简单粗暴 LaTeX

dk-huapen

本手册是自己学习 laTeX 的排版材料

最后更新于：2025 年 4 月 23 日

目录 | 0

序 | 1

为啥会有这个玩意儿呢（暂时还不能称它为一本书），就是因为爱折腾，这么能折腾不记录下是不是会有点遗憾呢？最开始这些记录是在我自己的那个网站里边记录的，但是使用的时候感觉不太方便也不太美观，直到遇见了它，它很对我的脾气，有点爱不释手，最后决定边学习边记录，说不定最后还真能写出点东西来，期待吧...

现在的书大部分是系统的专业的讲解一些东西，我个人感觉在现在这个信息化发达的环境中，学习资料是不缺乏的，反而因为学习资料太多对我这种泥腿子造成的困惑更大，我更需要的是一个方向和一个一个有效的小目标，可事实确实要么在原地打转重复的学着相同的内容，要么步子跨的太大学者不感兴趣莫名其妙的东西直到放弃。id 其实在之前我是有一稿手册的，开始撰写的日期大概在 2015 年 4 月，但是自己觉得写的太烂，因此索性推倒重写了这一版. 这一版的主要特征是:

1. 我希望能够吸引初学者快速上手，解决手头的问题. 因此去掉了枯燥的讲解和无穷无尽的宏包用法介绍，直接使用实例;
2. 力求突出实用性. 当然，也会提点一些可以深入学习的内容，读者可以自行查阅，或者阅读本手册中的扩展阅读章节（即带星号 * 的章节）.
3. 本手册使用的编辑器为 TEX Studio，而非之前的商业软件 WinEdt. 这使得学习 LATEX 的门槛更低. 当然了，你有权使用任何编辑器.

主体分为六大部分[1;2]:

写给读者 *
基础
数学排版
进阶
绘图 *
附录

由于工作全部由我一人完成，限于视野，难免存在错漏之处。恳请读者指正。如遇到的手册中无法解决的问题，欢迎向我提出。推荐书目可参考本手册附录。最后，还要感谢在 LaTeX 学习中为我解答疑惑的同学，特别是来自 LaTeX 度吧的吧友；本手册中许多的解决方案都是由他们提供的。我谨在此记录。

Mail:wklchris@botmail.com
Chris Wu
September 17，2016 于 Davis，CA

更新日志：版本号以更早的版本的更新细节，请到。。浏览。

2 写给自己和有缘人*

2.1 写给自己的话

感觉好多时候都在原地打转，重复做着一些事情，如果从这里开始可以记录下来作为查阅资料的话应该会好很多吧，那就从它开始吧。

2.2 TEX 与 LATEX 的优缺点

TEX 的优点：

2.3 为什么需要 LATEX ？

你可能基于以下原因学习

2.4 MS Word 难道不优秀吗？

我想说的是

2.5 LaTeX 生成的文件格式？

一般广为使用的是 pdf

操作系统 | 3

本章主要记录我在使用各款操作系统过程中值得记录的一些东西，包括安装各款操作系统时遇到的问题以及使用各款操作系统的体验，最重要的是记录一些常用的、零碎的命令或知识点方便以后查阅。

3.1 操作系统选择

操作系统还用选择？买上电脑用就完了，有啥好选择的。在我初次接触 linux 系统前真的没有想过这个问题，记得那是大专一年级 2018 年的时候，学习 C 语言课程后开始发现了自己有点喜欢编成，然后开始学习 MFC 写一些 Windows 下的程序，那会儿用的还是 Vis
操作系统就是计算机上的工作平台，目前主流的就是 Windows、macOS、Linux，Linux 系统呢又有很多版本，比如 Unbont、CentOS、Debian。

3.1.1 计算机磁盘规划

这里主要记录下计算机硬盘的选择和配置要求，个人计算机呢用的时间长了会卡经常会重装系统，针对这个怎样配置硬盘最方便呢？服务器要求是稳定运行，所以硬盘会采用磁盘阵列做冗余配置，那应该选择哪一种呢？

3.1.1.1 个人计算机磁盘规划

个人电脑的话我感觉使用两块硬盘，一块 SSD 固态硬盘用作安装操作系统和常用软件方便整套系统备份恢复，一块使用 HDD 机械硬盘主要用来存放数据文件方便数据备份和迁移。下面记录下关于 MBR 和 GPT 分区以及 Logic 和 UEFI 引导的记录。

MBR 和 GPT 最明显的区被就两点：

1. GPT 是在计算机发展过程中发现 MBR 不够用才出来的一种分区方式
2. MBR 最大只支持 2TB 硬盘，如果硬盘大于 2TB 只能用 GPT
3. MBR 支持分区数量有限，GPT 理论上无限个可以
4. MBR 支持 Logic 引导系统，GPT 支持 UEFI 引导系统
5. MBR 启动分区需要标记，GPT 启动分区默认 C 盘
6. win7 及以前系统用 MBR，win10 及以后系统用 GPT

这个是在一次安装 win10 电脑上安装 win7 过程中遇到情况，安装完 win7 后启动不了找不到系统盘最后猜测是原来年 win10 是 GPT 分区的 UEFI 引导启动，后来我把 win7 装到 GPT 分区后仍热无法引导启动查资料说 win7 本来不支持 UEFI 引导，需要做一些工作才能实现 UEFI 引导，这个后续有机会的话试一下。

3.1.1.2 服务器磁盘规划

工业用电脑或服务器的话还是配置 RAID 1 比较实用，毕竟备份恢复起来太方便了。这里我简单记录下我接触过的两种型号服务器配置方法，以及 RAID 故障恢复的方法。

首先是 DCS 用的 HP Gen8 服务器，磁盘阵列控制器型号是 HP Smart Array P420i, 硬盘是刀片式插槽，现场配置的是 RAID 1，在做硬盘备份恢复过程中用到的。

磁盘阵列配置

1. 将硬盘插入插槽，开机，看服务器硬件情况大概得 5 分钟以上

2. 当自检界面出现 Press <F5> to run the Option ROM Configuration for Arrays Utility 时按下 <F8> 进入磁盘阵列配置界面
3. 选择 View Logical Drive 选项查看当前阵列配置情况，如果是 RAID 1, 就会显示状态是 OK/RECOVERY，还会显示实际硬盘安装情况 OK/MISSING
4. 选择 Delete Logical Drive 可以删除当前阵列配置
5. 选择 Create Logical Drive 创建阵列，创建 RAID 0 的话就直接选择单硬盘就可以，如果是 RAID 1 的话就可以选择两块甚至更多硬盘组成镜像盘
6. Select Boot Volume 是设置系统启动硬盘，服务器里启动选项可以选择磁盘阵列卡

RAID 1 磁盘阵列备份硬盘

1. 服务器配置了 RAID 1 模式并指定了冗余硬盘盘位为前提，下边提到的插拔硬盘都是在 RAID 1 配置盘位上进行的
2. 操作系统正常运行过程一组硬盘中硬盘的禁止弹出指示灯不亮说明冗余正常拔出任意一块硬盘都不会影响当前系统正常运行，拔出其中一块硬盘后剩下的一块硬盘禁止弹出指示灯会点亮说明此时该硬盘拔出会影响正常运行
3. 插入一块空硬盘，重点是空硬盘，清除完 RAID 标记后的没有建分区表的空硬盘，此时新插入硬盘灯亮开始同步数据，这时原硬盘禁止弹出指示灯仍然点亮，说明备份没有完成 RAID 1 没有建立冗余
4. 视硬盘数据大小同步时间会有差别，大概 1 小时左右，同步完成后硬盘的禁止弹出指示灯都会熄灭，此时备份完成可以任意拔出其中一块硬盘作为备份盘或 RAID 1 冗余运行

