目录

学习方法思路及技术心得总结

- 最新版本：`v1.0`
- 更新时间：`20200625`

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。

简介

整理crifan在学习思路、学习方面方面的经验总结，包括学会利用搜索引擎，尤其是Google；以及用Google之前要先学会科学上网，以及善于利用网络上的维基百科，要学好英文才有利于深入学计算机技术，以及如何学习新技术给出思路和方法以及具体案例；以及在技术方面的感悟、思考和心得，比如技术开发和厨师做饭的类比，C语言指针变量和房子钥匙的类比，代码的味道，专业知识需要深入学习才能专精，技术层级对比，帮别人就是帮自己，做事情要有好习惯和好逻辑。以此希望能给需要的人以参考和帮助。

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/learn_tech_method_experience: 学习方法思路及技术心得总结

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

- 学习方法思路及技术心得总结 book.crifan.com
- 学习方法思路及技术心得总结 crifan.github.io

离线下载阅读

- 学习方法思路及技术心得总结 PDF
- 学习方法思路及技术心得总结 ePub
- 学习方法思路及技术心得总结 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。

学习方法和思路

在折腾了各种技术，包括不同计算机语言、平台、系统等，总结了一些学习方法和思路，总结如下，供参考。

搜索引擎

学习知识，要充分利用现有资源，尤其是网络资源。

网络资源中，最有价值的，最容易被大众所接触到的是：搜索引擎。

通过搜索引擎，找到你要的知识。

对于搜索引擎，个人的评价是：

- google：
  - 主页：http://www.google.com
  - 评价：搜索出来内容相关度最高，质量最好，算10分
  - 额外说明：需要科学上网后才能访问
- 百度：
  - 主页：https://www.baidu.com/
  - 评价：搜出来内容相关度一般
    - 之前：多数是广告
    - 现在：
      - 部分是广告
      - 剩下部分内容
        - 从内容可行度和可参考价值来说，算是：凑合用
- 其他
  - Bing=必应：
    - 主页：https://cn.bing.com/
      - 注：输入 https://www.bing.com/，会自动跳转到 https://cn.bing.com/
    - 评价：有国内版和国际版。
      - 个人很少用到
      - 总体算是还行吧，算是可以备用

结论：推荐用 **google**

Google

如何利用google搜索

用google搜索到你要的资料

核心要点是：搞清楚要搜索的**关键字**

如何知道要搜索什么呢?

