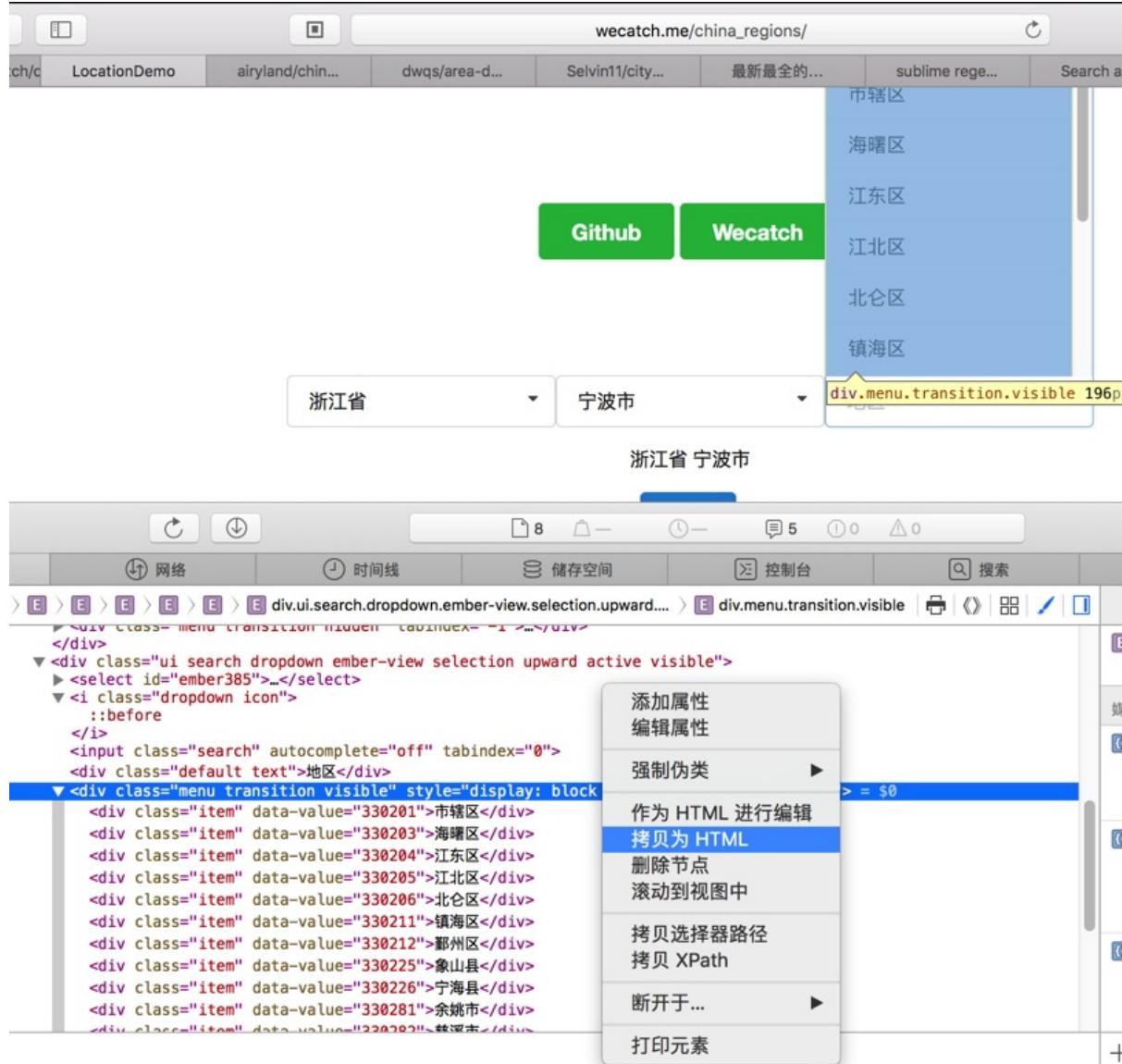
忽略掉最后的 </div>，拷贝出来，即可得到我要的所有的市：

```
杭州市
宁波市
温州市
嘉兴市
湖州市
```

绍兴市
金华市
衢州市
舟山市
台州市
丽水市

->

如此继续重复此步骤，直到把网页中的内容：



分多次，但是是批量的：

The screenshot shows two tabs open in Sublime Text. The top tab, titled 'untitled', contains a block of XML-like code with numerous nested `<div>` elements. The bottom tab, also titled 'untitled' and labeled '市辖区', contains a list of 12 items, each preceded by a number from 1 to 13. The items are separated by carriage returns (CR). A search/replace interface is visible at the bottom of the screen, with the 'Find' field containing the regex pattern `<div.*?data-value="\d+>(\S+?)</div>` and the 'Replace' field containing `$1\r\n`. The status bar at the bottom indicates 'Line 13, Column 7; Made 12 replacements'.

untitled

UNREGISTERED

```
<div class="menu transition visible" tabindex="-1">
<div class="item" data-value="330201">市辖区</div><div class="item" data-value="330203">海曙区</div><div class="item" data-value="330204">江东区</div><div class="item" data-value="330205">江北区</div><div class="item" data-value="330206">北仑区</div><div class="item" data-value="330211">镇海区</div><div class="item" data-value="330212">鄞州区</div><div class="item" data-value="330225">象山县</div><div class="item" data-value="330226">宁海县</div><div class="item" data-value="330281">余姚市</div><div class="item" data-value="330282">慈溪市</div><div class="item" data-value="330283">奉化市</div></div>
```

AB

Find: <div.*?data-value="\d+>(\S+?)</div>

Replace: \$1\r\n

12 matches

untitled

untitled

untitled

市辖区

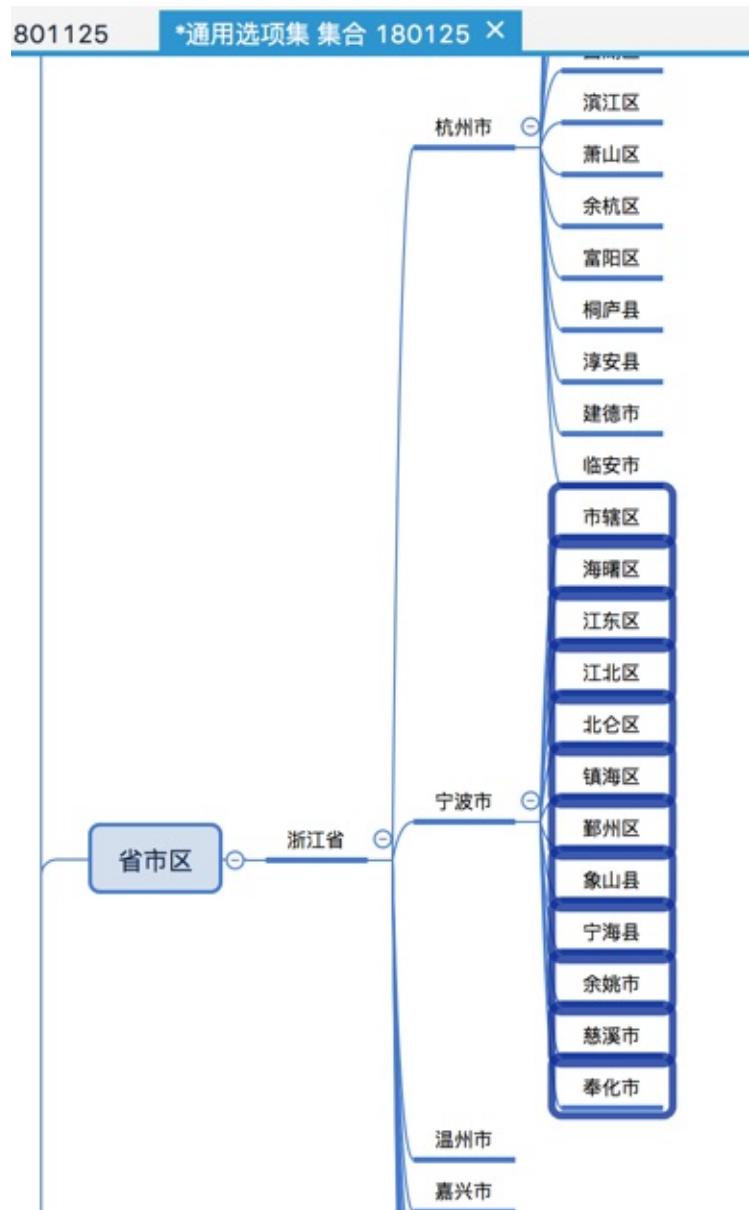
UNREGISTERED

```
1 市辖区CR
2 海曙区CR
3 江东区CR
4 江北区CR
5 北仑区CR
6 镇海区CR
7 鄞州区CR
8 象山县CR
9 宁海县CR
10 余姚市CR
11 慈溪市CR
12 奉化市CR
13 </div>
```

Line 13, Column 7; Made 12 replacements

Tab Size: 4 Plain Text

全部都整理到Xmind中：



就不用一个个拷贝，一个个粘贴了 -》 从而提高工作效率。

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: 2020-02-02 22:50:19

VSCode

普通的搜索和替换

下面是一些相对普通的正则的搜索和替换的应用举例：

去除内容中多余的 lessonxxx 的单词

正则：

```
lesson\s*\d+\n
```

从：

The screenshot shows the VSCode interface with a dark theme. On the left, there are two tabs: 'Const.java' and 'Untitled-1'. The 'Untitled-1' tab is active and displays a list of words, many of which are preceded by line numbers. In the center, a search and replace dialog box is open. The 'Search' field contains the regular expression 'lesson\s*\d+\n'. The 'Replace' field is empty. To the right of the search field, there are several icons: a font size dropdown, a case sensitivity switch ('Aa'), a search mode switch ('Ab'), a 'Match Case' checkbox (which is checked), a 'Match Whole Word' checkbox (which is checked), and a 'Match Selection' checkbox. Next to these are navigation icons for previous/next match ('← →'), a 'Replace All' button ('替换'), and a close button ('X'). At the bottom of the dialog is a button labeled '全部替换 (Esc/Enter)'.

替换成：

Const.java Untitled-1 ●

lesson\s*\d+\n Aa Abi * 无结果 替换

行 852, 列 1 空格: 2 UTF-8 LF 纯文

```
842 always
843 cd
844 dvd
845 jacket
846 magazine
847 video
848 blouse
849 skirt
850 vegetable
851 relative
852 kind
853 let me think
854 japanese
855 remember
856 i know!
857 lucky
858 yoghurt
859 another
860 like
861 birthday
862 at any time
863 british
864 german
865 plastic
866 leather
867 cd player
868 italian
869 watch
870 korean
871 handbag
872 clock
873 swiss
874 very much
875 love
876 salmon
877 piece
878 tonight
879 potato
880 lettuce
```

语句末尾去掉感叹号

正则:

```
\n\n
```

从:

替换成：

高级的搜索和替换

下面是一些，相对高级一些的用法，比如：

- 搜索带 分组
 - 即： (xxx)
- 替换时带 引用
 - 即： \$N
 - N=1,2,3,...

等正则应用举例，供参考。

文章标题和链接转换为Markdown的链接

正则替换规则：

```
(.+)\n(http.+)\n
* [$1]($2)\n
```

从：

```
android - decompiling DEX into Java sourcecode - Stack Overflow
https://stackoverflow.com/questions/1249973/decompiling-dex-into-java-sourcecode/55486175#55486175
decompiler - how to use DEXtoJar - Stack Overflow
https://stackoverflow.com/questions/5257830/how-to-use-dextojar/55486507#55486507
android - Is there a way to get the source code from an APK file? - Stack Overflow
```

<https://stackoverflow.com/questions/3593420/is-there-a-way-to-get-the-source-code-from-an-apk-file/55567538#55567538>
Android反编译简单实战 - 知乎
<https://zhuanlan.zhihu.com/p/51260384>
Android应用加固产品使用对比 - 『移动安全区』 - 番薯破解 - LCG - LSG | 安卓破解 | 病毒分析 | 破解软件
www.52pojie.cn
<https://www.52pojie.cn/thread-832804-1-1.html>
Android混淆（ProGuard）从0到1 - 简书
<https://www.jianshu.com/p/1b76e4c10495>
乐固加固脱壳实战 - faTe's Home
<http://www.holdheart.com/archives/33.html>
乐固壳分析 - bamb00 - 博客园
<http://www.cnblogs.com/goodhacker/p/8666217.html>
Android APK 反编译实践 - 简书
<https://www.jianshu.com/p/9e0d1c3e342e>
5分钟学会基于Xposed+DumpDex的apk快速脱壳方法 - 简书
<https://www.jianshu.com/p/9d988bdddb3d>
腾讯加固纯手工简易脱壳教程 - 『移动安全区』 - 番薯破解 - LCG - LSG | 安卓破解 | 病毒分析 | 破解软件 | www.52pojie.cn
<https://www.52pojie.cn/thread-428271-1-1.html>
ANDROID 逆向实例（八） - 乐固加固脱壳（2017.01） ~ and-rev
<https://and-rev.blogspot.com/2017/05/android-201701.html>
花生日记APP邀请注册机实战（360加固脱壳） - Silkage's Blog
<https://blog.silkage.net/software/peanutdiary.html>
如何反编译Android 的apk/dex/odex, 获得源码 - 码农日记
<https://www.androiddev.net/反编译android-的apk/>
HangZhouCat/ReaverAPKTools: 逆向APK工具
<https://github.com/HangZhouCat/ReaverAPKTools>
Android逆向之路---脱壳360加固 - 简书
<https://www.jianshu.com/p/d24c6694fe97>
26款优秀的Android逆向工程工具 - 简书
<https://www.jianshu.com/p/ef0b6f75c229>
Application Hardening - Mobile App Hardening | Promon
<https://promon.co/security-news/application-hardening/>
Cydia Substrate使用手册 - 简书
<https://www.jianshu.com/p/ba795ff3471a>