还使用过一种惠普塔式工作站，主板上磁盘阵列卡是定制的默认配置 RAID 1，直接就可以硬盘同步备份。

另一种服务器是搭建 SIS 系统时用的戴尔服务器，这个使用过程中给我映像比较深的是更换备份硬盘后需要选择配置启动硬盘，以后再用到的时候记录下。

3.1.2 操作系统安装

最开始安装 XP 系统时常用的还是 CD，后来到了 Windows7 以后用的 U 盘安装，硬盘安装比较多一点，安装 Linux 系统的时候用的 U 盘安装比较多一点，后来也用过 U 盘引导硬盘安装的方法，下面就简单记录下感受吧。现在 ISO 安装光盘都很大了尤其是 Linux 的超过 4.7GB 了，而且现

在电脑光驱都不用了尤其是个人电脑，所以 CD 和 DVD 安装就不多说了。U 盘安装是把 ISO 文件用 U 盘启动制作工具刻录在 U 盘里，但是废 U 盘啊。还有一种就是把 U 盘制作成 PE 文件引导安装 Windows 系统，这种的广告插件是真的多啊，这种应该只适用于 Windowns 安装。使用 U 盘引导硬盘安装 Linux 系统还可以就是制作 U 盘引导有点费劲，但是上边的方法需要根据要安装系统的情况和需求进行选择，做成各种安装介质备起来或用的时候临时做，我之前就是这么作的。直到遇见它中级杀人武器积各种技能与一身的 Ventoy，用它制作好 U 盘后只需要把你的所有 ISO 镜像文件拷贝放到 U 盘里，U 盘启动后选择你要用的 ISO 文件就可以了。

3.1.3 Ventoy

用官网的话来说Ventoy是一个制作可启动 U 盘的开源工具，你只需要把 ISO 等类型文件直接拷贝到 U 盘里面就可以启动了，无需其他操作，可以一次性拷贝很多不同的镜像，它会形成一个菜单供用户选择，一个 U 盘可以同时支持 x86Legacy BIOS、UEFI 模式。通俗点讲就是系统镜像拷贝进取就可以使用，只要 U 盘够大可以把想安装的操作系统镜像都放到 U 盘里，随时可以安装任何一款操作系统。这里记录下 U 盘安装 Ventoy 和硬盘安装 Ventoy 方法。

3.1.3.1 U 盘安装 Ventoy

安装的过程没有什么复杂的现在新版本是图形化操作，选中 U 盘安装就可以了，值得一提的是 U 盘里边应该放哪些系统 ISO，全方？太大！不全方？用的时候又不方便！我使用过程中感觉有以下这么几种选择：

1. 用小点 U 盘专门安装 Ventoy 工具，再用移动硬盘存储系统镜像用哪个选择哪个
2. 用 U 盘专门安装 Ventoy 工具，再放入 Linux Live 系统和 winPE 系统用来处理问题，需要安装什么系统再找对应 ISO 文件

3.1.3.2 移动硬盘安装 Ventoy

3.1.4 WinPE

网上有很多选择比如老毛桃、大白菜、雨沐林枫等，我老毛桃用的比较多就以它为例吧，在 Windwos 系统下安装、恢复用。用官网的话来说 Laomaotao-winPE 是一款系统预安装环境 (PE) 支持 BIOS(Legacy) 与 UEFI 两种启动模式。我感觉它比较实用的是 PE 系统里边的一些工具比如磁盘分区、系统密码破解，最好用的应该是 Windows 系统无法启动时拷贝数据

和 GHOST 系统备份和恢复。这里主要记录下使用 WinPE 盘备份和恢复系统。

3.1.4.1 WinPE 备份和恢复系统

ABB DCS 系统操作员站 OP53 恢复过程记录

1. 首先准备 WinPE 系统盘、惠普 Z230 主机箱要求 C 盘符 80GB、op16920.GHO 备份镜像
2. 进入 WinPE 系统后将 op16920.GHO 文件拷贝至 D 盘，启动 Ghost 工具
3. 在 Ghost 软件中选择恢复系统选项，选择 op16920.GHO 文件，选择目标分区 C 盘符，开始恢复... 等待 10 分钟左右，恢复完成
4. 关机重启进入新恢复的系统，默认登录用户是 OP16, 切换为管理员修改 IP 地址为 172.16.48.153 和 172.17.48.153
5. 计算机名称为 OP53
6. 将计算机接入 DCS 系统网络后，登录管理员确认网络正常，添加或修改域名为 LuanPP.Local，重启计算机后 LUANPP.operator 用户登录，查看站点状态是否正常。
7. 修改域名需要计算机正常接入 DCS 系统户对应语

打开 TEX Studio 后，选择选项 → 设置 TEX Studio → 构建 → 默认编译器，选择 XELATEX。这主要是基于中文文档编译的考虑，同时 XELATEX 也能很好的编译英文文档。我建议始终使用它作为默认编译器。

3.1.5 Debian Live

这个是一个可以装到 U 盘里的 Debian 系统，很方便的，我目前只使用它清空过硬盘数据。这里简单记录下我们厂使用的是 ABB DCS 系统，它属于 C/S 架构，光服务器就有 16 台，后来准备搭建一套 ABB DCS 系统服务器来做培训和测试用，但是 ABB 这个授权很贵，而且厂家费用很高，就计划自己试着搭建以下，后来一看安装软件就 20 多 GB，而且特别多，估计安装难度很大，现场服务器硬盘配置是 RAID 1 冗余，所以就选择了硬盘备份这种方式，从外边采购相型号配置的旧服务器，旧的服务器硬盘都有数据，如果用 RAID 1 同步的话就需要清空旧硬盘中的数据，关键是 RAID 标识，我在网上查了下清空 RAID 标识方法总结起来有三种：1.linux 下用 dd 命令擦除硬盘最后 RAID 数据；2.linux 下用命令删除 RAID 标识；3.linux 下用 fdisk 删除硬盘分区和 RAID 标识。这三中方法都是在 linux 系统下，所以就用 U 盘作了个 Debian Live 系统，用第三中方法试了下，可行。下边就分别记录下三种使用方法的具体过程。

3.1.5.1 Debian Live 清除 RIAD 标识

之后你可以在窗口输入一篇小文档，并保存为 tex 扩展名的文件进行测试：

```
\documentclass{ctexart}
\begin{document}
Hello,world!
你好，世界！
\end{document}
```

点击编译按钮生成，F7 查看。生成 pdf 在你的 tex 文件保存目录中。具体各行的含义我们后在后文介绍。

3.2 Windows 操作系统

Windows 操作系统安装有两种：一种是用原版的操作系统安装介质，一种是用别人 GHOST 的备份来还原。前者是安装步骤较多，最后需要自己激活甚至安装驱动，但是干净，后者是安装方便，傻瓜式一键安装，但是系统有别人给预装的插件广告等（自己脑补吧），我喜欢后者，原因很简单干净、还算有点挑战性（大折腾吗）。以下是针对使用前者安装方式的 Windows 操作系统安装最重要的几点就是安装介质、系统激活和安装驱动（尤其是安装完没有网卡驱动的）。磁盘规划: 最好是一块固态硬盘直接安装操作系统和软件，一块机械硬盘按需求分区后存放数据，这样备份恢复系统，迁移数据最方便。