需要较好的抽象思维和抓住问题要点，最终找到 问题核心关键词

下面就以举例方式来解释：

如何从问题入手，找到自己要搜的关键词

举例：html5的主页

比如：

【已解决】用什么技术实现公司官网首页H5页面

期间，对于想要用html5实现普通公司的主页，涉及到哪些技术和方法

但是除了知道应该搜索html5之外，不知道google中应该搜索什么关键词才比较合适。

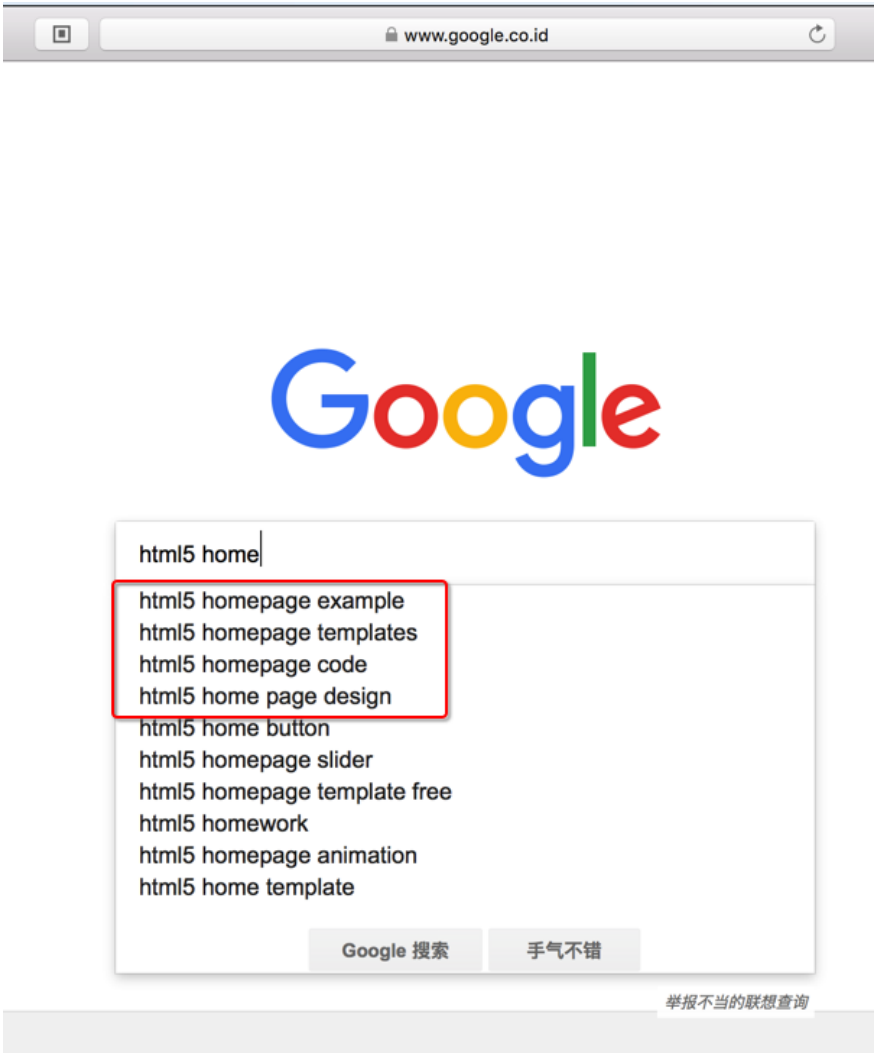
然后此处就正好利用了google的动态匹配，达到了自己的目的

才刚搜了

```
html5 home
```

之后，google就动态匹配的列出来，我希望查询的关键字，尤其是列表开始的几个：

- html5 homepage example
- html5 homepage templates
- html5 homepage code
- html5 homepage design

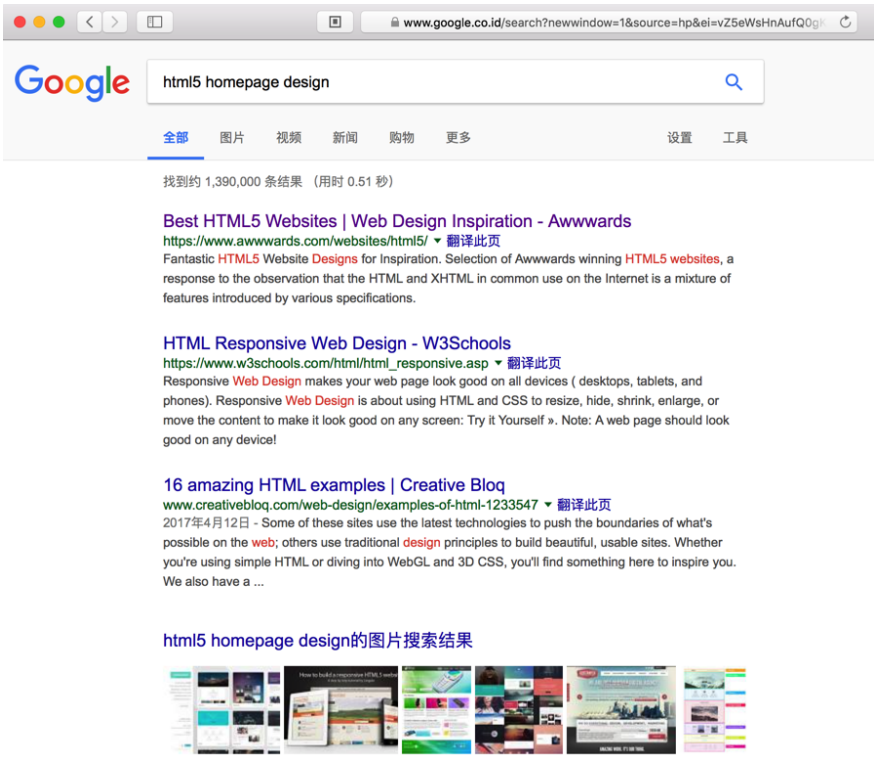


然后接着就可以分别去搜这些关键字，从而最终找到自己想要找的内容了

比如搜：

```
html5 homepage design
```

找到相关的技术内容：



举例：pyecharts的wordcloud的add的函数定义

比如在回答

关于Python pyecharts 的问题（已经找资料找了半天了）-CSDN论坛

的问题期间，对于别人代码：

```
wordcloud.add("",name,values,word_size_range=[20,100],shape= "circle")
```