把：

```

1 android - decompiling DEX into Java sourcecode - Stack Overflow
2 https://stackoverflow.com/questions/1249973/decompiling-dex-into-java-sourcecode/55486175#55486175
3 decompiler - how to use DEXtoJar - Stack Overflow
4 https://stackoverflow.com/questions/5257830/how-to-use-dextojar/55486507#55486507
5 android - Is there a way to get the source code from an APK file? - Stack Overflow
6 https://stackoverflow.com/questions/3593420/is-there-a-way-to-get-the-source-code-from-an-apk-file/55567538#55567538
7 Android反编译简单实战 - 知乎
8 https://zhuanlan.zhihu.com/p/51260384
9 Android应用加固产品使用对比 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn
10 https://www.52pojie.cn/thread-832804-1-1.html
11 Android混淆 (ProGuard) 从0到1 - 简书
12 https://www.jianshu.com/p/1b76e4c10495
13 乐固加固脱壳实战 - faTe's Home
14 http://www.holdheart.com/archives/33.html
15 乐固壳分析 - bamb00 - 博客园
16 http://www.cnblogs.com/goodhacker/p/8666217.html
17 Android APK 反编译实践 - 简书
18 https://www.jianshu.com/p/9e0d1c3e342e
19 5分钟学会基于Xposed+DumpDex的apk快速脱壳方法 - 简书
20 https://www.jianshu.com/p/9d988bdddb3d
21 腾讯加固纯手工简易脱壳教程 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn
22 https://www.52pojie.cn/thread-428271-1-1.html
23 ANDROID 逆向实例 (八) - 乐固加固脱壳 (2017.01) ~ and-rev

```

换成：

```

* [android - decompiling DEX into Java sourcecode - Stack Overflow](https://stackoverflow.com/questions/1249973/decompiling-dex-into-java-sourcecode/55486175#55486175)
* [decompiler - how to use DEXtoJar - Stack Overflow](https://stackoverflow.com/questions/5257830/how-to-use-dextojar/55486507#55486507)
* [android - Is there a way to get the source code from an APK file? - Stack Overflow](https://stackoverflow.com/questions/3593420/is-there-a-way-to-get-the-source-code-from-an-apk-file/55567538#55567538)
* [Android反编译简单实战 - 知乎](https://zhuanlan.zhihu.com/p/51260384)
* [Android应用加固产品使用对比 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn](https://www.52pojie.cn/thread-832804-1-1.html)
* [Android混淆 (ProGuard) 从0到1 - 简书](https://www.jianshu.com/p/1b76e4c10495)
* [乐固加固脱壳实战 - faTe's Home](http://www.holdheart.com/archives/33.html)
* [乐固壳分析 - bamb00 - 博客园](http://www.cnblogs.com/goodhacker/p/8666217.html)
* [Android APK 反编译实践 - 简书](https://www.jianshu.com/p/9e0d1c3e342e)
* [5分钟学会基于Xposed+DumpDex的apk快速脱壳方法 - 简书](https://www.jianshu.com/p/9d988bdddb3d)
* [腾讯加固纯手工简易脱壳教程 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn](https://www.52pojie.cn/thread-428271-1-1.html)
* [ANDROID 逆向实例 (八) - 乐固加固脱壳 (2017.01) ~ and-rev](https://and-rev.blogspot.com/2017/05/android-201701.html)
* [花生日记APP邀请注册机实战 (360加固脱壳) - Silkage's Blog](https://blog.silkage.net/software/peanutdiary.html)
* [如何反编译Android 的apk/dex/odex, 获得源码 - 码农日记](https://www.androiddev.net/反编译android-的apk/)
* [HangZhouCat/ReaverAPKTools: 逆向APK工具](https://github.com/HangZhouCat/ReaverAPKTools)
* [Android逆向之路---脱壳360加固 - 简书](https://www.jianshu.com/p/d24c6694fe97)
* [26款优秀的Android逆向工程工具 - 简书](https://www.jianshu.com/p/ef0b6f75c229)
* [Application Hardening - Mobile App Hardening | Promon](https://promon.co/security-news/application-hardening/)
* [Cydia Substrate使用手册 - 简书](https://www.jianshu.com/p/ba795ff3471a)

```

```

1  * [android - decompiling DEX into Java sourcecode - Stack Overflow](https://stackoverflow.com/questions/5257830/android-decompiling-dex-into-java-sourcecode)
2  * [decompiler - how to use DEXtoJar - Stack Overflow](https://stackoverflow.com/questions/5257830/decompiler-how-to-use-dextojar)
3  * [android - Is there a way to get the source code from an APK file? - Stack Overflow](https://stackoverflow.com/questions/5257830/android-is-there-a-way-to-get-the-source-code-from-an-apk-file)
4  * [Android反编译简单实战 - 知乎](https://zhuanlan.zhihu.com/p/51260384)
5  * [Android应用加固产品使用对比 - 『移动安全区』 - 吾爱破解 - LCG - LSG | 安卓破解|病毒分析|破解软件|www.52pojie.cn](https://www.jianshu.com/p/1b76e4c10495)
6  * [Android混淆 (ProGuard) 从0到1 - 简书](https://www.jianshu.com/p/1b76e4c10495)
7  * [乐固加固脱壳实战 - faTe's Home](http://www.holdheart.com/archives/33.html)
8  * [乐固壳分析 - bamb00 - 博客园](http://www.cnblogs.com/goodhacker/p/8666217.html)
9  * [Android APK 反编译实践 - 简书](https://www.jianshu.com/p/9e0d1c3e342e)
10 * [5分钟学会基于Xposed+DumpDex的apk快速脱壳方法 - 简书](https://www.jianshu.com/p/9d988bdddb3d)
11 * [腾讯加固纯手工简易脱壳教程 - 『移动安全区』 - 吾爱破解 - LCG - LSG | 安卓破解|病毒分析|破解软件|www.52pojie.cn](https://www.jianshu.com/p/9d988bdddb3d)
12 * [ANDROID 逆向实例（八） - 乐固加固脱壳（2017.01）~ and-rev](https://and-rev.blogspot.com/2017/05/android-reverse-engineering-example-eight.html)
13 * [花生日记APP邀请注册机实战（360加固脱壳） - Silkage's Blog](https://blog.silkage.net/software/peanutdiary-app-invite-registrator)
14 * [如何反编译Android 的apk/dex/odex, 获得源码 - 码农日记](https://www.androiddev.net/reverse-engineer-android-apk/)
15 * [HangZhouCat/ReaverAPKTools: 逆向APK工具](https://github.com/HangZhouCat/ReaverAPKTools)
16 * [Android逆向之路---脱壳360加固 - 简书](https://www.jianshu.com/p/d24c6694fe97)
17 * [26款优秀的Android逆向工程工具 - 简书](https://www.jianshu.com/p/ef0b6f75c229)
18 * [Application Hardening - Mobile App Hardening | Promon](https://promon.co/security-news/application-hardening)
19 * [Cydia Substrate使用手册 - 简书](https://www.jianshu.com/p/ba795ff3471a)
20
21

```

用于：放在 markdown 作为参考资料。

json后缀的字符串变成代码中字符串列表

正则：

```
(\w+)\n
"$1",\n
```

从：

```

booklistsJson
rbrsJson
latestCommentJson
myCommentJson
whoHasThisBookJson
topicArrayJson
bookFeaturesArrayJson
bookFeaturesWithContentArrayJson
worksCollectionArrayJson
readingAgeDistributionArrayJson
topReadingAgeDistributionArrayJson
scoreDistributionArrayJson
likeThisBookKidsAlsoLikeBookArrayJson
childdataArrayJson
inPagePictureArrayJson
xunxiArrayJson
experienceArrayJson
answerArrayJson
englishLevelArrayJson

```

```
1 booklistsJson
2 rbrsJson
3 latestCommentJson
4 myCommentJson
5 whoHasThisBookJson
6 topicArrayJson
7 bookFeaturesArrayJson
8 bookFeaturesWithContentArrayJson
9 worksCollectionArrayJson
10 readingAgeDistributionArrayJson
11 topReadingAgeDistributionArrayJson
12 scoreDistributionArrayJson
13 likeThisBookKidsAlsoLikeBookArrayJson
14 childdataArrayJson
15 inPagePictureArrayJson
16 xunxiArrayJson
17 experienceArrayJson
18 answerArrayJson
19 englishLevelArrayJson
20
```

变成：

```
"booklistsJson",
"rbrsJson",
"latestCommentJson",
"myCommentJson",
"whoHasThisBookJson",
"topicArrayJson",
"bookFeaturesArrayJson",
"bookFeaturesWithContentArrayJson",
"worksCollectionArrayJson",
"readingAgeDistributionArrayJson",
"topReadingAgeDistributionArrayJson",
"scoreDistributionArrayJson",
"likeThisBookKidsAlsoLikeBookArrayJson",
"childdataArrayJson",
"inPagePictureArrayJson",
"xunxiArrayJson",
"experienceArrayJson",
"answerArrayJson",
"englishLevelArrayJson",
```

```

1 "booklistsJson",
2 "rbrsJson",
3 "latestCommentJson",
4 "myCommentJson",
5 "whoHasThisBookJson",
6 "topicArrayJson",
7 "bookFeaturesArrayJson",
8 "bookFeaturesWithContentArrayJson",
9 "worksCollectionArrayJson",
10 "readingAgeDistributionArrayJson",
11 "topReadingAgeDistributionArrayJson",
12 "scoreDistributionArrayJson",
13 "likeThisBookKidsAlsoLikeBookArrayJson",
14 "childdataArrayJson",
15 "inPagePictureArrayJson",
16 "xunxiArrayJson",
17 "experienceArrayJson",
18 "answerArrayJson",
19 "englishLevelArrayJson",
20

```

用于：

拷贝到代码里，用于列表变量的值：

```

509     fieldNameList = [
510         "booklistsJson",
511         "rbrsJson",
512         "latestCommentJson",
513         "myCommentJson",
514         "whoHasThisBookJson",
515         "topicArrayJson",
516         "bookFeaturesArrayJson",
517         "bookFeaturesWithContentArrayJson",
518         "worksCollectionArrayJson",
519         "readingAgeDistributionArrayJson",
520         "topReadingAgeDistributionArrayJson",
521         "scoreDistributionArrayJson",
522         "likeThisBookKidsAlsoLikeBookArrayJson",
523         "childdataArrayJson",
524         "inPagePictureArrayJson",
525         "xunxiArrayJson",
526         "experienceArrayJson",
527         "answerArrayJson",
528         "englishLevelArrayJson",
529     ]
530     bookInfoDict = self.dictJsonStrToDict(bookInfoDict, fieldNameList)
531
532     saveJsonToFile(singleBookFullPath, bookInfoDict)

```

省去：自己手动去对每一行手动去加上 "" 再控制 缩进 的繁琐工作了。

获取到康美通的版本历史

正则：

`(\\.\\d+)\\n`

\$1

从：

相关历史版本**23.34MB最新版****康美通 4.3.0****23.34MB 安全下载****康美通 4.2.3****19.47MB 安全下载****康美通 4.2.2****19.29MB 安全下载****康美通 4.2.1****19.29MB 安全下载****康美通 4.2.0****19.18MB 安全下载****康美通 4.1.1****19.47MB 安全下载****康美通 4.1.0****19.48MB 安全下载****康美通 4.0****14.37MB 安全下载****康美通 3.2****14.22MB 安全下载****康美通 3.1****23.2MB 安全下载****康美通 3.0****23.1MB 安全下载****康美通 2.0.9****7.93MB 安全下载****康美通 2.0.8****7.3MB 安全下载****康美通 2.0.7****7.29MB 安全下载****康美通 2.0.6****7.29MB 安全下载****康美通 1.0.1****6.27MB 安全下载****康美通 1.0beta****5.86MB 安全下载****下载豌豆荚客户端 (更多历史版本) 下载****康美通 历史版本年份合集**

```

1 相关历史版本
2
3 23.34MB最新版
4 康美通 4.3.0
5 23.34MB 安全下载
6 康美通 4.2.3
7 19.47MB 安全下载
8 康美通 4.2.2
9 19.29MB 安全下载
10 康美通 4.2.1
11 19.29MB 安全下载
12 康美通 4.2.0
13 19.18MB 安全下载
14 康美通 4.1.1
15 19.47MB 安全下载
16 康美通 4.1.0
17 19.48MB 安全下载
18 康美通 4.0
19 14.37MB 安全下载
20 康美通 3.2
21 14.22MB 安全下载
22 康美通 3.1
23 23.2MB 安全下载
24 康美通 3.0
25 23.1MB 安全下载
26 康美通 2.0.9
27 7.93MB 安全下载
28 康美通 2.0.8