3.2.1 XP 操作系统

曾经神一样的存在，只是时代在进步，不论是从硬件还是软件上它都不足以支撑，不过工业上还有很多在用，因为工业软件需求单一，稳定为主没有更新，操作系统还使用经典的 XP 系统。我记得 2008 年时开始接触操作系统，后来就帮人装系统，那个时候就是 XP，后来不知道从哪里找到一个 XP 系统 ISO 安装文件 (Microsoft Windows XP Professional 版本 2002 Service Pack3)，然后又试出一个:CM3HY-26VYW-6JRYC-X66GX-JVY2D，可以一直重复使用的哦，值得提一句的是 XP 系统是安装过程中就要求输入授权码的，否则就不能继续往下走了。然后就一直使用它们，直到 2020 年上班后因为连接西门子 S7200PLC 时还在虚拟机上安装了 XP 系统专门用来上载、修改、下载 PLC 程序。

3.2.2 Win7/Win10 操作系统

Win7 和 Win10 安装方式基本一样，其实 Win7 及后来的操作系统安装方式基本没有什么变化，都是安装后进行注册激活。相比 XP 多了一种安装方式，硬盘安装，这种方式前提是安装操作系统的电脑有旧的有 windows 操作系统存在，在旧操作系统下将 ISO 镜像文件解压到 D 盘根目录下然后点击 SETUP 安装程序即可，简单吧。

下边就以记录下我最近安装 WIN10 系统的过程吧，这个 WIN10 安装介质是我们厂 SIS 系统安装时厂家在客户端安装的 WIN10(Windows 10 专业版) 系统时用的正版光盘，我做成了 ISO 文件。

1. 拷贝 WIN10.ISO 文件到安装 Ve 的 U 盘
2. 电脑 U 盘启动选择 WIN10,nomal install
3. 系统安装过程。。。
4. 安装完成后检查设备驱动正常，简直不要太顺

3.2.2.1 Win7 操作系统激活

Win7 系统激活是有一个激活程序 (PCSKYS_Windows7Loader_v3.27.exe)，安装万系统后，打开激活软件按照提示激活，如果激活失败需要到我的电脑-> 管理-> 磁盘管理里检查是不是有盘符是空白，给空白盘符添加驱动号后，再重新激活

3.2.2.2 激活 WIN10 系统

目前在 Windows 10 专业版测试是可以激活的

1. 进入 win10 系统桌面中，鼠标右键点击桌面左下角的 window 按钮，在弹出的菜单中选择“命令提示符 (管理员)”选
2. 在打开的命令符号符界面中输入 slmgr.vbs /upk 命令，将 win10 系统中原来的激活密钥卸载，进入下一步，弹出成功卸载了产品密匙
3. 输入 slmgr /ipk NPPR9-FWDCX-D2C8J-H872K-2YT43，弹出：”成功的安装了产品密钥”
4. 输入 slmgr /skms zh.us.to，弹出”密钥管理服务计算机名成功的设置为 zh.us.to”
5. 输入 slmgr /ato，弹出”成功的激活了产品”
6. 在系统属性界面中可以看到 win10 企业版的激活状态

LaTeX 中的**命令**通常是由一个反斜杠加上命令名称，再加上花括号内的参数构成的（有的命令不带参数，例如：`\TeX`）。

```
\documentclass{ctexart}
```

如果有一些选项是备选的，那么通常会在花括号前用方括号标出。比如：

```
\documentclass[a4paper]{ctexart}
```

还有一种重要指令叫做**环境**。它被定义与控制命令`\begin{environment}`和`\end{environment}`间的内容。比如：

```
\begin{document}
...内容...
\end{document}
```

环境如果有备选参数，只需要写在`\begin[...]{name}`这里就行。

注意：不带花括号的命令后面如果想打印空格，请加上一对内部为空的花括号再键入空格。否则空格会被忽略。例如：`\LaTex{} Studio`。

3.2.3 保留字符

LaTeX 中有许多字符有着特殊的含义，在你生成文档时不会直接打印。例如每个命令的第一个字符：反斜杠。单独输入一个反斜杠在你的行文中不会有任何帮助，甚至可能产生错误。LaTeX 中的保留字符有：

```
# $ % & _ { } \
```

它们的作用分别是：

#：自定义命令时，用于标明参数序号。

§：数学环境命令符。

以上除了反斜杠外，均能用前加反斜杠的形式输出。即你只需要键入：

```
\# \$ \% \^ \& \_ \{ \}
```

唯独反斜杠的输出比较头痛，你可以尝试：

```
$\backslash$ \textbackslash
\texttt{\char92}
```

\ \\

其中命令`\char[num]`是一个特殊的命令

```
\texttt{\char`~}%输出一个波浪线
```

3.2.4 导言区

任何一份 LaTeX 文档都应该包含以下结构：

```
\documentclass['\itshape options']{doc-class}%没有斜体option
\begin{document}
...
\end{document}
```

其中，在语句`\begin{document}`之前的内容成为**导言区**。导言区可以留空，也可以进行一些、文档的准备操作。你可以粗浅地理解为：导演区即模板定义。

文档类的参数 doc-class 和可选选项 *options* 有取值：

表 3.1: 文档类和选项

doc-class 文档类[1]	
article	- 科学期刊，演示文稿，短报告，邀请函。
proc	- 基于 article 的会议论文集。
report	- 多章节的长报告、博士论文、短篇书。
book	- 书籍。
slides	- 幻灯片，使用了大号 Scans Serif 字体。
options	
字体	- 默认 10pt，可选 11pt 和 12pt。

在本文中，多数的文档类提及的均为 report/book 类。如果有 article 类将会特别指明。其余的文档类不与说明。本手册排版即使用了 report 类。

在导言区最常见的是**宏包**的加载工作，命令形式如：`\usepackage{package}`。通俗地讲，宏包是指一系列已经制作好的功能“模块”，在你需要使用一些原生 LaTeX 不带有的功能时，只需要调用这些宏包就可以了。比如文本的代码就是利用`listings`宏包实现的。

宏包的具体使用将参在个部分内容说明中进行讲解。如果你想学习一个宏包的使用，按 Win+R 组合键呼出运行对话框，输入 texdoc 加上宏包名称即可打开宏包帮助 pdf 文档。例如：`texdoc xeCJK`。

[1] 此外还有`beamer`宏包定义的 beamer 文档类，常用于创建幻灯片。

3.2.5 错误的排查

在编辑器界面上，下方的日志是显示编译过程的地方。在你编译通过后，会出现这样的字样：/

- **Errors 错误**：严重的错误。一般地，编译若通过了，该项是零。
- **Warnings 警告**：一些不影响生成文档的瑕疵。
- **Bad Boxes 坏箱**[2]：指排版中出现的长度问题，比如长度超出（Overfull）等。后面的 Badness 表示错误的严重程度，程度越高数值越大。这类问题需要检查，排除 Badness 高的选项。

你可以向上翻越日志记录（即.log 文件），来找到 Warning 开头的记录，或者 Overfull/ Underfull 开头的记录。这些记录会指出你的问题出在哪一行（比如 line 1-2）或者在 pdf 的哪一页（比如 active[12]。注意，这个 12 表示计数器技术页码，而不是文件打印出来的真实页数）。此外你还需要了解：/

- 值得指出的是，由于 LaTeX 的编译原理（第一次生成 aux 文件，第二次再引用它），目录想要合理显示**需要连续编译两次**。在连续编译两此后，你会发现一些 Warnings 会在第二次编译后消失。在 TeX Studio 中，你可以只单击一次“构建并查看”，他会检测到文章的变化并自动决定是否需要编译两次。
- 对于大型文档，寻找行号十分痛苦。你需要学会合理地拆分 tex 文件，参阅内容。

这里也推荐宏包`syntonly`，在导言区加入它支持的`\syntaxonly`命令，会只排查语法错误而不生成任何文档，这可以使你更快地编译。不过他似乎不太稳定，例如本文档可以正常编译，但是使用该命令时则会出错。