需要去找到 `wordcloud.add` 的函数的定义

此处已知是Python的pyecharts的库

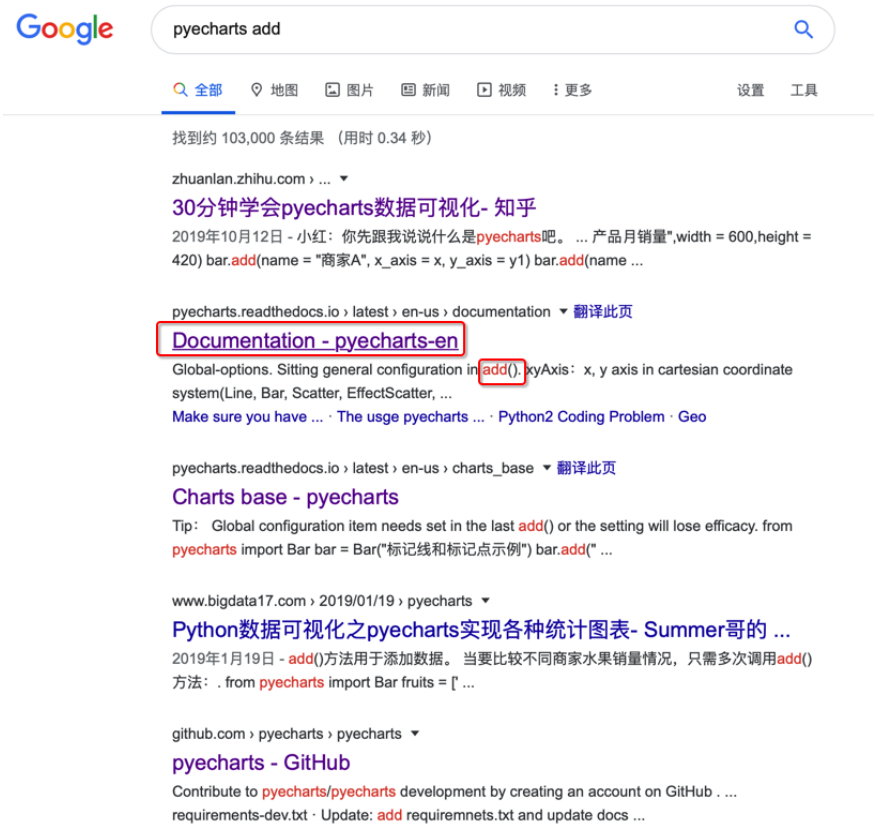
所以去搜：

`pyecharts add`

或：

`python pyecharts add`

找到：



可见，第二个，一眼就能看出是**官网资料**

因为其网址是：

`pyecharts.readthedocs.io › latest › en-us › documentation`

这种地址

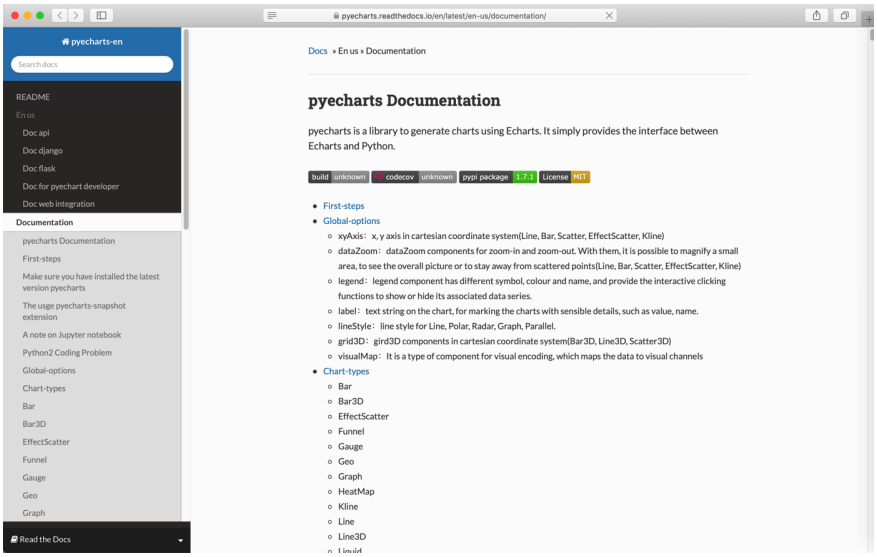
而（你需要具有，或逐渐熟悉）已知知识：

`readthedocs.io` 是个存放Python的库的官网文档的一个网站

-》从而就能确定：这个资料，肯定是官网资料（之一）了。

进入后是：

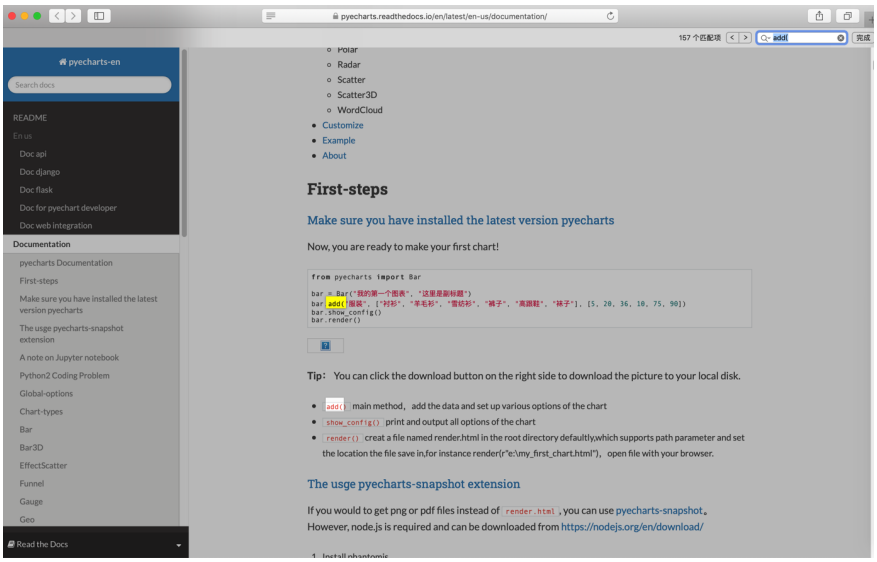
Documentation - pyecharts-en



然后尝试在页面中直接搜：

`add(`

也的确会找到，很多个 `add()`



但好像不是我们要的那个add函数的定义

-》至此，也还是不能一下子找到我们要的add函数的定义

但是能注意到：好像是pyecharts中，具体还分很多个功能模块

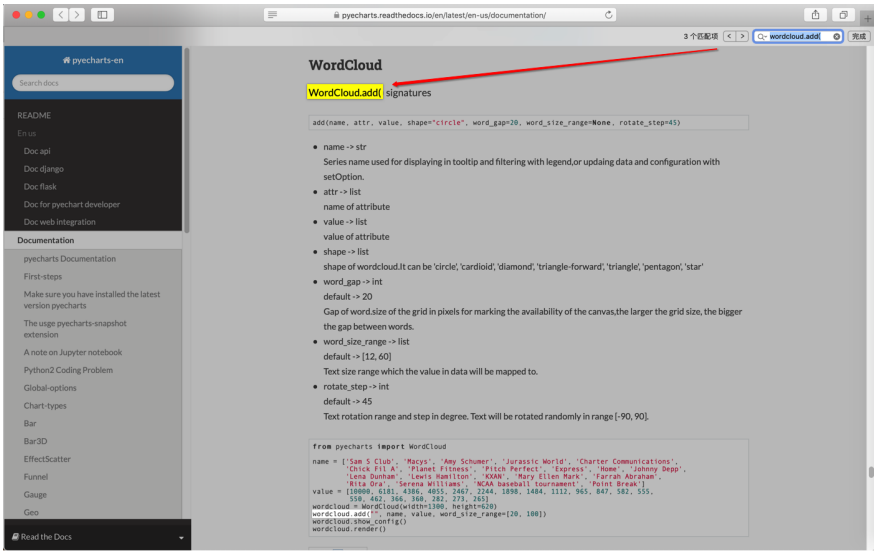
因为原问题中指的是wordcloud

然后此处就去搜索

`wordcloud.add(`