```

替换成：

相关历史版本

23.34MB最新版

康美通 4.3.0 23.34MB 安全下载
 康美通 4.2.3 19.47MB 安全下载
 康美通 4.2.2 19.29MB 安全下载
 康美通 4.2.1 19.29MB 安全下载
 康美通 4.2.0 19.18MB 安全下载
 康美通 4.1.1 19.47MB 安全下载
 康美通 4.1.0 19.48MB 安全下载
 康美通 4.0 14.37MB 安全下载
 康美通 3.2 14.22MB 安全下载
 康美通 3.1 23.2MB 安全下载
 康美通 3.0 23.1MB 安全下载
 康美通 2.0.9 7.93MB 安全下载
 康美通 2.0.8 7.3MB 安全下载
 康美通 2.0.7 7.29MB 安全下载
 康美通 2.0.6 7.29MB 安全下载
 康美通 1.0.1 6.27MB 安全下载
 康美通 1.0beta
5.86MB 安全下载
[下载豌豆荚客户端](#) ([更多历史版本](#)) 下载
[康美通 历史版本年份合集](#)

```

1 相关历史版本
2
3 23.34MB最新版
4 康美通 4.3.0 23.34MB 安全下载
5 康美通 4.2.3 19.47MB 安全下载
6 康美通 4.2.2 19.29MB 安全下载
7 康美通 4.2.1 19.29MB 安全下载
8 康美通 4.2.0 19.18MB 安全下载
9 康美通 4.1.1 19.47MB 安全下载
10 康美通 4.1.0 19.48MB 安全下载
11 康美通 4.0 14.37MB 安全下载
12 康美通 3.2 14.22MB 安全下载
13 康美通 3.1 23.2MB 安全下载
14 康美通 3.0 23.1MB 安全下载
15 康美通 2.0.9 7.93MB 安全下载
16 康美通 2.0.8 7.3MB 安全下载
17 康美通 2.0.7 7.29MB 安全下载
18 康美通 2.0.6 7.29MB 安全下载
19 康美通 1.0.1 6.27MB 安全下载
20 康美通 1.0beta
21 5.86MB 安全下载
22 | 下载豌豆荚客户端 (更多历史版本)下载
23 康美通 历史版本年份合集
24
25 康美通 2018 年历史版本合集
26 康美通 2017 年历史版本合集
27 康美通 2016 年历史版本合集
28 康美通 2015 年历史版本合集

```

再去用正则：

安全下载

替换，得到我们要的：

相关历史版本：

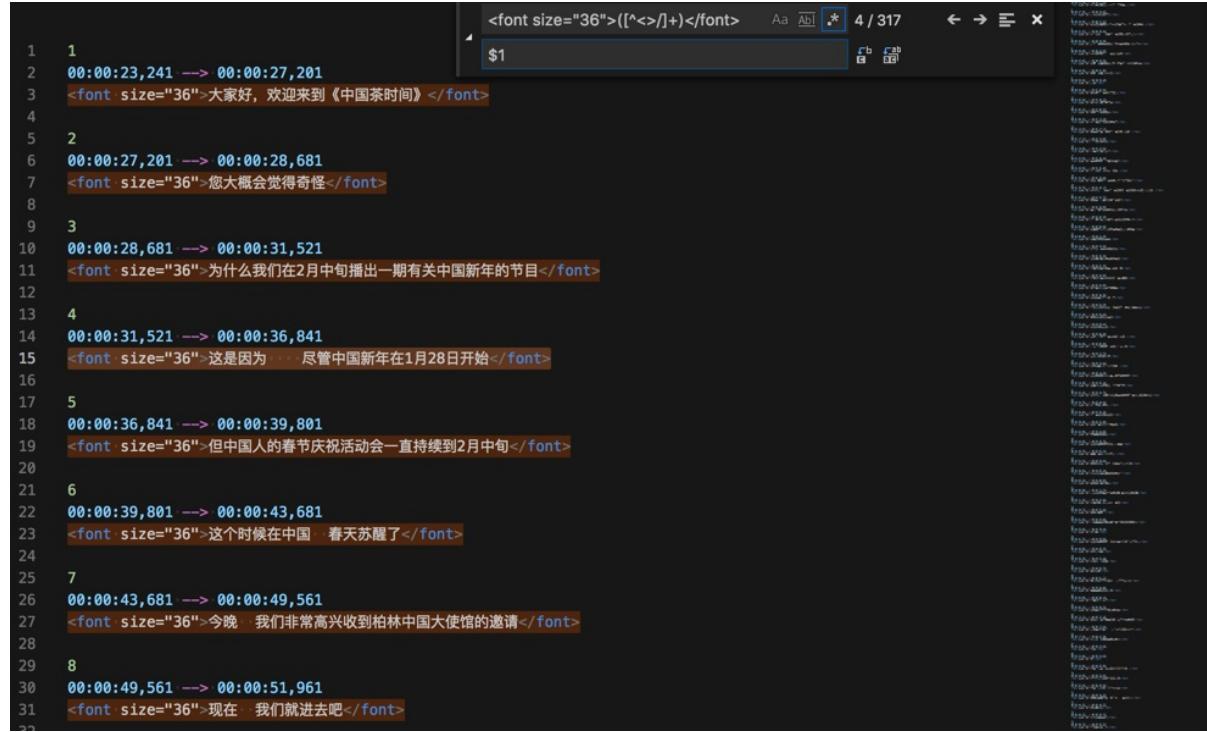
康美通 4.3.1 23.34MB
康美通 4.3.0 23.34MB
康美通 4.2.3 19.47MB
康美通 4.2.2 19.29MB
康美通 4.2.1 19.29MB
康美通 4.2.0 19.18MB
康美通 4.1.1 19.47MB
康美通 4.1.0 19.48MB
康美通 4.0 14.37MB
康美通 3.2 14.22MB
康美通 3.1 23.2MB
康美通 3.0 23.1MB
康美通 2.0.9 7.93MB
康美通 2.0.8 7.3MB
康美通 2.0.7 7.29MB
康美通 2.0.6 7.29MB
康美通 1.0.1 6.27MB
康美通 1.0beta 5.86MB

去掉srt字幕中font size

正则：

```
· font size="36">([^</]+) /font>
$1
```

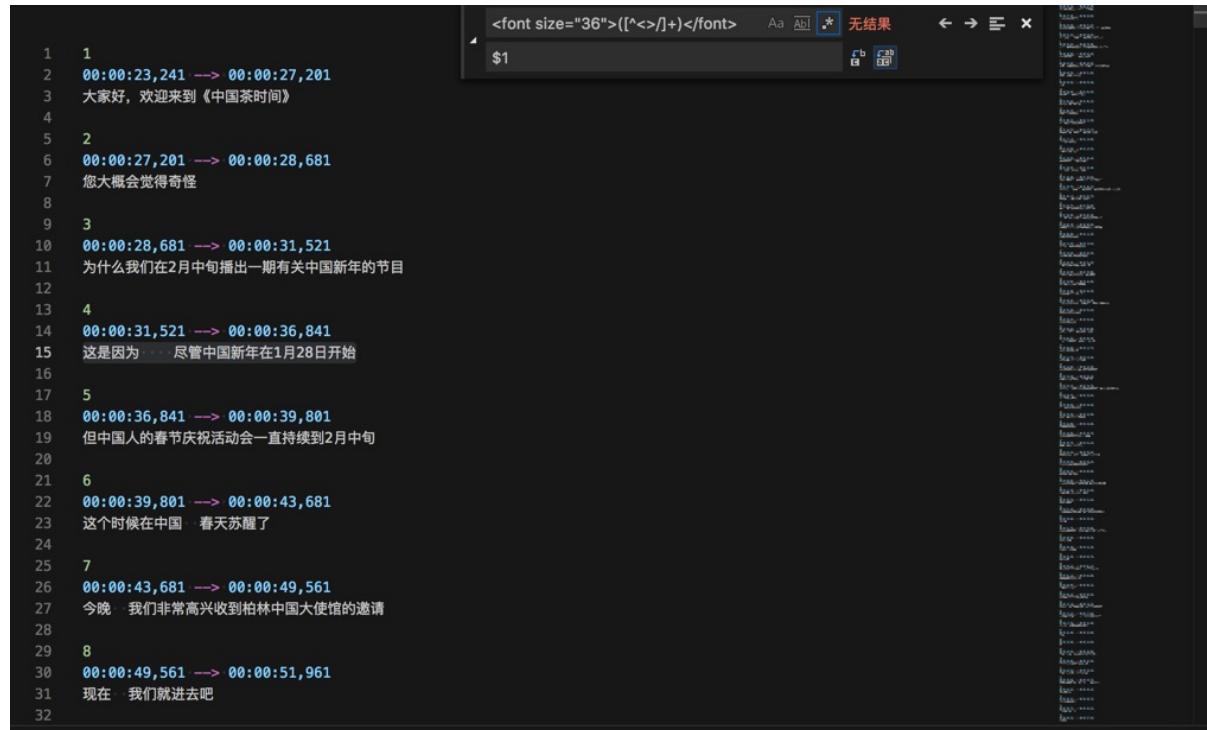
从：



The screenshot shows a code editor window in VSCode. The text is displayed in a monospaced font. Several lines contain the HTML tag . The search bar at the top contains the regular expression ([<>/]+) /font>. Below the search bar, the status bar shows '4 / 317'. The code editor interface includes tabs for 'File', 'Edit', 'View', 'Insert', 'Search', 'Run', and 'Terminal'.

```
1 1
2 00:00:23,241 --> 00:00:27,201
3 <font size="36">大家好，欢迎来到《中国茶时间》</font>
4
5 2
6 00:00:27,201 --> 00:00:28,681
7 <font size="36">您大概会觉得奇怪</font>
8
9 3
10 00:00:28,681 --> 00:00:31,521
11 <font size="36">为什么我们在2月中旬播出一期有关中国新年的节目</font>
12
13 4
14 00:00:31,521 --> 00:00:36,841
15 <font size="36">这是因为 ··· 尽管中国新年在1月28日开始</font>
16
17 5
18 00:00:36,841 --> 00:00:39,801
19 <font size="36">但中国人的春节庆祝活动会一直持续到2月中旬</font>
20
21 6
22 00:00:39,801 --> 00:00:43,681
23 <font size="36">这个时候在中国 ··· 春天苏醒了</font>
24
25 7
26 00:00:43,681 --> 00:00:49,561
27 <font size="36">今晚 ··· 我们非常高兴收到柏林中国大使馆的邀请</font>
28
29 8
30 00:00:49,561 --> 00:00:51,961
31 <font size="36">现在 ··· 我们就进去吧</font>
```

替换成：



The screenshot shows the same code editor window after the replacement. The search bar now displays '无结果' (No results). The rest of the interface and code content remain the same as in the previous screenshot.

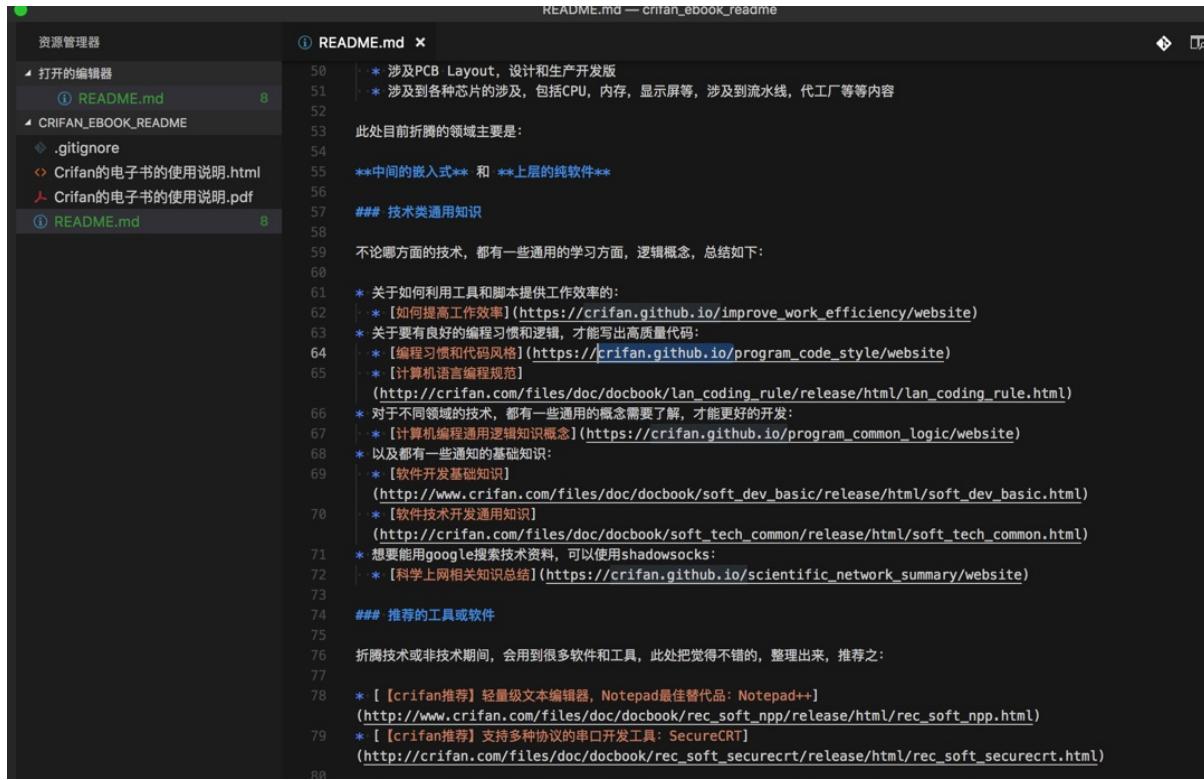
```
1 1
2 00:00:23,241 --> 00:00:27,201
3 大家好，欢迎来到《中国茶时间》
4
5 2
6 00:00:27,201 --> 00:00:28,681
7 您大概会觉得奇怪
8
9 3
10 00:00:28,681 --> 00:00:31,521
11 为什么我们在2月中旬播出一期有关中国新年的节目
12
13 4
14 00:00:31,521 --> 00:00:36,841
15 这是因为 ··· 尽管中国新年在1月28日开始
16
17 5
18 00:00:36,841 --> 00:00:39,801
19 但中国人的春节庆祝活动会一直持续到2月中旬
20
21 6
22 00:00:39,801 --> 00:00:43,681
23 这个时候在中国 ··· 春天苏醒了
24
25 7
26 00:00:43,681 --> 00:00:49,561
27 今晚 ··· 我们非常高兴收到柏林中国大使馆的邀请
28
29 8
30 00:00:49,561 --> 00:00:51,961
31 现在 ··· 我们就进去吧
```