此处注解在后续章节，目前未链接

3.2.6 文件输出

LaTeX 的输出一般推荐 pdf 格式，有 LaTeX 直接生成 dvi 的方法并不推荐。

你在 tex 文档的文件夹下可能看到的其他文件类型：

.sty	宏包文件
.cls	文档类文件。
.aux	用于存储交叉引用信息的文件。
.out	宏包生成的 pdf 书签记录。

[2] Box 是 LaTeX 中的一个特殊概念，具体将在进行讲解。

3.3 UNIX 操作系统

UNIX 系统种类很多啊，尤其是近几年，越来越火，记得我是 2009 年第一次接触 Ubuntu，那时候只是好奇，为了好玩，去电脑商城买电脑的时候非要选预装 Ubuntu 系统的笔记本，别人看我的眼神是那样的，最后转了一圈也没找下，不过最后找下了一款 Nseries 标志的笔记本电脑，就是现在用的 DELL vostro 1088，到现在已经 15 年了，记得当时买回去第一件事情就是重装 Ubuntu 系统，当时纯属装 C，不过也多亏了那股劲才坚持到了现在，中间学习操作系统的时候还安装了 minix 操作系统，后来上班后真正搭建服务器时用上了 CentOS，CentOS7 后就停止维护了寻找替代系统过程中试着用 Debian，这会儿已经到了 Debian12 版本了，不得不说 Debian 真是稳定啊，占用资源少，果断把 15 年的笔记本换成 Debian，健步如飞啊。就剩 Fedora 了，有机会了试试。

3.3.1 minix 操作系统

英文单引号并不使用两个'符号组合。左单引号是重音符`（键盘上数字 1 左侧），而右单引号是常用的引号符。英文中，左双引号就是连续两个重音符号。英文下的引号嵌套需要借助`\thinspace`命令分隔，比如：

```
%方括号里的是啥意思?
``\thinspace`Max' is here.''
```

“‘Max’ is here.”

中文下的单引号和双引号你可以用中文输入法直接输入。

3.3.2 Linux 操作系统

接触 Linux 操作系统很早大约是在 10 年，那个时候我记得用的还是 CentOS5，那会主要是接触了一本名叫一个 Orange 操作系统的书籍就迷上了操作系统的实现，这本书是在 linux 操作系统下使用 Boch 虚拟机来测试调试实现的迷你操作系统，中间会用到许多修改查看二进制数据的小工具，这些工具 linux 系统都自带不需要安装很方便，所以就开始使用 Linux 操作系统了，不过那会儿的 Linux 操作系统体验很差，只有在写操作系统的时候才使用，后来慢慢有所改善，一直到 CentOS8, 后来 CentOS 系统停止发布，又改为了 CentOS Stream，感觉不好就开始寻找其它的 Linux 操作系统，最后选择了 Debian，使用后感觉 Debian 是真稳定啊。linux 系统下边需要特别说的两点分别是硬盘规划和软件源配置。英文的短横分为三种：/

- 连字符：输入一个短横：-，效果如 daughter-in-law

- 数字起止符：输入两个短横：--，效果如：page 1–2
- 破折号：输入三个短横：---，效果如：Listen—I'm serious.

中文的破折号你也许可以直接使用日常的输入方式。中文的省略号同样。但是注意，英文的省略号使用`\ldots`这个命令而不是三个句点。

3.3.3 Debian 操作系统

使用 Debian 系统比较晚大约是在 2023 年，当时最新版本是 Debian11 代号 Bullseye，这部分内容更新于 2025 年 2 月份，最新版本 Debian 系统为 12 代号 Bookworm，Debian11 和 Debian12 的区别是 Debian11 拥有大量成熟的软件包适合在服务器和生产环境中使用，Debian12 功能更新颖，软件版本更新对部分软件支持不是很好，更适合尝鲜和体验。所以当时我就在现场使用的笔记本上安装了 Debian11，在个人笔记本上安装了 Debian12，这样两者就都能兼顾了。下面以 Debian11 和 12 系统安装为例子简单记录下，系统安装完成后需要优化的一些操作。需要特别注意的是安装 Debian11 过程中先不要配置网络参数，否则安装过程特被漫长，感觉它在网络上下载镜像源而不是使用本地 ISO 镜像源，安装过程中有个选项是不使用网络镜像源，但是还是特别慢，如果不配置网络参数的话也就 40 分钟就安装完成了，Debian12 安装不会有这种现象，只用选着不使用网络镜像源就可以了。

1. 配置镜像源，系统安装完成 apt 源默认配置的是本地光盘镜像，网络镜像源速度比较慢，更新为国内镜像源,Debian12 源地址，Debian11 源地址。

```
su#登陆root权限
cd /etc/apt/#换到源地址配置目录
move sources.list sources.list.back#备份原来的源文件
#用指定源替代sources.list文件中的源
apt-get update#更新源
```

2. 将用户添加到 sudo 组中，将当前用户添加到 sudo 组中，这样在执行需要 root 权限时临时使用 sudo 命令即可，直接登陆 root 用户比较危险哦！将用户添加到 sudo 组后需要重启电脑后方可完全生效，这一点不是很明白，理论上应该不需要重启才对。

```
su#登陆root权限，这个时候还是需要的哦
apt-get install sudo#安装sudo软件
/usr/sbin/usermod -a -G sudo $(echo $USER)
#将当前用户添加至sudo组
#以后使用sudo输入用户密码就可以临时切至root权限了
```

3. 将/usr/sbin 添加至环境变量，不然/usr/sbin 下命令使用时还需要使用路径，比较麻烦

```
sudo vim /etc/profile#需要root权限才能修改该文件
#在最后一行加入
export PATH=$PATH:/usr/sbin#$将/usr/sbin加入PATH
#这样该目录下程序就可以直接执行了，比如上例中的usermod命令
#如果新安装软件需要的话也可以加进去，比如我们的\LaTeX
#保存退出后
source /etc/profile#重新加载配置文件使修改生效
```

4. 设置 Debian 启动后默认运行级别，如果是个人 PC 电脑的话默认的 graphical.target 就可以，如果是服务器的话就需要设置成 multi-user.target 级别（服务器一般不设置桌面环境，浪费资源而且桌面不稳定因素较多影响服务器稳定性）。

```
systemctl get-default#查看当前运行级别
graphical.target#图形话桌面
systemctl list-units --type=target
#查看可供替换的运行级别
systemctl set-default multi-user.target(修改为多用户文本)
startx#多用户文本级别下启动图形桌面
```

5. Debian 系统桌面环境安装与切换，一般安装是就已经默认安装号了桌面环境和启动配置，如果使用过程中想要切换桌面的画可以参考这里。这里主要记录下服务器设置为多用户文本启动后，startx 默认启动的桌面环境。

```
update-alternatives --config
    x-session-manager#设置默认桌面
```

LaTeX 中专门有个叫做`\emph{text}`的命令，可以强调文本。对于通常的西文文本，上述命令的作用就是斜体。如果你对一段已经这样转换为斜体的文本再使用这个命令，它就会取消斜体，而成为正体。

西文中一般采用上述的斜体强调方式而不是粗体，例如在说明书的时候可能就会使用以上命令。关于字体更多内容参考字体这一节。

3.4 常用命令

LaTeX 原生提供的`\underline`命令简直烂的可以，建议你使用`ulem`宏包下的 `uline` 命令代替，它还支持换行文本。`ulem`宏包还提供了一些实