注：万一还有 其他xxxcloud.add，那么也可以去搜： `cloud.add(` ，多找找看，也是可以找到此处的 `wordcloud.add(` 的。

即可找到我们要的：



看到函数定义 `WordCloud.add() signatures` 是：

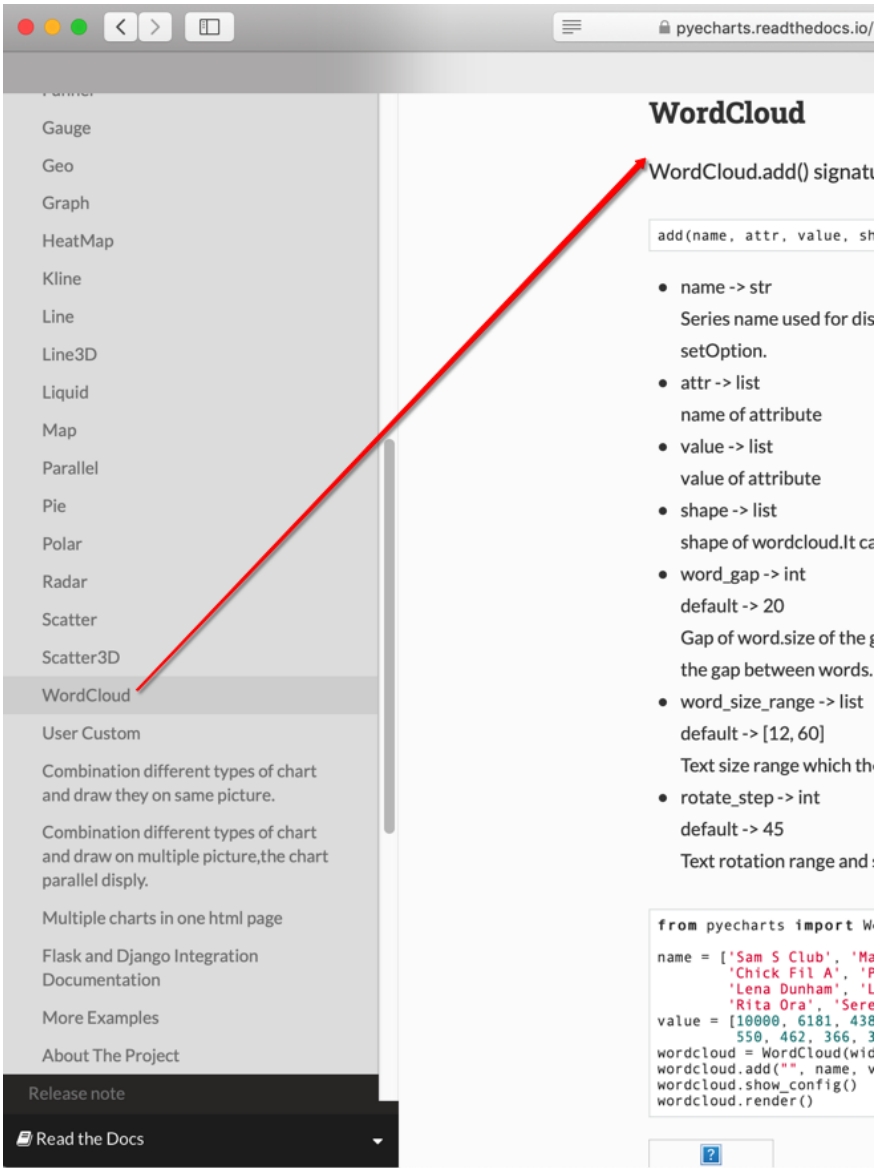
```
add(name, attr, value, shape="circle", word_gap=20, word_size_range=None, rota
```

至此，完成目标任务：

找到此处 `pyecharts` 的(`wordcloud` 的) `add` 函数的定义

-》从而搞清楚：每个参数含义，以及如何传递参数

另外也注意到，左边的目录中有这个 `WordCloud`



-》验证了之前的判断：`pyecharts`中有多个模块，`wordcloud`只是其中一个模块的推断。

Google高级用法

关于google搜索技巧，想说的是：自己这么多年利用Google找资料，其实很少用到高级用法。

毕竟核心在于前面提到的，把你遇到的问题，提炼出准确的关键词，然后即可快速找到你要的内容。

当然，懂更多的搜索技巧，肯定是好事，能灵活运用，才能锦上添花。

下面总结一些Google高级的搜索技巧，供参考：

- 想要精确匹配，则用**引号**
  - "你要搜索的内容"
- 不想要搜索结果包含某些东西，用：**减号** `-`
  - "你要搜索的内容" - "要排除的内容"