crifan电子书中链接替换

对于我的电子书的说明：

https://github.com/crifan/crifan_ebook_readme

的markdown源码：



```

README.md — crifan_ebook_readme
-----



资源管理器
-----



 README.md 8
CRIFAN_EBOOK_README
 .gitignore
 Crifan的电子书的使用说明.html
 Crifan的电子书的使用说明.pdf
 README.md 8

 README.md x
-----



50  * 涉及PCB Layout, 设计和生产开发版
51  * 涉及到各种芯片的涉及, 包括CPU, 内存, 显示屏等, 涉及到流水线, 代工厂等等内容
52
53  此处目前折腾的领域主要是:
54
55  **中间的嵌入式** 和 **上层的纯软件**
56
57  ### 技术类通用知识
58
59  不论哪方面的技术, 都有一些通用的学习方面, 逻辑概念, 总结如下:
60
61  * 关于如何利用工具和脚本提供工作效率的:
62  * [如何提高工作效率] (https://crifan.github.io/improve\_work\_efficiency/website)
63  * 关于要有良好的编程习惯和逻辑, 才能写出高质量代码:
64  * [编程习惯和代码风格] (https://crifan.github.io/program\_code\_style/website)
65  * [计算机语言编程规范]
   (http://crifan.com/files/doc/docbook/lan\_coding\_rule/release/html/lan\_coding\_rule.html)
66  * 对于不同领域的技术, 都有一些通用的概念需要了解, 才能更好的开发:
67  * [计算机编程通用逻辑知识概念] (https://crifan.github.io/program\_common\_logic/website)
68  * 以及都有一些通知的基础知识:
69  * [软件开发基础知识]
   (http://www.crifan.com/files/doc/docbook/soft\_dev\_basic/release/html/soft\_dev\_basic.html)
70  * [软件技术开发通用知识]
   (http://crifan.com/files/doc/docbook/soft\_tech\_common/release/html/soft\_tech\_common.html)
71  * 想要能用google搜索技术资料, 可以使用shadowsocks:
72  * [科学上网相关知识总结] (https://crifan.github.io/scientific\_network\_summary/website)
73
74  ### 推荐的工具或软件
75
76  折腾技术或非技术期间, 会用到很多软件和工具, 此处把觉得不错的, 整理出来, 推荐之:
77
78  * [【crifan推荐】轻量级文本编辑器, Notepad最佳替代品: Notepad++]
   (http://www.crifan.com/files/doc/docbook/rec\_soft\_npp/release/html/rec\_soft\_npp.html)
79  * [【crifan推荐】支持多种协议的串口开发工具: SecureCRT]
   (http://crifan.com/files/doc/docbook/rec\_soft\_securecrt/release/html/rec\_soft\_securecrt.html)
80

```

想要把其中的地址：

<https://crifan.github.io/xxx/website>

替换为：

<https://book.crifan.com/books/xxx/website/>

比如：

https://crifan.github.io/program_code_style/website

替换成：

https://book.crifan.com/books/program_code_style/website/

用正则：

```

https://crifan.github.io/(\w+)/website/
https://book.crifan.com/books/$1/website/

```

实现从：

README.md — crifan_ebook_readme

① README.md ×

```
53 此处目前折腾的领域主要是：  
54  
55 **中间的嵌入式** 和 **上层的纯软件**  
56  
57 #### 技术类通用知识  
58  
59 不论哪方面的技术，都有一些通用的学习方面，逻辑概念，总结如下：  
60  
61 * 关于如何利用工具和脚本提供工作效率的：  
62   * [如何提高工作效率] (https://crifan.github.io/improve\_work\_efficiency/website)  
63 * 关于要有良好的编程习惯和逻辑，才能写出高质量代码：  
64   * [编程习惯和代码风格] (https://crifan.github.io/program\_code\_style/website)  
65   * [计算机语言编程规范]  
     (http://crifan.com/files/doc/docbook/lan\_coding\_rule/release/html/lan\_coding\_rule.html)  
66 * 对于不同领域的技术，都有一些通用的概念需要了解，才能更好的开发：  
67   * [计算机编程通用逻辑知识概念] (https://crifan.github.io/program\_common\_logic/website)  
68 * 以及都有一些通知的基础知识：  
69   * [软件开发基础知识]  
     (http://www.crifan.com/files/doc/docbook/soft\_dev\_basic/release/html/soft\_dev\_basic.html)  
70   * [软件技术开发通用知识]  
     (http://crifan.com/files/doc/docbook/soft\_tech\_common/release/html/soft\_tech\_common.html)  
71 * 想要能用google搜索技术资料，可以使用shadowsocks：  
72   * [科学上网相关知识总结] (https://crifan.github.io/scientific\_network\_summary/website)  
73  
74 #### 推荐的工具或软件  
75  
76 折腾技术或非技术期间，会用到很多软件和工具，此处把觉得不错的，整理出来，推荐之：  
77  
78 * [crifan推荐] 轻量级文本编辑器，Notepad最佳替代品：Notepad++  
  (http://www.crifan.com/files/doc/docbook/rec\_soft\_npp/release/html/rec\_soft\_npp.html)  
79 * [crifan推荐] 支持多种协议的串口开发工具：SecureCRT  
  (http://crifan.com/files/doc/docbook/rec\_soft\_securecrt/release/html/rec\_soft\_securecrt.html)  
80  
81 #### 硬件类  
82  
83 折腾嵌入式期间，其实也想去了解硬件方面的知识，只不过没有深入。  
84  
85 只整理了点和硬件相关的皮毛：
```

替换为：

README.md — crifan_ebook_readme

README.md •

```

13 此处目前折腾的领域主要是:
14
15 **中间的嵌入式** 和 **上层的纯软件**
16
17 ### 技术类通用知识
18
19 不论哪方面的技术，都有一些通用的学习方面，逻辑概念，总结如下：
20
21 * 关于如何利用工具和脚本提供工作效率的：
22   * [如何提高工作效率] (https://book.crifan.com/books/improve\_work\_efficiency/website/)
23 * 关于要有良好的编程习惯和逻辑，才能写出高质量代码：
24   * [编程习惯和代码风格] (https://book.crifan.com/books/program\_code\_style/website/)
25   * [计算机语言编程规范]
26     (http://crifan.com/files/doc/docbook/lan\_coding\_rule/release/html/lan\_coding\_rule.html)
27 * 对于不同领域的技术，都有一些通用的概念需要了解，才能更好的开发：
28   * [计算机编程运用逻辑知识概念] (https://book.crifan.com/books/program\_common\_logic/website/)
29 * 以及都有一些通知的基础知识：
30   * [软件开发基础知识]
31     (http://www.crifan.com/files/doc/docbook/soft\_dev\_basic/release/html/soft\_dev\_basic.html)
32   * [软件技术开发通用知识]
33     (http://crifan.com/files/doc/docbook/soft\_tech\_common/release/html/soft\_tech\_common.html)
34 * 想要能用google搜索技术资料，可以使用shadowsocks：
35   * [科学上网相关知识总结] (https://book.crifan.com/books/scientific\_network\_summary/website/)
36
37 ### 推荐的工具或软件
38
39 折腾技术或非技术期间，会用到很多软件和工具，此处把觉得不错的，整理出来，推荐之：
40
41 * [ [crifan推荐] 轻量级文本编辑器，Notepad最佳替代品: Notepad++]
42   (http://www.crifan.com/files/doc/docbook/rec\_soft\_npp/release/html/rec\_soft\_npp.html)
43 * [ [crifan推荐] 支持多种协议的串口开发工具: SecureCRT]
44   (http://crifan.com/files/doc/docbook/rec\_soft\_securecrt/release/html/rec\_soft\_securecrt.html)
45
46 ### 硬件类
47
48 折腾嵌入式期间，其实也想去了解硬件方面的知识，只不过没有深入。
49
50 只整理了点和硬件相关的皮毛：
51
52

```

将Chrome中拷贝出来的cookie处理成代码中要的dict

Chrome中拷贝出来的cookie是：

```
welcomeflash 20050606_107001; zzpaneluin=; zzpanelkey=; pgv_pvi=7640393728; pgv_si=s314729
8816; pgv_pvid=3951804270; pgv_info ssid=s7487670374; ptisp=ctc; ptui_loginuin=2539619267;
pt2gguin=o2539619267; uin=o2539619267; skey=@nDTkOJm1m; RK=Ye5Jmtb0ly; ptcz=3b6806bf7cdcc
375bc2d23b04ff9c366b47fac808b9256322ad69a23f2dc580f; p_uin=o2539619267; pt4_token=aC5vGfNA
A2M3fS7ngcAHdXoiCvqwrAGcEuL54gs63oE_=; p_skey=kSC7q75Gk93gLlo*mRMg*h2m3iYUuubjQqVBIGEMi*o_=;
Loading Yes
```

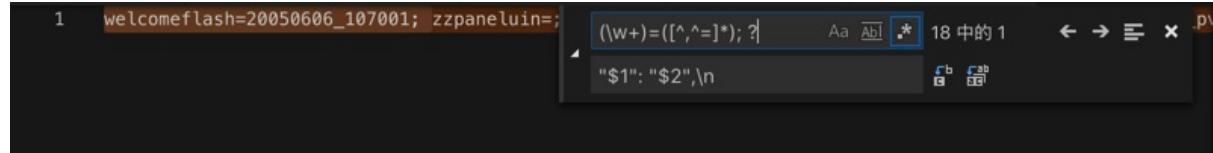
最后加上分号：

```
welcomeflash 20050606_107001; zzpaneluin=; zzpanelkey=; pgv_pvi=7640393728; pgv_si=s314729
8816; pgv_pvid=3951804270; pgv_info ssid=s7487670374; ptisp=ctc; ptui_loginuin=2539619267;
pt2gguin=o2539619267; uin=o2539619267; skey=@nDTkOJm1m; RK=Ye5Jmtb0ly; ptcz=3b6806bf7cdcc
375bc2d23b04ff9c366b47fac808b9256322ad69a23f2dc580f; p_uin=o2539619267; pt4_token=aC5vGfNA
A2M3fS7ngcAHdXoiCvqwrAGcEuL54gs63oE_=; p_skey=kSC7q75Gk93gLlo*mRMg*h2m3iYUuubjQqVBIGEMi*o_=;
Loading Yes;
```

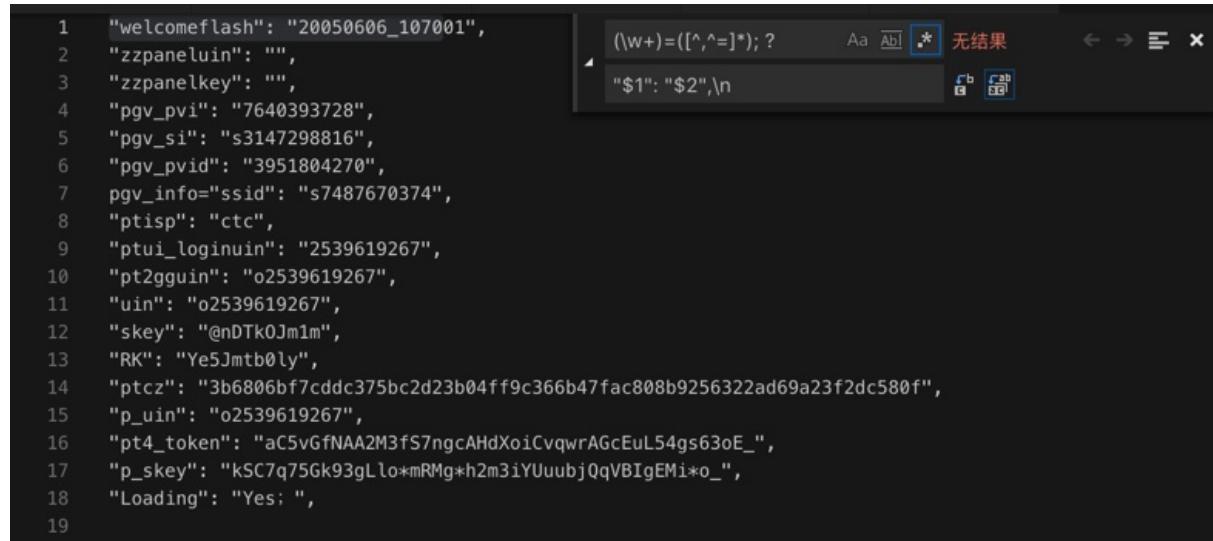
再去用正则：

```
(\w+)=([^\n]*); ?
"\$1": "$2",\n
```

替换:



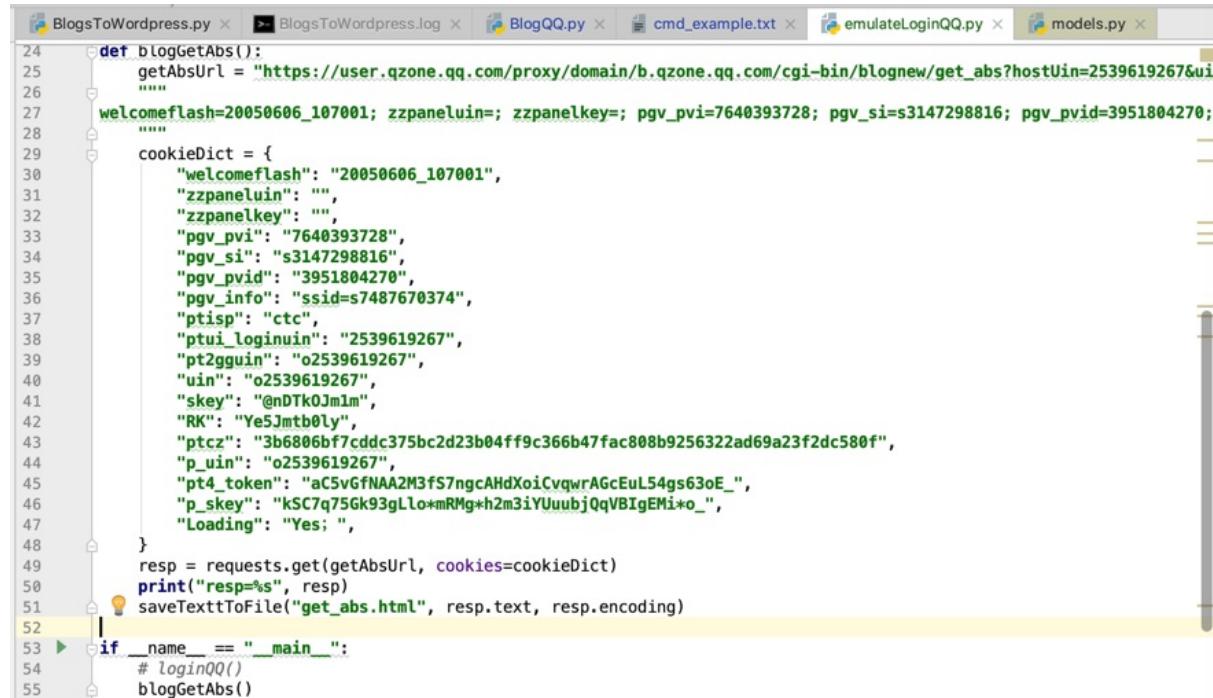
成自己要的dict的内容:



中间有个特殊的，自己手动改一下即可:

```
"welcomeflash": "20050606_107001",
"zzpaneluin": "",
"zzpanelkey": "",
"pgv_pvi": "7640393728",
"pgv_si": "s3147298816",
"pgv_pvid": "3951804270",
"pgv_info": "ssid=s7487670374",
"ptisp": "ctc",
"ptui_loginuin": "2539619267",
"pt2gguin": "o2539619267",
"uin": "o2539619267",
"skey": "@nDTkOJm1m",
"RK": "Ye5Jmtb0ly",
"ptcz": "3b6806bf7cddc375bc2d23b04ff9c366b47fac808b9256322ad69a23f2dc580f",
"p_uin": "o2539619267",
"pt4_token": "aC5vGfNAA2M3fS7ngcAHdXoiCvqwrAGcEuL54gs63oE_",
"p_skey": "kSC7q75Gk93gLlo*mRMg*h2m3iYUuubjQqVBIgEMi*o_",
"Loading": "Yes;" ,
```

粘贴到代码中即可使用了:



```

24     def blogGetAbs():
25         getAbsUrl = "https://user.qzone.qq.com/proxy/domain/b.qzone.qq.com/cgi-bin/blognew/get_abs?hostUin=2539619267&ui
26         .....
27         welcomeflash=20050606_107001; zzpaneluin=; zzpanelkey=; pgv_pvi=7640393728; pgv_si=s3147298816; pgv_pvid=3951804270;
28         .....
29         cookieDict = {
30             "welcomeflash": "20050606_107001",
31             "zzpaneluin": "",
32             "zzpanelkey": "",
33             "pgv_pvi": "7640393728",
34             "pgv_si": "s3147298816",
35             "pgv_pvid": "3951804270",
36             "pgv_info": "ssid=s7487670374",
37             "ptisp": "ctc",
38             "ptui_loginuin": "2539619267",
39             "pt2gguin": "o2539619267",
40             "uin": "o2539619267",
41             "skey": "@nDTk0Jm1m",
42             "RK": "Ye5Jmtb0ly",
43             "ptcz": "3b6806bf7cdcc375bc2d23b04ff9c366b47fac808b9256322ad69a23f2dc580f",
44             "p_uin": "o2539619267",
45             "pt4_token": "aC5vGfNAA2M3fS7ngcAHdXoiCvqwrAGcEuL54gs63oE_",
46             "p_skey": "kSC7q75Gk93gLlo+mRMg*h2m3iYUubjQqVBIgEMi+o_",
47             "Loading": "Yes; ",
48         }
49         resp = requests.get(getAbsUrl, cookies=cookieDict)
50         print("resp=%s", resp)
51         saveTextToFile("get_abs.html", resp.text, resp.encoding)
52
53     if __name__ == "__main__":
54         # loginQQ()
55         blogGetAbs()

```

处理得到城市名称

除了：

【整理】中国常见的城市的名字

以及：

对于：

https://en.wikipedia.org/wiki/List_of_urban_areas_by_population

中的城市名，用正则：

\[\d+\]

去除掉 [数字]

从：

```

1 Tokyo-Yokohama[4]
2 Jakarta ( Greater Jakarta)[5]
3 Delhi (CNCR)[6]
4 Manila (Metro Manila)[7]
5 Seoul-Incheon (Seoul National Capital Area)[8]
6 Shanghai[9]
7 Mumbai[10]
8 New York City[11]
9 Beijing[12]
10 São Paulo[13]
11 Mexico City (Valley of Mexico)[14]
12 Guangzhou-Foshan(Guangfo)[15]
13 Dhaka[16]
14 Osaka-Kobe-Kyoto(Keihanshin)[17]
15 Moscow[18]
16 Greater Cairo
17 Bangkok
18 Los Angeles[19]
19 Buenos Aires[20]
20 Kolkata
21 Istanbul
22 Tehran[21]
23 Lagos
24 Tianjin[22]
25 Karachi[23]
26 Shenzhen[15]
27 Kinshasa[24]
28 Rio de Janeiro
29 Chengdu
30 Lima
31 Lahore
32 Paris
33 Bangalore
34 Ho Chi Minh City(Saigon)
35 London[25]
36 Chennai
37 Nagoya (Chūkyō)[26]
38 Bogotá
39 Hyderabad

```

替换为：

```

1 Tokyo-Yokohama
2 Jakarta ( Greater Jakarta)
3 Delhi (CNCR)
4 Manila (Metro Manila)
5 Seoul-Incheon (Seoul National Capital Area)
6 Shanghai
7 Mumbai
8 New York City
9 Beijing
10 São Paulo
11 Mexico City (Valley of Mexico)
12 Guangzhou-Foshan(Guangfo)
13 Dhaka
14 Osaka-Kobe-Kyoto(Keihanshin)
15 Moscow
16 Greater Cairo
17 Bangkok
18 Los Angeles
19 Buenos Aires
20 Kolkata
21 Istanbul
22 Tehran
23 Lagos
24 Tianjin
25 Karachi
26 Shenzhen
27 Kinshasa
28 Rio de Janeiro
29 Chengdu
30 Lima
31 Lahore
32 Paris
33 Bangalore
34 Ho Chi Minh City(Saigon)
35 London
36 Chennai
37 Nagoya (Chūkyō)
38 Bogotá
39 Hyderabad

```

以及继续用：

```
\(((\w+\s)+)\)$  
$1
```

把 (xxx) 中的 xxx 放到下一行

从：

```

1 Tokyo-Yokohama
2 Jakarta ( Greater Jakarta)
3 Delhi (CNR)
4 Manila (Metro Manila)
5 Seoul-Incheon (Seoul National Capital Area)
6 Shanghai
7 Mumbai
8 New York City
9 Beijing
10 São Paulo
11 Mexico City (Valley of Mexico)
12 Guangzhou-Foshan(Guangfo)
13 Dhaka
14 Osaka-Kobe-Kyoto(Keihanshin)
15 Moscow
16 Greater Cairo
17 Bangkok
18 Los Angeles
19 Buenos Aires
20 Kolkata
21 İstanbul
22 Tehran
23 Lagos
24 Tianjin
25 Karachi
26 Shenzhen
27 Kinshasa
28 Rio de Janeiro
29 Chengdu
30 Lima
31 Lahore
32 Paris
33 Bangalore
34 Ho Chi Minh City(Saigon)
35 London
36 Chennai
37 Nagoya (Chūkyō)
38 Bogotá
39 Hyderabad

```

替换为：

```

8 Seoul-Incheon
9 Seoul National Capital Area
10 Shanghai
11 Mumbai
12 New York City
13 Beijing
14 São Paulo
15 Mexico City
16 Valley of Mexico
17 Guangzhou-Foshan
18 Guangfo
19 Dhaka
20 Osaka-Kobe-Kyoto
21 Keihanshin
22 Moscow
23 Greater Cairo
24 Bangkok
25 Los Angeles
26 Buenos Aires
27 Kolkata
28 İstanbul
29 Tehran
30 Lagos
31 Tianjin
32 Karachi
33 Shenzhen
34 Kinshasa
35 Rio de Janeiro
36 Chengdu
37 Lima
38 Lahore
39 Paris
40 Bangalore
41 Ho Chi Minh City
42 Saigon
43 London
44 Chennai
45 Nagoya (Chūkyō)
46 Bogotá

```

当然，也注意到了，没有匹配到：

Nagoya (Chūkyō)

是因为里面有unicode的字符，由于数量不多，手动处理即可。

再去用：

```
-([ \w\s ]+)
\n$1
```

把 xxx-yyy 中的 yyy 放到下一行



```
14 São Paulo
15 Mexico City
16 Valley of Mexico
17 Guangzhou-Foshan
18 Guangfo
19 Dhaka
20 Osaka-Kobe-Kyoto
21 Keihanshin
22 Moscow
23 Greater Cairo
24 Bangkok
25 Los Angeles
26 Buenos Aires
27 Kolkata
28 Istanbul
29 Tehran
30 Lagos
31 Tianjin
32 Karachi
33 Shenzhen
34 Kinshasa
35 Rio de Janeiro
36 Chengdu
37 Lima
38 Lahore
39 Paris
40 Bangalore
41 Ho Chi Minh City
42 Saigon
43 London
44 Chennai
45 Nagoya
46 Chūkyō
47 Bogotá
48 Hyderabad
49 Chicago
50 Johannesburg-East Rand
```

替换成：

```

14 Beijing
15 São Paulo
16 Mexico City
17 Valley of Mexico
18 Guangzhou
19 Foshan
20 Guangfo
21 Dhaka
22 Osaka
23 Kobe
24 Kyoto
25 Keihanshin
26 Moscow
27 Greater Cairo
28 Bangkok
29 Los Angeles
30 Buenos Aires
31 Kolkata
32 Istanbul
33 Tehran
34 Lagos
35 Tianjin
36 Karachi
37 Shenzhen
38 Kinshasa
39 Rio de Janeiro
40 Chengdu
41 Lima
42 Lahore
43 Paris
44 Bangalore
45 Ho Chi Minh City
46 Saigon
47 London
48 Chennai
49 Nagoya
50 Chūkyō
51 Bogotá
52 Hyderabad

```

行 23, 列 5 (已选择5) 空格: 2 UTF-8 LF 纯文本 ☺ 🔔

以及，用：

```
\s*\((.+)\)$
```

把 xxx (yyy) 中的 空格(yyy) 去掉：

```

214 Samantha
215 Sophia
216 Andrea
217 Angela
218 Janine
219 Sofia
220 Seo-yeon (서연)
221 Seo-yeon(서연)
222 Ji-woo (지우)
223 Seo-hyeon(서현)
224 Min-seo (민서)
225 Yun-seo(윤서)
226 Chae-won(채원)
227 Ha-yoon (하윤)
228 Ji-ah (자아)
229 Eun-seo(은서)
230 Fatima
231 Aisha
232 Nora
233 Hessa
234 Sheikha
235 Maha
236 Shu-fen (淑芬)
237 Shu-hui (淑惠)
238 Mei-ling (美玲)
239 Ya-ting (雅婷)
240 Mei-hui (美蕙)
241 Li-hua (麗華)
242 Shu-chuan (淑娟)
243 Shu-chen (淑貞)
244 I-chun (怡君)
245 Shu-hua (淑華)
246 Sumayah
247 Asiya
248 Oisha
249 Googoosh
250 Anohito
251 Indira
252 Mariam

```

行 145, 列 16 (已选择9) 空格: 2 UTF-8 LF 纯文本 ☺ 🔔

替换为：

```
145 Ayzere
146 Inzhu
147 Ayaru
148 Käwsar
149 Ayşa
150 Aruzhan
151 Amina
152 Ayaulym
153 Sezim
154 İhkär
155 Fatma
156 Mariam
157 Hussa
158 Sherifa
159 Sara
160 Reem
161 Aisha
162 Dalal
163 Lulwa
164 Shaikha
165 Marie
166 Fatima
167 Jessica
168 Zeinab
169 Mariam
170 Sarah
171 Maya
172 Layla
173 Christina
174 Amal
175 Nor
176 Hannah
177 Aishah
178 Siti
179 Zara
180 Puteri
181 Nurul
182 Sophia
183 Sara
```