用命令：这里记录一些常用命令的常用格式，便于忘记时查阅。

3.4.1 ls 命令

ls 列出文件，这里主要记录 ls 的通配符用法
systemctl COMMAND server

```
ls *.tex#列出以tex结尾的文件
ls *.[!t]??#列出倒数第三个字符不是t的文件
```

3.4.2 systemctl 命令

systemctl 检查和控制系统与服务管理的状态
systemctl COMMAND server

```
systemctl start apache2#启动apache2服务
systemctl status apache2#检查apache2服务状态
systemctl stop apache2#停止apache2服务
systemctl enable apache2#设置apache2服务开机自启动
systemctl disable apache2#禁止apache2服务开机自启动
```

3.4.3 ssh 命令

OpenSSH SSH 客户端（远程登录程序）
ssh [-p port] user@hostname
-p port 指定远程主机端口

```
ssh debian_ibm@dklovelich.iok.la -p 18210
    #通过外网使用域名登录远程debian系统
ssh debian_ibm@192.168.1.16 -p 22
    #通过局域网使用IP地址登录debian系统
```

3.4.4 scp 命令

安全复制（远程文件复制程序，用法和 cp 命令相似）
scp [user@host1:]file1 [user@host2:]file2
-r 递归复制整个目录
-P port 是指定数据传输用到的端口号，默认 22 端口

```
scp -P 18210 file1
    debian_ibm@dklovelich.iok.la:/home/debian_ibm/dk/
    #从本地将文件file1传输到服务器/home/debian_ibm/dk/目录下
scp -Pr 18210 dir1
    debian_ibm@dklovelich.iok.la:/home/debian_ibm/dk/
    #从本地将文件夹dir1传输到服务器/home/debian_ibm/dk/目录下
scp -P 18210
    debian_ibm@dklovelich.iok.la:/home/debian_ibm/dk/file1 ./
    #将服务器上的file1文件传输到本地./目录下
scp -Pr 18210 debian_ibm@dklovelich.iok.la:/home/debian_ibm/dk
    ./ #将服务器上的dk文件夹传输到本地./目录下
```

3.4.5 screen 命令

screen 是一个多任务窗口管理器
screen [-ls]|[-S sessionname]|[-r sessionname]
-ls 列出所有会话
-S 指定会话名称
-r 恢复会话
-d 断开指定的会话，但不会杀死会话中的任务

```
screen -ls #查看当前正在运行的说有会话
screen -S log #启动一个名为log的会话
screen -r log #恢复一个已经脱离的名为log的会话
screen -d log #断开log的会话，但不会杀死会话中的任务
```

3.4.6 rar 命令

rar 是解压缩 rar 压缩文件的命令
rar [-ls]|[-S sessionname]|[-r sessionname]
-ls 列出所有会话
-S 指定会话名称
-r 恢复会话

```
rar x file.rar#解压缩file.rar文件到当前目录
```

3.4.7 du 命令

du 是 linux 系统里的文件大小查看的命令
du [-ls]|[-S sessionname]|[-r sessionname]

-s 查看文件夹总大小
-h 智能显示文件大小
-r 恢复会话

```
du -sh dk#查看dk文件夹总大小
```

3.4.8 grep 命令

grep 是 linux 系统里的搜索命令
grep [-rn] char dir
-r 当前路径下循环搜索
-n 结果输出显示行号
-l 只显示文件名，不显示匹配的文本
-I 忽略匹配二进制文件

```
grep -rn  topnav2.php ./
#递归查找当前目录下所有包含topnav2.php字符串的文件
```

3.4.9 sed 命令

du 是 linux 系统里流编辑器，用于批量修改文件内容
du [-nefri][动作]
-i 直接修改读取的文件内容
s 替换动作

```
sed -i "s/topnav2.php/topnav.php/g" grep "topnav2.php" -rIl ./
#替换当前目录下除了二进制文件的所有文件中topnav2.php字符串为topnav.php
#grep -I参数很有必要，在git目录下会避免修改git仓库内二进制文件
#否则损坏git仓库索引导致git不可使用
```

4 软件安装配置

这一章主要记录使用过的一些值得记录的软件的安装、配置和使用。

4.1 常用的软件

第一款软件无疑就是 VIM 了，那第二款肯定是 Latex 了，这玩意就是用他两鼓捣出来的。编辑器的配置大概是需要讲解一下的，毕竟对于初学者来说是很头疼的事情。本手册就以 TEX studio 为例进行配置。首先你应该安装一个 TEX Live，他是完全免费的，网址：http://tug.org/texlive/。

虽然它体积较大，但是却是最一劳永逸、最不需要花时间去配置的方法，同时它大概也是功能支持最强的 LATEX 发行版。

打开 TEX Studio 后，选择选项 → 设置 TEX Studio → 构建 → 默认编译器，选择 XELATEX。这主要是基于中文文档编译的考虑，同时 XELATEX 也能很好的编译英文文档。我建议始终使用它作为默认编译器。

之后你可以在窗口输入一篇小文档，并保存为 tex 扩展名的文件进行测试：

```
1 \documentclass{ctexart}
2 \begin{document}
3 Hello,world!
4 你好，世界!
5 \end{document}
```

点击编译按钮生成，F7 查看。生成 pdf 在你的 tex 文件保存目录中。具体各行的含义我们后在后文介绍。

4.2 VIM

vim 堪称上古神器，第一小节我们给这款神奇再叠加 buff。

4.2.1 vim-plug

刚安装的 VIM 功能都比较朴实，但是 VIM 的扩展性很强可以安装各种各样的插件，vim-plug 是一款 VIM 的插件管理工具，有了它可以很方便的安装、调用、卸载各种插件，项目网址：https://github.com/junegunn/vim-plug。

下载 plug.vim 文件至 /.vim/autoload/下，同时创建 /.vim/plugged 目录用来放置将来要安装的插件，配置 /.vimrc 文件并添加调用插件模块，Debian12 用户根目录下没有 vimrc 文件，复制/usr/share/vim/vim90/目录下 vimrc_example.vim 文件即可。

还有一个关于 Vim 插件的网站是VimAwesome，上边有几乎所有的 Vim 插件安装使用方法，其中有一篇叫vim-plug 的文章是一位大神 Vim 的配置文件，可以很容易配置一款强大的 Vim。

```
ls ~/ #~目录
.vim
#用户目录下没有vimrc文件
cp /usr/share/vim/vim90/vimrc_example.vim ~/.vimrc
ls ~/ #~目录
.vim .vimrc
ls ~/.vim
autoload plugged
ls ~/.vim/autoload
plug.vim#plug.vim下载原文件

vim ~/.vimrc #在.vimrc文件尾部添加下面内容
..........
call plug#begin('~/.vim/plugged')
#Plug 要安装的插件
call plug#end()

:PlugStatus#重新打开vim后在命令模式下使用该命令查看插件安装情况

:PlugInstall#在vim命令模式下安装插件

:PlugClean#卸载插件，需要先在vimrc配置文件中删除对应插件配置信息
:PlugUpgrade#更新vim-plug插件自身
```

4.2.1.1 NERDTree

NERDTree 是 Vim 编辑器的文件系统资源管理器。

```
vim ~/.vimrc #在.vimrc文件尾部添加下面内容
.........
call plug#begin('~/.vim/plugged')
Plug 'preservim/nerdtree'#加载nerdtree插件
call plug#end()
:source %#重新加载.vimrc配置文件
:PlugInstall#在vim命令模式下安装插件
:NERDTree#启动NERDTree插件
```