- 使用**通配符**
  - **星号** `*`
    - 举例：
      - `"the most * examples of censorship"`
        - 结果包含：
          - `"the most outrageous examples of censorship"`
          - `"the most dangerous examples of censorship"`
          - `"the most prolific examples of censorship"`
          - 等等
  - **波浪号** `~`
    - 举例：
      - `the importance of ~censorship`
        - 等价于搜索：
          - `the importance of censorship`
          - 同时搜索censorship的相关词汇，比如：propaganda：
          - `the importance of propaganda`
- 站内搜索=指定只搜索某个网站：用 `site:`
  - 举例：
    - `the purpose of education site:http://www.time.com/`
      - 只搜索网站 `http://www.time.com/` 中的 `the purpose of education`

科学上网

但是google在国内无法访问。想要用google，必须先学会科学上网。

关于如何科学上网，可以参考我的另外独立教程：

科学上网相关知识总结

维基百科

去学习一个，对于自己来说是全新的未知的新知识，其中利用网络资源的一个很有效的入门手段是：`维基百科`

- 维基百科
  - 英文：`Wikipedia`
  - 简称：`维基` = `wiki`
  - 主页：
    - https://en.wikipedia.org/
  - 是什么：一个百科全书式的网站
  - 做什么：对世界上的各种事物进行解释
    - 尤其是计算机技术，解释的透彻和明白
      - 对于搞技术的，往往都会用得到
  - 特点
    - 能把东西解释的清楚
      - 对于任何一个词条，往有概述，历史，版本，背景，特点等等非常全面的解释
    - 全球所有人都可以编辑和更新
      - 促进不断改进
        - 越加准确和符合实际
    - 有多种语言
      - 英文版
        - 很多技术和事物都是说英文国家发明的，源于英文系的国家
          - 所以
            - 英文版内容往往是最准确和最全面的
            - 且往往也是英文版解释的最清晰和易懂
      - 中文版
        - 虽然中国人更习惯看（简体）中文（或繁体中文）
          - 但是中文版很多词条的内容不是很全
            - 不过和百度百科等竞品比，依旧有很大的参考价值
  - 适用于
    - 尤其是对一个技术名词不是很了解的情况下，去wiki，可以快速了解其所属技术领域和该技术的概况
  - 相关
    - 国内有个百度百科
      - 其中有些（或者说很多）内容是参考（拷贝）了维基百科的中文版

举例：如何有效利用wikipedia

我之前是做嵌入式领域的技术，但是也没听过`现场总线`

所以对于想要 从无到有 从完全陌生到希望知道，对于现场总线， 其大概是个什么东西，大概是用来干啥的，希望有个基本的概念上的了解

所以去google搜：

现场总线

可以找到 wiki的解释：

作为中国人，可以先看看中文的解释：

现场总线 - 维基百科，自由的百科全书

> 现场总线是许多工业用通讯协定的总称，一般用在即时分散式控制系统，IEC 61158是有关现场总线的标准，不过也有一些现场总线未列在IEC 61158中，如Modbus、LonWorks、CANopen等。
>
> 一个复杂的自动化系统（例如组装生产线）会需要一个有组织的控制系统阶层才能运作。在此阶层的顶端一般是人机界面（Human Machine Interface, HMI），可以让操作员监控及使用此系统。中间层则由许多可编程逻辑控制器（PLC）组成，PLC之间借由网络系统（如Ethernet）传递资料。低层则是由现場总线连接PLC及感测器、致动器、马达、开关、阀门、接触器等实际动作或侦测的元件。

但是其实解释的不够透彻，还是看英文原版：

Fieldbus - Wikipedia

> Fieldbus is the name of a family of industrial computer network protocols used for real-time distributed control, standardized as IEC 61158.
>
> A complex automated industrial system — such as manufacturing assembly line — usually needs a distributed control system—an organized hierarchy of controller systems—to function. In this hierarchy, there is usually a Human Machine Interface (HMI) at the top, where an operator can monitor or operate the system. This is typically linked to a middle layer of programmable logic controllers (PLC) via a non-time-critical communications system (e.g. Ethernet). At the bottom of the control chain is the fieldbus that links the PLCs to the components that actually do the work, such as sensors, actuators, electric motors, console lights, switches, valves and contactors.