再继续，用：

`^\s`
`\n`

可以找到：

有哪些单词在行首有多余的空格（后续可以再去删除掉）：

提取mp3文件名和mp3链接地址

正则：

```
^\^r\n]+href\s+"(\w+\])\(\.\./\./assets/img/vscode_remove_start_empty.png)
```

```
```bash
^[\^\\r\\n]
$1
```

把：

 <a href="e10d3a.mp3">e10d3a.mp3</a> 2015-09-23 0

```
9:10 9.3M
 e10d3b.mp3 2015-09-23 0
9:10 7.0M
 e10d5a.mp3 2015-09-23 0
9:11 54M
```

替换为：

```
e10d3a.mp3
e10d3b.mp3
e10d5a.mp3
```

再进一步：

用正则：

```
^\^[\r\n]+href "(\w+\.\mp3)" [\r\n]+$
http://media.talkbank.org/CHILDES/Biling/Singapore/$1
```

从：

```
 e10d3a.mp3 2015-09-23 0
9:10 9.3M
 e10d3b.mp3 2015-09-23 0
9:10 7.0M
 e10d5a.mp3 2015-09-23 0
9:11 54M
```

替换和提取出：

```
http://media.talkbank.org/CHILDES/Biling/Singapore/e10d3a.mp3
http://media.talkbank.org/CHILDES/Biling/Singapore/e10d3b.mp3
http://media.talkbank.org/CHILDES/Biling/Singapore/e10d5a.mp3
```

详见：

[【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](#)

## 去除掉csv中多余的 = "xxx"

客户给的一个数据文件csv格式的，但是内部内容中发现有多余的 `= "xxx"`，应该改为 `xxx` 才对。

所以用VSCode去替换，用正则：

```
"(.+?)"
$1
```

实现了，把 `="7xxx1"` :

```
大东南区,上海二区,"7(1",上海有限公司,"18 3",2018/04/ ="(.+?)" Aa Abi 1 / 19999+ ← → ⌂ × 56
大东南区,上海二区,"7(1",上海有限公司,"18 4",2018/04/ ="(.+?)" 38
大东南区,上海二区,"7(1",上海有限公司,"18 5",2018/04/ $1 69
大东南区,上海二区,"7(1",上海有限公司,"18 6",2018/04/ ="(.+?)" 2
大东南区,上海二区,"7(1",上海有限公司,"18 7",2018/04/13,="2 7,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 8",2018/04/15,="2 8,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 9",2018/04/16,="2 9,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 10",2018/04/17,="2 10,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 11",2018/04/17,="2 11,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 12",2018/04/17,="2 12,LS 2,="(0
大东南区,上海二区,"7(1",上海有限公司,"18 13",2018/04/01,="39 13,LS 6,="(8
大东南区,上海二区,"7(1",上海有限公司,"18 14",2018/04/03,="99 14,LS 4,="(2
大东南区,上海二区,"7(1",上海有限公司,"18 15",2018/04/03,="100 15,LS 2,="(6

```

替换成 `7xxx1` :

```
大东南区,上海二区,7(1,上海有限公司,1 1,2018/04/02,201 ="(.+?)" Aa Abi 1 / 1563 ← → ⌂ × ,W
大东南区,上海二区,7(1,上海有限公司,1 2,2018/04/02,201 3,LS
大东南区,上海二区,7(1,上海有限公司,1 0,2018/04/02,201 $1 4,LS
大东南区,上海二区,7(1,上海有限公司,1 5,2018/04/03,201 5,LS
大东南区,上海二区,7(1,上海有限公司,1 4,2018/04/03,20 1,LS
大东南区,上海二区,7(1,上海有限公司,1 2,2018/04/09,20 0,LS
大东南区,上海二区,7(1,上海有限公司,1 3,2018/04/09,20 1,LS
大东南区,上海二区,7(1,上海有限公司,1 4,2018/04/09,20 3,LS
大东南区,上海二区,7(1,上海有限公司,1 9,2018/04/09,20 7,LS
大东南区,上海二区,7(1,上海有限公司,1 0,2018/04/09,20 8,LS
大东南区,上海二区,7(1,上海有限公司,1 0,2018/04/09,20 3,LS
大东南区,上海二区,7(1,上海有限公司,1 8,2018/04/09,20 4,LS
大东南区,上海二区,7(1,上海有限公司,1 3,2018/04/12,20 1,LS
大东南区,上海二区,7(1,上海有限公司,1 4,2018/04/12,20 0,LS
大东南区,上海二区,7(1,上海有限公司,1 5,2018/04/12,20 2,LS

```

即可。

## 给文章段落增加换行

用正则:

```
\n(\d.)
\n\n$1
```

把:

```
...互。
1.字节跳动
...说过。
2.陆奇给年轻人的话
...
8.网易 丁磊
```

53 上到大三汪滔就泄气了，“距离自己的梦想越来越远。”他  
 54 此后，汪滔向斯坦福、麻省理工等世界一流大学递上了申请  
 55 遗憾的是，当时美国的面试官没有发现这个来自中国内地的  
 56 唯独香港科技大学慧眼识珠，给汪滔发来了录取通知书，使  
 57 在港科大读书的汪滔对机器人设计充满兴趣，这门课程他选了两次，并且决定将从小的梦想——遥控直升机的飞行控制系统作为自己的毕业课题  
 58 拿着学校给的经费1.8万元港币，汪滔团队奋斗了5个月，经常要在凌晨四五点才能睡下。然而废寝忘食的努力，换来的结果却是在最终的演示  
 59 这次失败的毕业设计得了一个C，这个很差的成绩甚至让他失去了去欧洲名校继续深造的机会。  
 60 但收之桑榆的是，前文中机器人设计课程的老师李泽湘教授对汪滔青眼有加，他因此获得了在港科大师从李泽湘读研究生的机会。  
 61 **6. 赶集&瓜子 杨浩涌**  
 62 来源：人民网 赶集网CEO杨浩涌校友做客天津大学北洋大讲堂  
 63 他首先谈了自己艰苦的求学经历，话语之间略显激动，他打趣地将自己形容为一个“不安分”的人，不太爱上课，也常常有逃课的现象，曾经在  
 64 **7. 知乎 周源**  
 65 来源：成都理工大学公众号 成理人周源：创建知乎的体验  
 66 在周源读高中的时代，计算机专业很热门，他和周围不少同学很自然地对学习计算机产生兴趣。虽然周源当时对于学计算机意味着什么并没有  
 67 **8. 网易 丁磊**  
 68 来源：甬帮平谈：兴盛中华，网易有责——访网易公司创始人、CEO丁磊  
 69 总是碰到许多大学生问我毕业以后是怎么取得成功的，我说很遗憾，我大学选的专业并不是自己喜欢的。我很喜欢电脑，高中时就在苹果  
 70 据说这个系的学生毕业后历来是最难分配的，而且会被分配到山沟沟里去，所以我大一挺郁闷的。我想这是人生很重要的一步，它现在就这个  
 71 不过我一直没有放弃辅修计算机，我那时经常跑到图书馆去看计算机方面的书，还去计算机系蹭课旁听。当时我有两个困惑，第一，书本上的  
 72 因为我没有听第一堂课，又不得不做作业，所以我会努力去看老师上一堂讲的东西，也会很努力地去想老师想给我传达什么信息。很快我掌握  
 73 大学毕业后我被分配到宁波电信局，在那里度过了将近两年时光。我不喜欢电信局的环境，论资排辈很严重，年轻人没有什么机会，每天做的

变成：

...互。  
**1. 字节跳动**  
 ...说过。

**2. 陆奇给年轻人的话**

...

**8. 网易 丁磊**

57 来源：凤凰科技 马晓宁 留学遭拒，研究生“留级”三年，  
 58 高中毕业后，汪滔考入了上海本地一家大学（华东师范大学）  
 59 上到大三汪滔就泄气了，“距离自己的梦想越来越远。”他一  
 60 此后，汪滔向斯坦福、麻省理工等世界一流大学递上了申...  
 61 遗憾的是，当时美国的面试官没有发现这个来自中国内地的小青年有什么特别，更看不到十年后汪滔的成就，所以果断拒绝了汪滔。  
 62 唯独香港科技大学慧眼识珠，给汪滔发来了录取通知书，使得他得以进入了电子及计算机工程学系继续就读。  
 63 在港科大读书的汪滔对机器人设计充满兴趣，这门课程他选了两次，并且决定将从小的梦想——遥控直升机的飞行控制系统作为自己的毕业课题  
 64 拿着学校给的经费1.8万元港币，汪滔团队奋斗了5个月，经常要在凌晨四五点才能睡下。然而废寝忘食的努力，换来的结果却是在最终的演示  
 65 这次失败的毕业设计得了一个C，这个很差的成绩甚至让他失去了去欧洲名校继续深造的机会。  
 66 但收之桑榆的是，前文中机器人设计课程的老师李泽湘教授对汪滔青眼有加，他因此获得了在港科大师从李泽湘读研究生的机会。  
 67 **6. 赶集&瓜子 杨浩涌**  
 68 来源：人民网 赶集网CEO杨浩涌校友做客天津大学北洋大讲堂  
 69 他首先谈了自己艰苦的求学经历，话语之间略显激动，他打趣地将自己形容为一个“不安分”的人，不太爱上课，也常常有逃课的现象，曾经在  
 70 **7. 知乎 周源**  
 71 来源：成都理工大学公众号 成理人周源：创建知乎的体验  
 72 在周源读高中的时代，计算机专业很热门，他和周围不少同学很自然地对学习计算机产生兴趣。虽然周源当时对于学计算机意味着什么并没有  
 73 **8. 网易 丁磊**  
 74 来源：甬帮平谈：兴盛中华，网易有责——访网易公司创始人、CEO丁磊  
 75 总是碰到许多大学生问我毕业以后是怎么取得成功的，我说很遗憾，我大学选的专业并不是自己喜欢的。我很喜欢电脑，高中时就在苹果  
 76 据说这个系的学生毕业后历来是最难分配的，而且会被分配到山沟沟里去，所以我大一挺郁闷的。我想这是人生很重要的一步，它现在就这个  
 77 不过我一直没有放弃辅修计算机，我那时经常跑到图书馆去看计算机方面的书，还去计算机系蹭课旁听。当时我有两个困惑，第一，书本上的  
 78 因为我没有听第一堂课，又不得不做作业，所以我会努力去看老师上一堂讲的东西，也会很努力地去想老师想给我传达什么信息。很快我掌握  
 79 大学毕业后我被分配到宁波电信局，在那里度过了将近两年时光。我不喜欢电信局的环境，论资排辈很严重，年轻人没有什么机会，每天做的

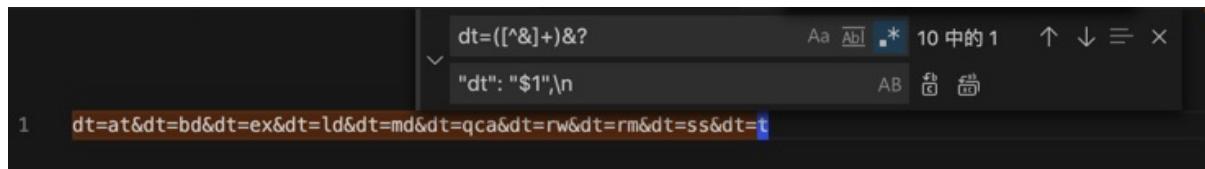
把url中查询参数换成代码中字典参数

用正则：

```
dt ([^&]+)&
"dt": "$1",\n
```

把：

```
dt at dt bd dt ex dt ld dt md dt qca dt rw dt rm dt ss dt t
```

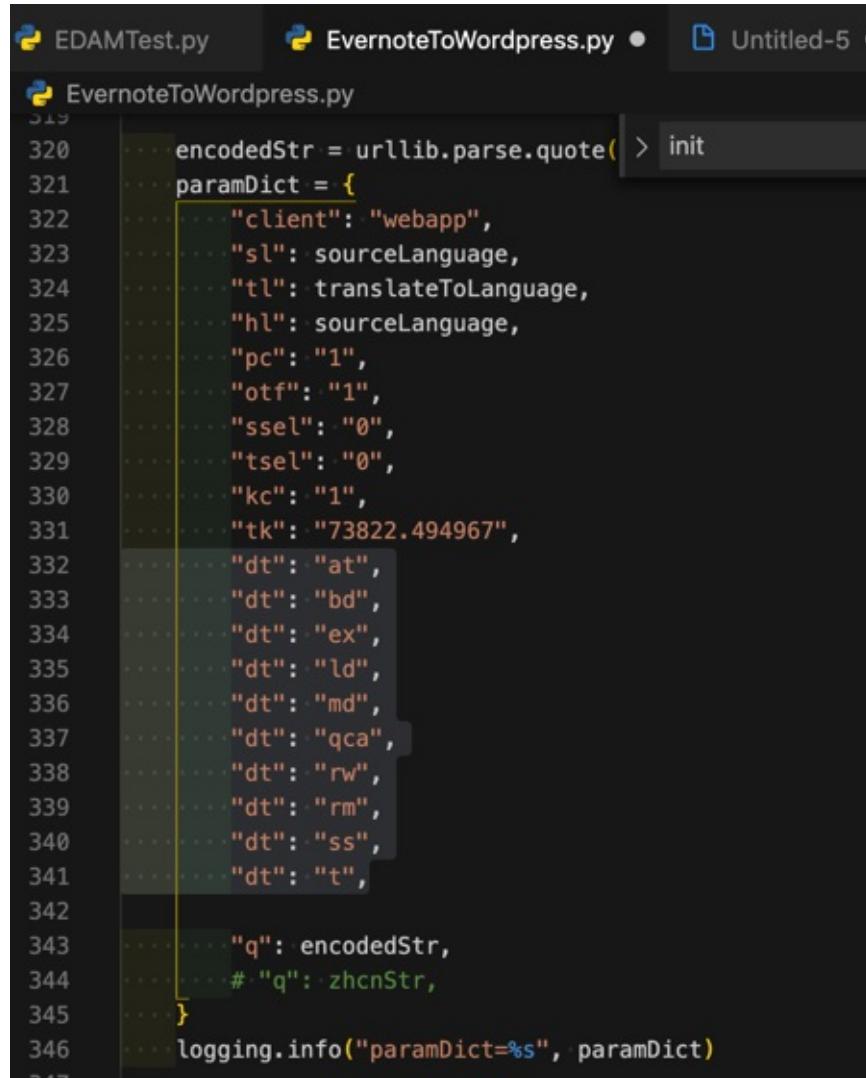


替换成：