4.2.1.2 markdown-preview

markdown-preview 插件可以让 vim 在编辑 markdown 文件时在浏览器下同步滚动展示效果，是同步哦!

```
vim ~/.vimrc #在.vimrc文件尾部添加下面内容
.........
call plug#begin('~/.vim/plugged')
Plug 'iamcco/markdown-preview.nvim'#加载插件
call plug#end()
:source %#重新加载.vimrc配置文件
:PlugInstall#安装插件
:call mkdp#util#install()#安装支持软件
:MarkdownPreview#开始预览
:MarkdownPreviewStop#停止预览
```

4.2.2 VIM 常用操作

4.2.2.1 使用寄存器与剪切板

用于与系统剪切板进行交互

```
vim --version|grep clipboard
    #查看clipboard与xtem_clipboard功能是否开启
#-clipboard -xtem_clipboard说明功能没有开启
sudo apt-get install vim-gtk3#安装对应功能模块
#再次查询为+clipboard +xtem_clipboard说明功能已开启，Debian12下
"+yy#复制当前行到系统剪切板
"+p#粘贴至其它文件中
```

LaTeX 中有许多字符有着特殊的含义，在你生成文档时不会直接打印。例如每个命令的第一个字符：反斜杠。单独输入一个反斜杠在你的行文中不会有任何帮助，甚至可能产生错误。LaTeX 中的保留字符有：

```
# $ % & _ { } \
```

它们的作用分别是：
#: 自定义命令时，用于标明参数序号。
§: 数学环境命令符。
以上除了反斜杠外，均能用前加反斜杠的形式输出。即你只需要键入：

```
\# \$ \% \^ \& \_ \{ \}
```

唯独反斜杠的输出比较头痛，你可以尝试：

```
1 $\backslash$ \textbackslash
2 \texttt{\char92}
```

\ \\

其中命令`\char[num]`是一个特殊的命令

`\texttt{\char`~}`%输出一个波浪线

任何一份 LaTeX 文档都应该包含以下结构：

```
1 \documentclass['\itshape options']{doc-class}%没有斜体option
2 \begin{document}
3 ...
4 \end{document}
```

其中，在语句`\begin{document}`之前的内容成为**导言区**。导言区可以留空，也可以进行一些、文档的准备操作。你可以粗浅地理解为：导演区即模板定义。

文档类的参数 doc-class 和可选选项 *options* 有取值：

表 4.1: 文档类和选项

doc-class	**文档类**[1]
article	- 科学期刊，演示文稿，短报告，邀请函。
proc	- 基于 article 的会议论文集。
report	- 多章节的长报告、博士论文、短篇书。
book	- 书籍。
slides	- 幻灯片，使用了大号 Scans Serif 字体。
options	
字体	- 默认 10pt，可选 11pt 和 12pt。

在本文中，多数的文档类提及的均为 report/book 类。如果有 article 类将会特别指明。其余的文档类不与说明。本手册排版即使用了 report 类。

在导言区最常见的是**宏包**的加载工作，命令形式如：`\usepackage{package}`。通俗地讲，宏包是指一系列已经制作好的功能“模块”，在你需要使用一些原生 LaTeX 不带有的功能时，只需要调用这些宏包就可以了。比如文本的代码就是利用`listings`宏包实现的。

宏包的具体使用将参在个部分内容说明中进行讲解。如果你想学习一个宏包的使用，按 Win+R 组合键呼出运行对话框，输入 texdoc 加上宏包名称即可打开宏包帮助 pdf 文档。例如：`texdoc xeCJK`。

4.2.3 错误的排查

在编辑器界面上，下方的日志是显示编译过程的地方。在你编译通过后，会出现这样的字样：/

- **Errors 错误**：严重的错误。一般地，编译若通过了，该项是零。
- **Warnings 警告**：一些不影响生成文档的瑕疵。
- **Bad Boxes 坏箱**[2]：指排版中出现的长度问题，比如长度超出（Overfull）等。后面的 Badness 表示错误的严重程度，程度越高数值越大。这类问题需要检查，排除 Badness 高的选项。

你可以向上翻越日志记录（即.log 文件），来找到 Warning 开头的记录，或者 Overfull/ Underfull 开头的记录。这些记录会指出你的问题出在哪一行（比如 line 1-2）或者在 pdf 的哪一页（比如 active[12]。注意，这个 12 表示计数器技术页码，而不是文件打印出来的真实页数）。此外你还需要了解：/

- 值得指出的是，由于 LaTeX 的编译原理（第一次生成 aux 文件，第二次再引用它），目录想要合理显示**需要连续编译两次**。在连续编译两此后，你会发现一些 Warnings 会在第二次编译后消失。在 TeX Studio 中，你可以只单击一次“构建并查看”，他会检测到文章的变化并自动决定是否需要编译两次。
- 对于大型文档，寻找行号十分痛苦。你需要学会合理地拆分 tex 文件，参阅内容。

这里也推荐宏包`syntonly`，在导言区加入它支持的`\syntaxonly`命令，会只排查语法错误而不生成任何文档，这可以使你更快地编译。不过他似乎不太稳定，例如本文档可以正常编译，但是使用该命令时则会出错。

此处注解在后续章节，目前未链接

[1] 此外还有`beamer`宏包定义的 beamer 文档类，常用于创建幻灯片。
[2] Box 是 LaTeX 中的一个特殊概念，具体将在进行讲解。

4.2.4 文件输出

LaTeX 的输出一般推荐 pdf 格式，有 LaTeX 直接生成 dvi 的方法并不推荐。

你在 tex 文档的文件夹下可能看到的其他文件类型：

.sty 宏包文件
.cls 文档类文件。
.aux 用于存储交叉引用信息的文件。
.out 宏包生成的 pdf 书签记录。

4.3 Git

Git 是一款分布式版本控制系统，Git 最初是由 Linux 开发者 Linus 用两周时间编写的，用来管理 Linux 源代码的。谈到 Git 就必须提到大名鼎鼎的github，它用的就是 git 系统来管网站的，github 和 git 是两个东西，github 是一个社区，git 是一个服务系统，github 只支持 git 分布式系统，所以故名成为 github。

4.3.1 Git 安装与配置

4.3.1.1 linux 系统下安装配置 git

linux 下安装和配置相当方便，甚至 git 已经内置与 linux 系统。

```
sudo apt install git#安装git
git config --list#查看git配置，用户名、邮箱
git config --global user.name "dk_huapen"#配置用户名称
git config --global user.email
    "zhaohuapeng7947@163.com"#配置用户邮箱
mkdir mygit
cd mygit
git init#初始化本地当前目录mygit为仓库
git status#查看本地仓库文件状态
git add file#将文件file提交到本地仓库的缓存区
git add --all#将所有改动过的文件提交到缓存区
git add .#将所有改动过的文件提交到缓存区
git commit -m "描述信息"#将本地仓库的提交缓存提交到本地仓库
git log#查看历史提交日至
#指定哪些文件或目录应该被Git忽略，不纳入版本控制
vim .gitignore
*~#忽略最后一个文件名字符是~的文件，用于忽略vim编辑生成的中间文件
```

4.3.1.2 windows 系统下安装配置 git

windows 下需要额外安装 git，安装 git 后会有一个类是 linux 的命令行工具 Git Bash。

4.3.2 Git 常用命令

4.3.3 github

github 是一款使用 git 命令作为基础框架的开源网站，可以把自己的项目分享到 github 上，gibhub 上是无限制代码的。首先需要在 github 上注册一个账号，我使用的是自己的 163 邮箱，密码是死骑 love 巫妖! 其实国内也有许多可以平替 github 的平台，而且 github 是国外平台，经常会网卡甚至掉线，不过我还是选择使用 github，因为它足够原生吧，就像我选择 Debian 一样。