会更加清楚:

- 首先 现场总线，只是一个 name名字 而已
  - -》 意思是 不是某种具体的技术和东西
- 但是是 一些东西的 总称
  - 具体包含哪些东西呢?
    - 包含的是：一组的 工业用的（industrial） 通讯协议 （computer network protocols ）
      - 与工业用相对应的：可能其他领域就是 民用的？军用的?
  - 用于什么场景或领域呢?
    - 用于实时的 分布式控制系统
      - 那肯定就有：
        - 与实时的相对应的：非实时的领域/场景
        - 与分布式相对应的：比如 集中式的系统
      - 虽然上述与此对应的领域，我们更不熟悉
      - 但是至少知道 现场总线 是 知道其侧重点是针对于实时的 分布式的

    - 而这类协议综合起来，形成了一个国际标准：IEC 61158

至此，基本上就大概了解了 现场总线 大概就是：

- 是一堆协议的总称
- 这堆协议：
    - 是工业用的 通讯协议
    - 侧重于规范 实时的 分布式的
    - 综合起来对应着国际标准：`IEC 61158`

由此实现了：

从无到有 对 现场总线 有了基本的了解。

后续教程

后来的后来，在对现场总线有了稍微多点的了解后，又去整理出了系列内容，需要的可以去参考：

现场总线Fieldbus简析

英文

作为中国人，虽然生活中没必要一定用中文。

但是计算机领域的开发，不论从写代码，还是看资料，往往离不开英文。

且越是对技术深入研究和追根溯源，往往也是需要看英文类的资料。

所以：

- 想要学好计算机技术的中国人
  - 如果英文很好：那么恭喜你
    - 可以发挥你的优势
  - 如果英文不好：
    - 请努力去学好英文
      - 没有其他捷径可走
      - 这样才能看懂英文材料，搞透技术
        - 真正成为技术大牛

如何学习新知识

学习新知识的方法

遇到一个新知识，要先把几个几个核心要点问题搞清楚：

- `what`：搞清楚一个东西**是什么**
- `why` 和 `when`
  - `why`：一个东西**为何**出现
    - 往往涉及到该东西出现的背景和缘由
  - `when`：一个东西**何时**出现
    - 从出现的时间范围，往往也有助于理解其前后因果关系
- `how`：**如何**实现这个东西，具体是怎么做的，大致原理

把这几个问题都解释清楚了，也就对一个东西有了大致的了解了。

学习新知识的步骤

核心思路：找到最权威的官网的资料，去参考官网教程去学习

比如：

- `scrapy`
- `Django`
- `ReactJS`
  - 官网教程的众多好处：
    - 权威，够准确
      - 内容解释更加到位
    - 内容最及时
      - 尤其是React等发展比较快的技术，最新的文档往往最有价值

举例：获取openpyxl当前excel的sheet的最大的行row数

在用Python的openpyxl去处理excel文件：

【已解决】python解析excel文件并读取其中的sheet和row和column的值

期间，需要找到openpyxl中，如何获得当前excel的sheet的最大的行（row）数

根据经验，确信官网文档：

openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files — openpyxl 2.5.0 documentation

中是可以找到的

但是看了很多个具体的页面的example例子中：

Simple usage — openpyxl 2.5.0 documentation

Manipulating a workbook in memory — openpyxl 2.5.0 documentation

并没有找到。

思路：按照道理，这个最大的行数，应该属于当前的（excel的）sheet的，所以感觉应该属于sheet类中的

（背景知识：一般官网文档，除了turorial教程，example示例等，都还有API接口文档的）

所以就去找api文档：

openpyxl package — openpyxl 2.5.0 documentation

中搜sheet方面的类。只有一个：

openpyxl.comments.comment_sheet module

明显不是我要的。

然后继续利用google去搜：

Openpyxl get current sheet max row

而找到了：

python - Is it possible to get an Excel document's row count without loading the entire document into memory? - Stack Overflow

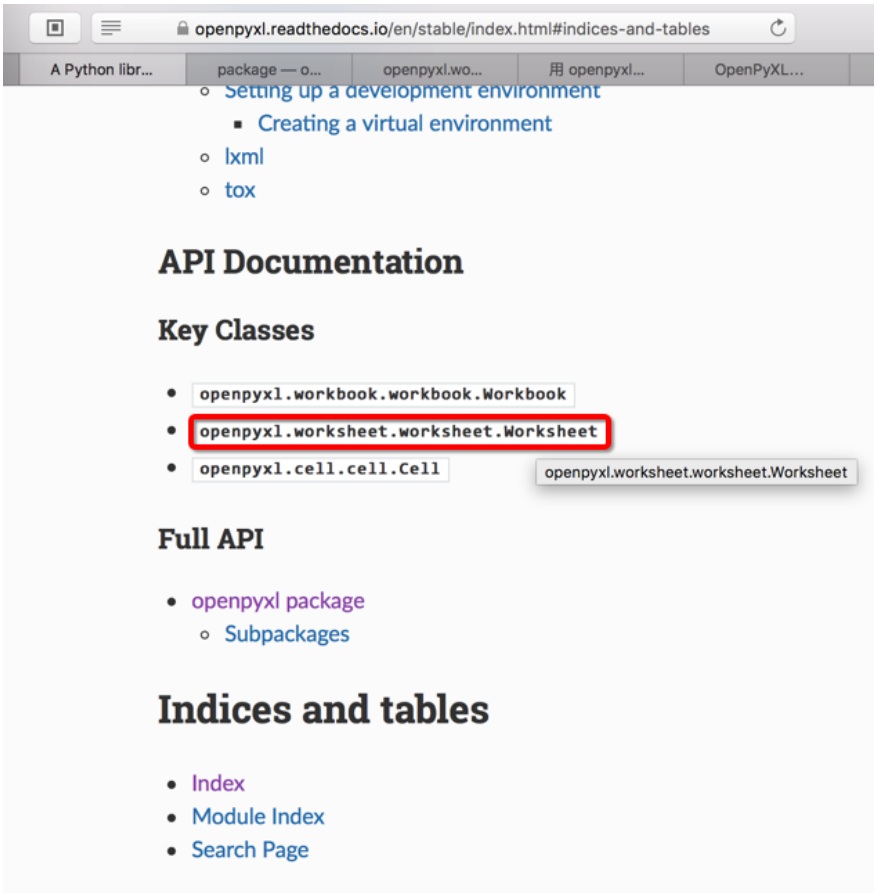
中的，我们要的：

```
    row_count = sheet.max_row
    column_count = sheet.max_column
```