```
"dt": "at",
"dt": "bd",
"dt": "ex",
"dt": "ld",
"dt": "md",
"dt": "qca",
"dt": "rw",
"dt": "rm",
"dt": "ss",
"dt": "t",
```



用于后续放到代码中使用：



```

EDAMTest.py EvernoteToWordpress.py • Untitled-5 •
EvernoteToWordpress.py
320 encodedStr = urllib.parse.quote(> init
321 paramDict = {
322 "client": "webapp",
323 "sl": sourceLanguage,
324 "tl": translateToLanguage,
325 "hl": sourceLanguage,
326 "pc": "1",
327 "otf": "1",
328 "ssel": "0",
329 "tsel": "0",
330 "kc": "1",
331 "tk": "73822.494967",
332 "dt": "at",
333 "dt": "bd",
334 "dt": "ex",
335 "dt": "ld",
336 "dt": "md",
337 "dt": "qca",
338 "dt": "rw",
339 "dt": "rm",
340 "dt": "ss",
341 "dt": "t",
342 }
343 "q": encodedStr,
344 # "q": zhcnStr,
345 }
346 logging.info("paramDict=%s", paramDict)
347

```

把每个词都加上引号，用于放代码中用

用正则：

```
(.+)
"$1",
```

从输入：

```
更新公告
签到奖励
离线经验
在线奖励
等级礼包
资源找回
活动时间
活动内容
累计充值
```

Untitled-1 — crifanLibPython

Untitled-1 ●

```
1 更新公告
2 签到奖励
3 离线经验
4 在线奖励
5 等级礼包
6 资源找回
7 活动时间
8 活动内容
9 累计充值
10
```

(.+)

Aa Abi \* 9 中的 9 ↕ ↓ ⌂ ×

AB ⌂ ⌂

变成：

```
"更新公告",
"签到奖励",
"离线经验",
"在线奖励",
"等级礼包",
"资源找回",
"活动时间",
"活动内容",
"累计充值",
```

Untitled-1 — crifanLibPython

Untitled-1 ●

```
1 "更新公告",
2 "签到奖励",
3 "离线经验",
4 "在线奖励",
5 "等级礼包",
6 "资源找回",
7 "活动时间",
8 "活动内容",
9 "累计充值",
10
```

(.+)

Aa Abi \* 9 中的 9 ↕ ↓ ⌂ ×

AB ⌂ ⌂

用于拷贝到代码中使用：

```

src > common > MainUtils.py > MainUtils > isPopupTypePage
1359 "已经升后",
1360 "丰厚奖励",
1361 "亲爱的玩家",
1362 "答题活动",
1363 # 圣诞狂欢累充活动
1364 "更新公告",
1365 "签到奖励",
1366 "离线经验",
1367 "在线奖励",
1368 "等级礼包",
1369 "资源找回",
1370 "活动时间",
1371 "活动内容",
1372 "累计充值",
1373
1374]
1375 optionalList.extend(possibleTitleList)
1376 optionalList.extend(otherOptionalList)

```

提取104协议示例数据，并格式化成java代码中字符串数组

从：

```

EB90EB90(端口号:21站0)[2020/1/6 18:18:58] 发送:
68 04 07 00 00 00
EB90EB90(端口号:21站5)[2020/1/6 18:19:07] 接收:
68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
EB90EB90(端口号:21站5)[2020/1/6 18:19:07] 发送:
68 04 01 00 E6 B7
EB90EB90(端口号:21站5)[2020/1/6 18:19:12] 接收:
68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
EB90EB90(端口号:21站5)[2020/1/6 18:19:12] 发送:
68 04 01 00 E8 B7
EB90EB90(端口号:21站5)[2020/1/6 18:19:17] 接收:
68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01
00 FE FF 7C 00 94 02 00
00 00 00 86 13 00 00 02 00 00 00 00 00 FE FF 00 00 00 00 87 00 00 00 00 00 00 02 00 00 00 00 00 00
00
...
EB90EB90(端口号:21站5)[2020/1/6 23:08:55] 发送:
68 04 01 00 12 D3
EB90EB90(端口号:21站5)[2020/1/6 23:09:00] 接收:
68 59 14 D3 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 BC 5D 88 5D A7 5D 44 00 01
00 FE FF 7C 00 84 02 00
00 00 00 88 13 00 00 02 00 00 00 00 00 FE FF 00 00 00 00 B6 00 00 00 00 00 00 00 02 00 00 00 00 00
00
EB90EB90(端口号:21站5)[2020/1/6 23:09:06] 接收:
68 12 16 D3 00 00 0F 81 05 00 05 00 01 0C 00 B6 42 03 00 00
EB90EB90(端口号:21站5)[2020/1/6 23:09:06] 发送:
68 04 01 00 16 D3

```

提取出：接收：的下一行的一连串数字

(1) 先去除发送的部分

正则：

```
^EB.+发送: \n(^.+$)\n
```

The screenshot shows a code editor window in VSCode. The title bar includes tabs for 'Analysis104Test.java', 'SmartElectric\_104\_sample\_data.txt', 'Untitled-1', and 'Untitled-2'. A search bar at the top contains the regular expression '^EB.+发送: \n(^.+\$)\n'. Below the search bar, there is a dropdown menu with the option '替换' (Replace). The main area displays a log file with numerous lines of text. Several lines are highlighted in orange, specifically lines 136, 140, 142, 144, 146, 152, 154, 155, 156, 158, and 161. These lines represent '发送' (Send) messages. The text in these lines is partially redacted with the 'AB' placeholder.

替换为：

```

1 EB90EB90(端口号:21站5)[2020/1/6 18:19:07] 接收:
2 68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
3 EB90EB90(端口号:21站5)[2020/1/6 18:19:12] 接收:
4 68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
5 EB90EB90(端口号:21站5)[2020/1/6 18:19:17] 接收:
6 68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01 00
7 EB90EB90(端口号:21站5)[2020/1/6 18:19:22] 接收:
8 68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
9 EB90EB90(端口号:21站5)[2020/1/6 18:19:27] 接收:
10 68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
11 EB90EB90(端口号:21站5)[2020/1/6 18:19:32] 接收:
12 68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F2 5C 45 00 01 00
13 EB90EB90(端口号:21站5)[2020/1/6 18:19:37] 接收:
14 68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
15 EB90EB90(端口号:21站5)[2020/1/6 18:19:42] 接收:
16 68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
17 EB90EB90(端口号:21站5)[2020/1/6 18:19:47] 接收:
18 68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E8 5C 45 00 01 00
19 EB90EB90(端口号:21站5)[2020/1/6 18:19:52] 接收:
20 68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
21 EB90EB90(端口号:21站5)[2020/1/6 18:19:57] 接收:
22 68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
23 EB90EB90(端口号:21站5)[2020/1/6 18:21:57] 接收:
24 68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
25 EB90EB90(端口号:21站5)[2020/1/6 18:22:02] 接收:
26 68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 79 5D 5A 5D 6C 5D 45 00 01 00
27 EB90EB90(端口号:21站5)[2020/1/6 18:22:07] 接收:
28 68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 00

```

(2) 再去把接收部分中数字提取出来

从：

```

EB90EB90(端口号:21站5)[2020/1/6 18:19:07] 接收:
68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
EB90EB90(端口号:21站5)[2020/1/6 18:19:12] 接收:
68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
...
EB90EB90(端口号:21站5)[2020/1/6 23:08:55] 接收:
68 14 12 D3 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
EB90EB90(端口号:21站5)[2020/1/6 23:09:01] 接收:
68 59 14 D3 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 BC 5D 88 5D A7 5D 44 00 01
00 FE FF 7C 00 84 02 00
00 00 00 88 13 00 00 02 00 00 00 00 00 FE FF 00 00 00 00 B6 00 00 00 00 00 00 02 00 00 00 00 00
00
EB90EB90(端口号:21站5)[2020/1/6 23:09:06] 接收:
68 12 16 D3 00 00 0F 81 05 00 05 00 01 0C 00 B6 42 03 00 00

```

用正则：

```

^EB.+接收: \n(^.+$)
$1

```

把：

```

1 EB90EB90(端口号:21站5) [2020/1/6 18:19:07] 接收:
2 68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
3 EB90EB90(端口号:21站5) [2020/1/6 18:19:12] 接收:
4 68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
5 EB90EB90(端口号:21站5) [2020/1/6 18:19:17] 接收:
6 68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01 00
7 EB90EB90(端口号:21站5) [2020/1/6 18:19:22] 接收:
8 68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
9 EB90EB90(端口号:21站5) [2020/1/6 18:19:27] 接收:
10 68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
11 EB90EB90(端口号:21站5) [2020/1/6 18:19:32] 接收:
12 68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F2 5C 45 00 01 00
13 EB90EB90(端口号:21站5) [2020/1/6 18:19:37] 接收:
14 68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
15 EB90EB90(端口号:21站5) [2020/1/6 18:19:42] 接收:
16 68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
17 EB90EB90(端口号:21站5) [2020/1/6 18:19:47] 接收:
18 68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E8 5C 45 00 01 00
19 EB90EB90(端口号:21站5) [2020/1/6 18:19:52] 接收:
20 68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
21 EB90EB90(端口号:21站5) [2020/1/6 18:19:57] 接收:
22 68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
23 EB90EB90(端口号:21站5) [2020/1/6 18:21:57] 接收:
24 68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
25 EB90EB90(端口号:21站5) [2020/1/6 18:22:02] 接收:
26 68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 79 5D 5A 5D 6C 5D 45 00 01 00
27 EB90EB90(端口号:21站5) [2020/1/6 18:22:07] 接收:
28 68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00

```

替换成：

```

1 68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
2 68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
3 68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01 00
4 68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
5 68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
6 68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F2 5C 45 00 01 00
7 68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
8 68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
9 68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E8 5C 45 00 01 00
10 68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
11 68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
12 68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
13 68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 79 5D 5A 5D 6C 5D 45 00 01 00
14 68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
15 68 14 30 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
16 68 59 32 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 60 5D 6F 5D 44 00 01 00
17 68 12 34 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
18 68 14 36 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
19 68 59 38 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 67 5D 71 5D 45 00 01 00
20 68 14 72 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
21 68 59 74 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 70 5D 4E 5D 4F 5D 44 00 01 00
22 68 12 76 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
23 68 14 78 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
24 68 59 7A B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 6B 5D 47 5D 61 5D 44 00 01 00
25 68 12 7C B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
26 68 59 EA D2 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 B9 5D 7E 5D 9D 5D 45 00 01 00
27 68 12 EC D2 00 00 0F 81 05 00 05 00 01 0C 00 B5 42 03 00 00
28 68 14 EE D2 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01

```

得到每一行的数字：

```
68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01
00 FE FF 7C 00 94 02 00
00 00 00 86 13 00 00 02 00 00 00 00 00 FE FF 00 00 00 00 00 87 00 00 00 00 00 00 00 02 00 00 00 00 00
00
.
.
.
68 12 16 D3 00 00 0F 81 05 00 05 00 01 0C 00 B6 42 03 00 00
```

(3) 再去变成java字符串数组，即给每一行加上前后双引号

用正则：

```
^(.+)$
"$$1$$",
```

把：

The screenshot shows the VSCode interface with the following details:

- File Explorer:** Shows 'Analysis104Test.java' and 'SmartElectric\_104\_sample\_data.txt'.
- Search Bar:** Contains the regular expression '^(.+)\$'.
- Replace Bar:** Contains the replacement string '\"\$1\"'.
- Code Editor:** Displays a list of lines, each starting with a number (1-28) followed by a hex value. The hex values are enclosed in double quotes, indicating they have been converted into strings.