4.3.3.1 本地 git 与 github 关联

```
#在本地创建一个ssh的可以，使用自己的邮箱
ssh-keygen -t rsa -C "zhaohuapeng7947@163.com"
ls
id_rsa#ssh生成的私匙，不能告诉任何人
id_rsa.pub#ssh生成的公匙，添加到github中的
ssh -T git@github.com
#测试ssh keys是否设置成功，也就是本地git与github是否关联成功
#下面的操作需要本地与github关联成功后才可以正常使用
#将本地仓库关联至github仓库并推送本地仓库至github
git remote add origin github仓库ssh地址#本地仓库与github仓库关联
git push -u origin master#将本地仓库推送至github
#clone的仓库自动关联github仓库
git clone <url>#用github远程默认仓库main初始化本地仓库
git clone <url> myhubname
#用github远程默认仓库main初始化本地仓库时自定义本地仓库名字
git clone -b master
    <url>#用github远程仓库master分支初始化本地仓库
git push#将本地仓库更新推送至github
git pull#从github获取更新
```

4.3.3.2 github 加速

4.4 make

make 是一个自动化工具。

4.5 LaTeX

LaTeX 是免费的、开源的排版系统，它的设计目标是分离内容与格式，以便作者能够专注于内容创作而非版式设计，并能得到高质量的排版作品。

4.5.1 latex 安装与配置

latex 的光盘镜像发布于 https://www.tug.org/texlive。在光盘镜像中 texlive-doc 目录下有一份 texlive-zh-cn.pdf 文件，里边详细描述了各操作系统安装 latex 的方法。

这里主要说两点注意事项：

1. 默认安装目录在/usr/local/texlive/下，安装完成后该目录有 8.7GB，安装系统时需要预留够磁盘分区空间
2. 安装菜单中 create symlinks in standard directories 选项默认不选，安装手册也不推荐，所以我安装也没选，安装完成后就得手动添加 texlive 安装路径到环境变量中。

```
#在/etc/profile文件尾部加入所有用户都生效
#在~/.profile文件尾部加入当前用户生效$
PATH=$PATH:/usr/local/texlive/2023/bin/x86_64-linux
export PATH
```

4.5.2 latex 常用命令

4.5.2.1 texdoc 命令

可以查询宏包帮助文档和其他相关的有用文档

texdoc [想要查询的内容]

```
texdoc fancyhdr #查看fancyhdr宏包帮助文档
texdoc usrguide #自带的用户手册
texdoc clsguide #自带文档类和宏包编写手册
texdoc fntguide #自带的字体使用手册
```

```
texdoc symbols-a4 #命令速查表
texdoc latexcheat #有趣的命令表
```

4.5.2.2 latexmk 命令

latexmk 命令可以实现 LaTeX 文档自动编译，指定输出目录等功能。我目前在 Makefine 文件中使用，看说明和论坛好像该命令本身就类是 Make 工具，不需要借助 make 就可以实现自动编译，目前还没弄明白这一点。

texdoc [参数][file]

```
latexmk -xelatex file.tex #自动编译file.tex文件
latexmk -c #清空指定临时文件
latexmk ? #指定输出文件目录，这个暂时没弄明白
```

中文的破折号你也许可以直接使用日常的输入方式。中文的省略号同样。但是注意，英文的省略号使用`\ldots`这个命令而不是三个句点。

4.6 MySQL

MySQL 是一款关系型数据库管理系统，这里主要记录下在 Windows 系统下的安装配置方法，在 Linux 下使用的是 MariaDB，在搭建 LAMP 服务器中再具体介绍 MariaDB 的安装配置方法。MariaDB 可以作为 MySQL 的替代品，MariaDB 是开源的性能更加优秀。

4.6.1 安装 MySQL

MySQL官网网站可以下载各个版本，每种版本中有安装版和解压版两种，前者是直接安装不需要过多配置的傻瓜式安装，后者是解压后自己配置文件直接就可以使用的免安装版本，果断选择后者。这种方式在 Windows 下很方便，很直观自己知道每一步在做什么，其实也没有几部，不像安装包干了点啥都不知道，还有可能出现安装失败的情况。我这里使用的是 MySQL5.5 版本，原因就是初次使用网上有篇参考教程讲的就是这个版本。

解压压缩包后，文件夹下有 5 个 ini 文件，这 5 个文件是针对用户五种情况下的配置文件，根据自己使用场景选择一个复制为 my.ini 文件，并修改部分设置主要是文件路径。

```
mysqld -install#安装mysql服务
mysqld -remove#卸载mysql服务
mysql -u root -p #登陆mysql，初次登录密码为空
#下边是在SQL环境下操作
update mysql.user set password=PASSWORD( '123456' ) where
    User= 'root' ;
#设置root用户密码
flush privileges;
#立即启用修改
GRANT SELECT ON tanhecha.* TO 'tanhechauser'@'localhost'
    IDENTIFIED BY 'tanhechapassword';
#添加用户并授权select权限
```

4.6.2 配置 MySQL

解压压缩包后，文件夹下有 5 个 ini 文件，这 5 个文件是针对用户五种情况下的配置文件，根据自己使用场景选择一个复制为 my.ini 文件，并修改部分设置主要是文件路径。

```
#开启日志备份
#设置日志备份时间
#在配置文件中打开独立的表空间
innodb_file_per_table=1
#在mysql中查看独立的表空间配置
show variables like '%per_table%';
```

4.6.3 MySQL 常用操作

4.6.3.1 备份与恢复数据库

我这里记录的备份是通过 mysqldump 工具将数据库备份到 sql 文本文件中，再登录数据库通过 source 命令将 sql 文件导入，整个恢复过程相当于按照备份的 sql 文件中 sql 语句操作数据库。

source 命令还有一个很好的用法就是批量操作数据库时，先按要求编辑号 sql 文本文件，然后 source 导入 sql 文本文件，让数据库自动批量执行 slq 文本中的命令，我在碳核查过程中使用的就是这个法方法备份、裁减、拼接成为每月独立的数据库的，需要进一步写成系统自动执行该系列操作。

```
mysqldump -u root -p test > test.sql
#输入密码开始备份,备份test数据库数据至sql文本文件
mysql -u root -p
```

```
#输入密码登陆mysql
mysql>create database test;#创建test数据库
mysql>use test;#切换至test数据库
mysql>set names utf8;#设置编码格式为utf8
mysql>source test.sql;#将test.sql导入test数据库
```

4.6.3.2 批量操作数据库

这里记录使用 MySQL 的 sql 文件批量操作数据库，主要是包括批量删除指定条件的数据、批量修改数据库内数据表的注释、清除主键自增值、合并两个数据表的值。使用的时命令行下登录数据库后通过 source 命令将写好的 sql 文件导入来批量操作数据库，在 windows 下使用 MySQL Workbench 执行时会提示错误，尤其时数据量大时，好像时数据库安全方面的而且速度慢，在命令行下没问题而且数据也可以。

source 命令还有一个很好的用法就是批量操作数据库时，先按要求编辑号 sql 文本文件，然后 source 导入 sql 文本文件，让数据库自动批量执行 slq 文本中的命令，我在碳核查过程中使用的就是这个法方法备份、裁减、拼接成为每月独立的数据库的，需要进一步写成系统自动执行该系列操作。

```
mysqldump -u root -p test > test.sql
#输入密码开始备份,备份test数据库数据至sql文本文件
mysql -u root -p
#输入密码登陆mysql
mysql>create database test;#创建test数据库
mysql>use test;#切换至test数据库
mysql>set names utf8;#设置编码格式为utf8
mysql>source test.sql;#将test.sql导入test数据库
```

```
delete from table;
#删除指定行数的数据
drop table if exists table;
#删除数据表所有数据和结构
truncate table table1,table2;
#删除数据表中所有数据但保留表结构，重置自增字段
ALTER TABLE course RENAME TO coursecopy;
#将course表重命名为coursecopy
CREATE TABLE course LIKE coursecopy;
#—将coursecopy表复制结构并创建名为course的新表
INSERT INTO course SELECT * FROM coursecop;
#将coursecopy表数据复制到course表
```