而反过来回到官网文档，突然才发现，原来官网文档首页：

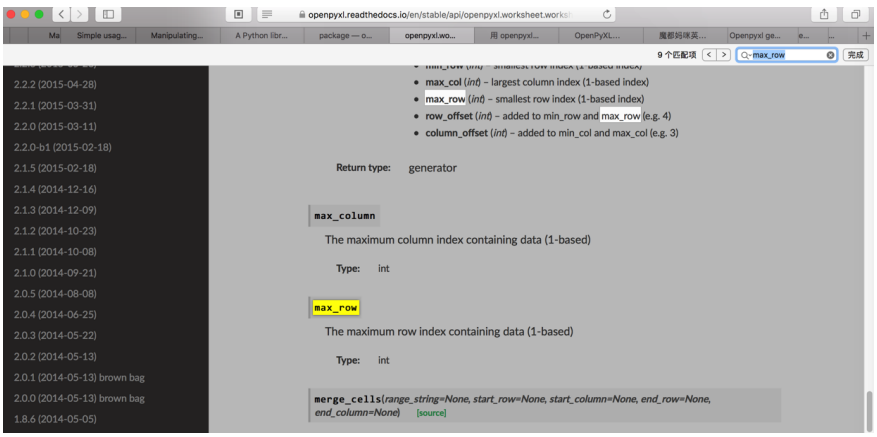
openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files — openpyxl 2.5.0 documentation

是有对应的类：



`openpyxl.worksheet.worksheet.Worksheet`

进去 openpyxl.worksheet.worksheet module — openpyxl 2.5.0 documentation 后果然就可以搜到 `max_row` 的：



所以此处总结过程就是：

1. 首先要求自己：充分利用官网文档和Google搜索，可以帮忙我们快速精准的找到要的东西
2. 其次要求对方：官网文档首先要写的更清楚，准确，方便查找
3. 然后要求自己：看官网文档要足够细心，才能看准，看清，找到自己要的东西

而所有这一切，要有一些背景知识和经验：

1. 要知道很多好的库，往往文档也写的很好
    - 所谓文档写得好，至少意味着：

        - 文档和代码是同步的，不是滞后的
        - 文档要有清晰的简明的示例，即example，tutorial等内容
            - 用于给新手快速上手，了解如何使用
        - 也有相关的API接口文档
            - 当用户需要深入了解具体某个类有哪些功能、参数、属性、注意事项等，可以去查API文档
2. 要知道作为一般的技术人员的你，针对尤其是成熟的第三方的软件、系统、库等等，所遇到的绝大多数问题，往往别人都遇到过，往往别人都上网去讨论过，往往都有热心人回答过，而你要做事情只是：
    - 去充分利用google等去搜索到自己需要的内容
        - 当然如何去选择搜索的关键词，也是很重要的
            - 即使你不太会把当前问题提取出合适的关键词，往往google的动态匹配，给你合适的提示
        - 和google比，其他搜索引擎，比如百度，搜狗，bing等，对于搜索出的帖子的质量（相关程度，帖子回答的质等），都没google好，所以有条件的，尽量用google

技术心得

此处整理一些自己的技术感悟，技术心得，技术方面的总结。

技术感悟

技术开发 vs 厨师做法

技术开发 就像 厨师做饭：

解决一个问题，实现一个功能，可能有多种方案，需要技术人员根据经验和能力，做个权衡和取舍，且十八般武艺要都略懂，且部分技术要精通，好在需要的时候，选择适合的工具去组合和搭配实现自己要的东西。

就像厨师做饭，各种工具，比如电饭锅，高压锅，微波炉，电烤箱，各种尺寸的锅，油盐酱醋，等等，都要懂，且有些工具精通优缺点在哪里，这样才能用不同工具组合搭配，做出自己希望的美味。

C语言：指针和变量

C语言的**指针**和**实体**之间的关系。就像**钥匙**和**房子本身**之间的关系。

且对应的变量，比如**结构体有几个值**，或者实体占用空间的大小几个字节？就像普通的房子的内部结构占的大小。比如**两房或三房**，更大的话那就快变成别墅了。

普通使用者想要使用指针，就像在一个小区里很多房子不知道要去哪里。拿了指针就像知道了你的房间的地址，获得楼栋号码，就可以拿着钥匙去找到你要的房间地址了。

不同领域知识需要大量精力才能专而精

不同行业，领域内有专门的协议，如果想要研究透，需要花不少精力的

比如：

- 软件框架或协议
  - `jabber`
    - 邮件服务器领域：
      - jabber协议概述 - codefans - BlogJava
  - `XMPP`
    - 即时通讯=聊天 系统
- 硬件协议
  - `USB` 协议
  - `蓝牙` 协议

等等。

如果只是了解个概念，那很快的，可以速成。

但是想要深入了解和研究具体实现细节，就需要专门从事该领域很长时间，才可能达到专业和精通的水平的。

代码写的不好-》有难闻的味道

比如：

Killing Switch Statements in React with the Strategy Pattern

> I'm of the opinion that the presence of a switch statement, or an if-else, is a very pungent code smell