```

1 68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
2 68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
3 68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C DF 5C 45 00 01 00
4 68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
5 68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
6 68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F2 5C 45 00 01 00
7 68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
8 68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
9 68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E8 5C 45 00 01 00
10 68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
11 68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
12 68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
13 68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 79 5D 5A 5D 6C 5D 45 00 01 00
14 68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
15 68 14 30 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
16 68 59 32 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 60 5D 6F 5D 44 00 01 00
17 68 12 34 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
18 68 14 36 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
19 68 59 38 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 67 5D 71 5D 45 00 01 00
20 68 14 72 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
21 68 59 74 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 70 5D 4E 5D 4F 5D 44 00 01 00
22 68 12 76 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
23 68 14 78 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01
24 68 59 7A B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 6B 5D 47 5D 61 5D 44 00 01 00
25 68 12 7C B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00
26 68 59 EA D2 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 B9 5D 7E 5D 9D 5D 45 00 01 00
27 68 12 EC D2 00 00 0F 81 05 00 05 00 01 0C 00 B5 42 03 00 00
28 68 14 EE D2 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01

```

变成：

```

1 "68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

2 "68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

3 "68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 EC 5C CB 5C D

4 "68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

5 "68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

6 "68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F

7 "68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

8 "68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

9 "68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E

10 "68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

11 "68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

12 "68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

13 "68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 79 5D 5A 5D 6

14 "68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

15 "68 14 30 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

16 "68 59 32 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 7E 5D 60 5D 6

17 "68 12 34 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

18 "68 14 36 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

19 "68 59 38 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 7E 5D 67 5D 7

20 "68 14 72 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

21 "68 59 74 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 70 5D 4E 5D 4

22 "68 12 76 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

23 "68 14 78 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

24 "68 59 7A B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 68 5D 47 5D 6

25 "68 12 7C B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

26 "68 59 EA D2 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 B9 5D 7E 5D 9

27 "68 12 EC D2 00 00 0F 81 05 00 05 00 01 0C 00 B5 42 03 00 00 ",

28 "68 14 EE D2 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 "

```

```

"68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

"68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 01 00 00 00 01 ",

"68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 00 EC 5C CB 5

C DF 5C 45 00 01 00 FE FF 7C 00 94 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0 00 00 00 00 00 00 00 00 86 13 00 00 02 00 00 00 00 00 FE FF 00 00 00 00 87 00 00 00 00

0 02 00 00 00 00 00 00 ",

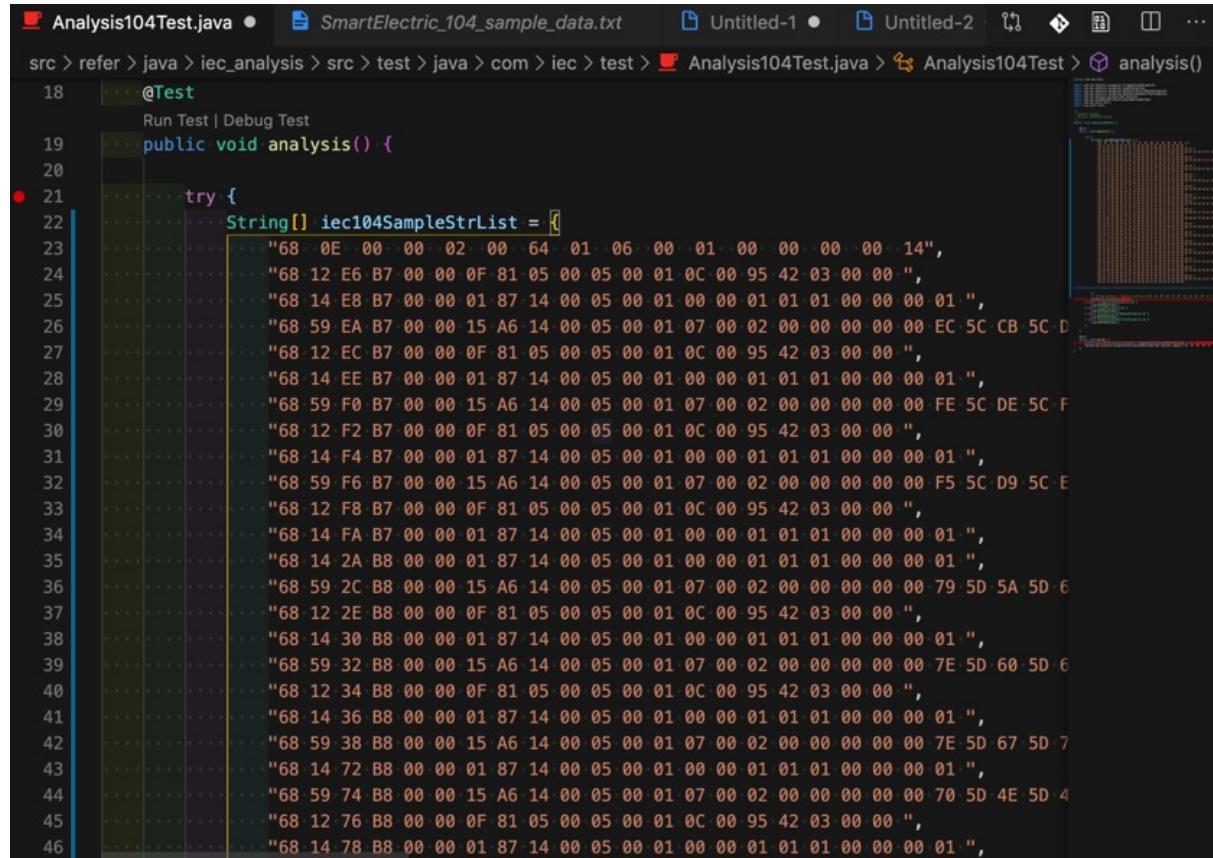
"68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00 ",

...

"68 12 16 D3 00 00 0F 81 05 00 05 00 01 0C 00 B6 42 03 00 00 "

```

用于粘贴到代码中使用：



```

18 @Test
19 Run Test | Debug Test
20
21 public void analysis() {
22
23 try {
24 String[] iec104SampleStrList = [
25 "68 0E 00 00 02 00 64 01 06 00 01 00 00 00 00 00 14",
26 "68 12 E6 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
27 "68 14 E8 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
28 "68 59 EA B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 EC 5C CB 5C D",
29 "68 12 EC B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
30 "68 14 EE B7 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
31 "68 59 F0 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 FE 5C DE 5C F",
32 "68 12 F2 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
33 "68 14 F4 B7 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
34 "68 59 F6 B7 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 F5 5C D9 5C E",
35 "68 12 F8 B7 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
36 "68 14 FA B7 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
37 "68 14 2A B8 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
38 "68 59 2C B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 79 5D 5A 5D 6",
39 "68 12 2E B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
40 "68 14 30 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
41 "68 59 32 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 60 5D 6",
42 "68 12 34 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
43 "68 14 36 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
44 "68 59 38 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 7E 5D 67 5D 7",
45 "68 14 72 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01",
46 "68 59 74 B8 00 00 15 A6 14 00 05 00 01 07 00 02 00 00 00 00 70 5D 4E 5D 4",
47 "68 12 76 B8 00 00 0F 81 05 00 05 00 01 0C 00 95 42 03 00 00",
48 "68 14 78 B8 00 00 01 87 14 00 05 00 01 00 00 01 01 00 00 00 01"
]
 }
}

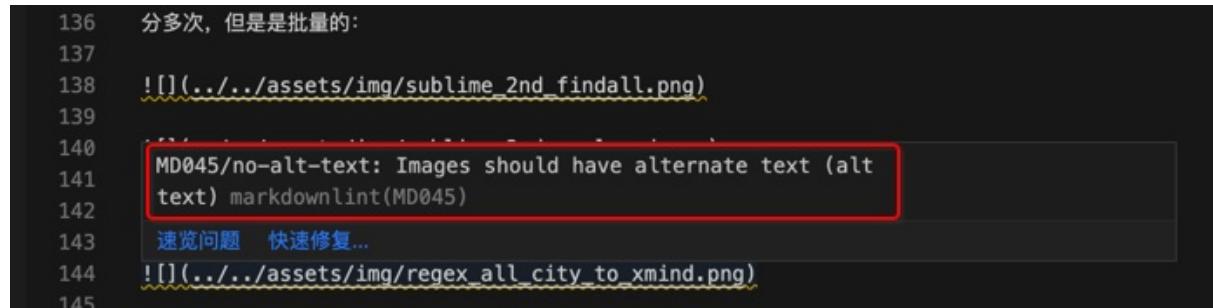
```

->从而把：

- 繁琐的，手工的，从原始文件中拷贝和粘贴的重复劳动，
- 快捷的，自动的，完成，且更准确，不会出现手动操作的失误。

## 填充Markdown中图片文件名

此处正在写教程期间，正好有个需求：为了避免和消除Markdown中的，关于图片的文件名即image的alt的text是空的警告：



```

136 分多次，但是是批量的:
137
138
139
140 MD045/no-alt-text: Images should have alternate text (alt
141 text) markdownlint(MD045)
142
143 速览问题 快速修复...
144
145

```

然后正好用正则，去自动填充此处image的alt的text，即图片的文件名

用正则：

```

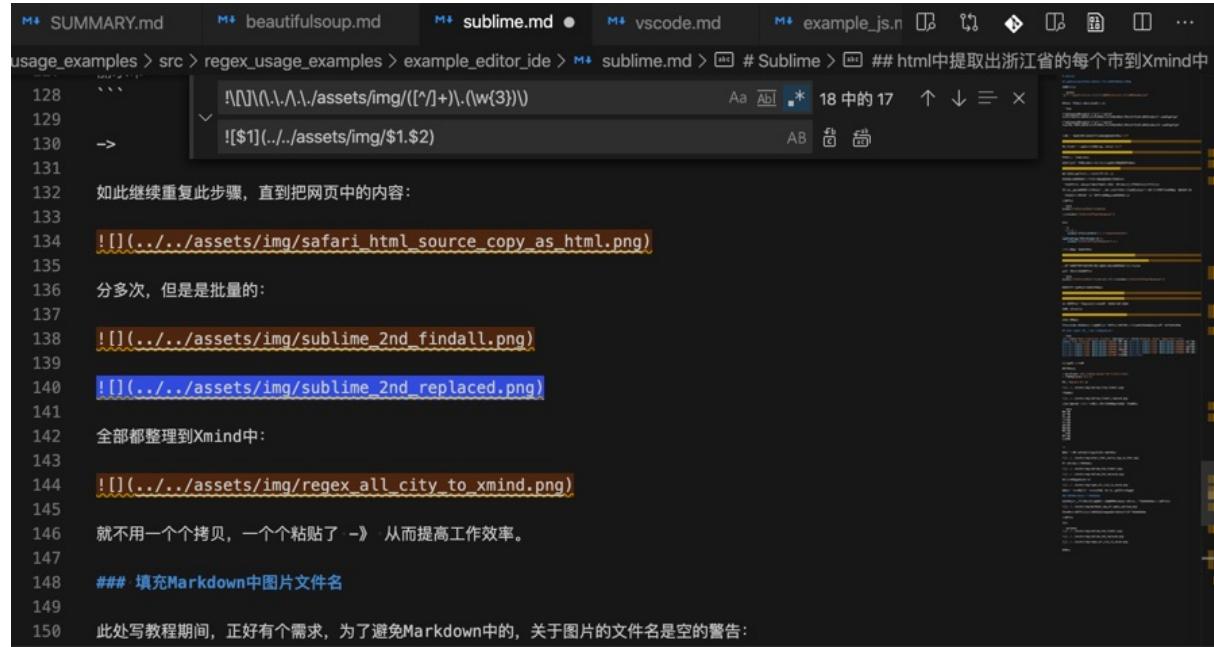
\[\]\(\.\.\./assets/img/([^\]+)\.(.\w{3})\)
[$1](../../assets/img/$1.$2)

```

把：

```
[sublime_2nd_findall](../../assets/img/sublime_2nd_findall.png)
[sublime_2nd_replaced](../../assets/img/sublime_2nd_replaced.png)

[regex_all_city_to_xmind](../../assets/img/regex_all_city_to_xmind.png)
```



The screenshot shows the VSCode interface with the file `sublime.md` open. The code editor displays several regex patterns:

- `\.(w{3})\!`
- `![\$1]\.\./assets/img/\$1.\$2`
- ``
- ``
- ``
- ``

The code editor has a yellow background for the selected text. The status bar at the bottom indicates the current file is `sublime.md`.

变成：

```
[sublime_2nd_findall](../../assets/img/sublime_2nd_findall.png)
[sublime_2nd_replaced](../../assets/img/sublime_2nd_replaced.png)

[regex_all_city_to_xmind](../../assets/img/regex_all_city_to_xmind.png)
```

```
127 丽水市
128 ```
129 !\[\\(\\.|\\.\\|\\.\\./assets/img/([^\"]+).\\(\\w\\{3\\})\\)\\]
130 ![$1](../../assets/img/$1.$2)
131
132 如此继续重复此步骤，直到把网页中的内容：
133
134 !\[safari_html_source_copy_as_html\](../../assets/img/safari_html_source_copy_as_html.png)
135
136 分多次，但是是批量的：
137
138 !\[sublime_2nd.findall\](../../assets/img/sublime_2nd.findall.png)
139
140 !\[sublime_2nd_replaced\](../../assets/img/sublime_2nd_replaced.png)
141
142 全部都整理到Xmind中：
143
144 !\[regex_all_city_to_xmind\](../../assets/img/regex_all_city_to_xmind.png)
145
146 就不用一个个拷贝，一个个粘贴了 -》 从而提高工作效率。
147
148 ### 填充Markdown中图片文件名
149
```

即可自动填充image的alt的text，消除Markdown中的警告了。

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：2020-02-02 22:54:11

## 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: 2020-02-02 18:46:07

## 参考资料

- [【教程】BeautifulSoup中使用正则表达式去搜索多种可能的关键字 – 在路上](#)
- [【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](#)
- [【整理】中国常见的城市的名字](#)

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: 2020-02-02 22:49:25