提到的：

> Code smell, also known as bad smell, in computer programming code, refers to any symptom in the source code of a program that possibly indicates a deeper problem

如果代码写的不好，作为有经验的人员，能闻出**代码**中不好的**味道**。

技术层级类比

关于 学历等级 和 IT技术层级 类比关系：

目的：给对于IT技术不熟悉，没概念的人，解释不同技术领域的划分和层级之间的关系。

- 说明
  - L=Level
    - 举例
      - L1=Leve 1=层级1
    - 用途：
      - 用于 某个技术面或技术点的研究领域 和 对应学历的分层 去做对比
- 类比
  - 学历等级：由低级到高级
    - L1：小学
      - L2：中学
        - L3：大学
          - L4：硕士
            - L5：博士
  - IT技术层级：由较广的技术范围到较细分的研究领域
    - L1：硬件
    - L1：软件
      - L2：移动端
      - L2：web前端
      - L2：后端
        - L3：Java后端
        - L3：PHP后端
        - L3：Python后端
          - L4：Python的Web框架
            - L5：Flask
            - L5：Django
          - L4：Python部署工具
            - L5：supervisor
            - L5：gunicorn
          - L4：Python虚拟环境
            - L5：pipenv
          - L4：Python操作数据库
            - L5：Python中操作MySQL
            - L5：Python中操作MongoDB
- 举例解释：
  - IT领域的L5的Flask 类比到学历是 L5的博士
    - 就相当于：某个学历是L5的博士，其研究领域的范围只是此处的：
      - IT中的软件的后端的Python后端的Web框架的Flask
    - 特点：
      - 研究领域明显很窄

        - 但研究的深度很深
            - 而具体的某个研究的细分方向，可能就是博士的研究论文课题
                - 比如
                    - 如何实现Flask的高并发
                    - Flask的项目代码结构最佳实践
    - 结果
        - 从博士精通此领域，对该领域内的问题比较有话语权
        - 但是跨了领域的其他学科和领域（比如硬件领域），其实并不一定比普通人知道的多多少
            - 比如 硬件领域的初中生，就比此Flask领域的博士，对于硬件更了解

帮别人就是帮自己

给别人方便就是给自己方便 -》 东西设计的好用已用，才容易被别人使用，才容易变得流行

典型的有：

- 计算机语言
  - Python语言
    - 设计的很好用，已用，所以加上人工智能这一波的助理，变得用途更广更流行
  - Swift语言
    - 本身设计的也很不错，所以流行度也不错
- 深度学习
  - cuDNN
    - NVIDIA的cuDNN，之所以能流行，估计和这句
      > It allows them to focus on training neural networks and developing software applications rather than spending time on low-level GPU performance tuning
    - 很有关系，即：
      - 帮你省时间进行了底层优化了
        - -》就不用使用者去操心底层的事情，使用者就只需要关系自己的上层的神经网络的训练即可
          - -〉由此就是：
            - NVIDIA帮别人忙 -》设计了自己的GPU硬件的上层软件库，利于别人使用
              - 就是帮自己的忙-》使得自己的NVIDIA显卡GPU硬件卖的好，成了神经网络算法加速的最佳选择

总的说就是：

在计算机领域，不论是软件还是硬件，都遵循一个普适的原理：

帮别人就是帮自己

-》 能多为别人（你的语言/软件/代码/程序/硬件/库）多考虑，让别人少操心，省事

最终就是帮自己-》你的代码/语言/库/硬件等等，就被别人用的更多/流行度更高/使用更广泛/卖的更好

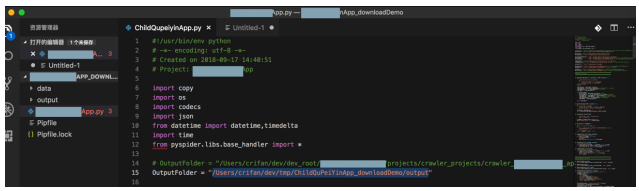
好习惯好逻辑

做事情要养成好的习惯和逻辑

下面举例说明：**什么叫好的习惯和逻辑**?

之前写了个爬虫，去爬取数据

其中需要保存爬取的数据到本地电脑中

- 好的习惯是：
  - 在第一次写代码时，就对于要保存的数据的路径，去提取出一个：
    - 全局的根目录
      - 后续保存数据都是放在根目录下的子目录
  - 这样做的好处是：
    - 万一，如果，后面的变更了要保存数据的路径
      - 只要改变一处即可，不用多个保存的路径都去修改
  - -> 随着后续开发，真的遇到这种需求：
    - 真的是只需要用同样代码，但是只是改动了爬取数据的保存路径即可。
      - 所以此时只需，不改动其他代码，而只是改变根目录：
        - 
      - 就能实现需求了

附录

下面列出相关参考资料。

参考资料

- 【已解决】用什么技术实现公司官网首页H5页面
-
- 【整理】学会google搜索的高级技巧
- 【已解决】python解析excel文件并读取其中的sheet和row和column的值
- 现场总线Fieldbus简析
- 【crifan推荐】利用搜索引擎google帮助你解决问题 – 在路上
-
- Fieldbus - Wikipedia
- 现场总线 - 维基百科，自由的百科全书
-