4.6.3.3 数据库同步

单向同步

4.7 MariaDB

讲解 MySQL 数据库时提到了 MariaDB 数据库，在 Linux 系统下还是安装使用 MariaDB 方便。MariaDB 使用基本与 MySQL 相同，在这里记录在 MariaDB 下的操作。

4.7.1 安装配置 MariaDB

下面以在Debian 下安装 MariaDB进行记录。

```
sudo apt install mariadb-server mariadb-client
#安装MariaDB数据库
sudo apt purge mariadb-server#卸载MariaDB
sudo rm -rf /var/lib/mysql/#彻底删除MariaDB
sudo systemctl start mariadb #启动MariaDB数据库
sudo systemctl enable mariadb #设置MariaDB自动启动
sudo systemctl status mariadb #查看MariaDB状态
sudo mariadb-secure-installation #初始化MariaDB数据库
#包括设置root密码（默认为空），删除匿名用户等
#有一项是禁止root用户远程登陆，需要根据个人需求选择
mysql -u root -p#登陆MariaDB数据库
```

4.7.2 触发器

触发器是一种特殊的存储过程，它在插入，删除或修改特定表中的数据时触发执行，我是在更新数据表的同时用更新的数据计算更新另一张表内数据。触发器必须是在指定数据库内的，就是得先 use database 然后在在 database 上创建触发器。

```
use buff;#进入buff数据库
#创建触发器
create trigger update_trigger2 after update on ua for each row
    update boler set value = NEW.value where kks =
    left(OLD.kks,12);
show triggers\G;#查看所有触发器
show create trigger update_trigger2\G;#查看指定触发器
drop trigger update_trigger2;#删除触发器
```

4.7.3 事物

4.8 PHP

PHP 全称 Hypertext Preprocessor，中文名：“超文本预处理器”是一种通用开源脚本语言。语法吸收了 C 语言、java 和 Perl 的特点，利于学习，使用广泛，主要适用于 web 开发领域。用 PHP 做出的动态页面与其他的编程语言相比，PHP 是将程序嵌入到 HTML（标准通用标记语言下的一个应用）文档中去执行，执行效率比完全生成 HTML 标记的 CGI 要高许多；PHP 还可以执行编译后代码，编译可以达到加密和优化代码运行，使代码运行更快。

4.8.1 phpmyadmin 的使用

搭建 LAMP 平台后将解压后的 phpmyadmin 文件放到网站根目录下/var/www/html/即可访问数据库。

4.8.2 PHP 二维码生成

PHP QRCode 全称 PHP Quick Response Code，是一个在 PHP 平台下生成二维码的开源库。官网下有很多使用它生成二维码的例子，这里记录两种我使用过的情况。

4.8.2.1 静态网页直接显示二维码

这种情况是在生成网页的时候直接将二维码打印在网页上，用官网的话叫 outputs image directly into browser, as PNG stream。

```
<?php

    include('../lib/full/qrlib.php');

    $param = $_GET['id']; // remember to sanitize that - it is
    user input!

    // we need to be sure ours script does not output
    anything!!!
    // otherwise it will break up PNG binary!

    ob_start("callback");

```

```
12      // here DB request or some processing
13
14      // end of processing here
15      $debugLog = ob_get_contents();
16      ob_end_clean();
17
18      // outputs image directly into browser, as PNG stream
19      QRcode::png($codeText);
20  ?>
```

4.8.2.2 AjaX 动态更新二维码

这种情况是在点击不同元素动态生成二维码打印在网页上，动态更新网页就需要用到 AjaX 技术，但是 AjaX 返回数据要求是字符串不能是二进制流，所以需要先将 PNG stream 转换成字符串返回到前端后再显示。

```
1  <?php
2  include_once("./phpqrcode/qrlib.php");
3  $param = $_GET['id'];
4  $codeText = $param;
5  ob_start();
6  QRcode::png($codeText);
7  $debugLog = base64_encode(ob_get_contents());
8  ob_end_clean();
9  echo $debugLog;
10 ?>
11 #前端javascript中
12 document.getElementById('src').src=
13 ''+this.responseText;
```

4.8.3 php 生成 PDF 文件

加载离线文件就可以生成 PDF 文件

4.9 Python

Python 是一种解释型语言，它的优点就是有丰富的库，使得编成变的简单，我是在实现 OPC DA 的 DCOM 编成和 OPC UA 服务器时开始使用它，库确实很强大，很容易就完成了想要的功能，所以决定学习该语言。

4.9.1 pip

pip 是用来安装和更新 Python 库的，由于 pip 默认的源是国外的，安装更新很慢导致安装库时经常超时失败，所以安装 pip 后需要将源换成阿里云 http://mirrors.aliyun.com/pypi/simple/

国内镜像源：

1. 清华大学 https://pypi.tuna.tsinghua.edu.cn/simple/
2. 中国科技大学 https://pypi.mirrors.ustc.edu.cn/simple/
3. 豆瓣 http://pypi.douban.com/simple
4. Python 官方 https://pypi.python.org/simple/
5. v2ex http://pypi.v2ex.com/simple/
6. 中国科学院 http://pypi.mirrors.opencas.cn/simple/
7. 中国科学技术大学 [http://pypi.mirrors.ustc.edu.cn/simple/
8. 华中理工大学：http://pypi.hustunique.com/
9. 山东理工大学：http://pypi.sdutlinux.org/

```
#首先省级最新pip版本
python -m pip install --upgrade pip
#单次安装使用国内镜像源以tensorflow库为例
pip install tensorflow -i
    http://mirrors.aliyun.com/pypi/simple/ --trusted-host
    mirrors.aliyun.com#安装最新版本
pip install tensorflow==2.13.0 -i
    http://mirrors.aliyun.com/pypi/simple/ --trusted-host
    mirrors.aliyun.com#安装指定版本
#永久设置国内镜像源，两项都要设置否则报错
pip config set global.index-url
    http://mirrors.aliyun.com/pypi/simple/#设置index-url
pip config set global.trusted-host
    mirrors.aliyun.com#设置trusted-host
#以上两句命名相当于在~/.config/目录下创建pip配置文件
cd ~/.config/
mkdir pip
vim pip.conf

[global]
index-url = http://mirrors.aliyun.com/pypi/simple/
trusted-host = mirrors.aliyun.com
```

4.9.2 Python 虚拟环境

Python 虚拟环境可以创建一个独立的环境，用于安装不同项目所需的特定 Python 包和依赖项，甚至是不同版本的 Python 环境，这个功能对于需要不同版本 Python 同时安装和 Python 环境迁移非常有用。下面以 Linux 和 Windows 下安装分别记录

Debian 系统安装 Python 虚拟环境

```
sudo apt-get install python3-venv#创建虚拟环境的env模块
sudo apt-get install python3-pip#Python包管理工具
python3 -m venv myvenv#创建名称为myvenv的虚拟环境
source myvenv/bin/activate#激活myvenv虚拟环境
deactivate#推出myvenv虚拟环境
```

Windows 系统安装 Python 虚拟环境

```
pip install virtualenv #创建虚拟环境的env模块
virtualenv -p D:\Python\Python312\Python.exe myvenv
#创建名称为myvenv的虚拟环境
myvenv/Scripts/activate#激活myvenv虚拟环境
myvenv/Scripts/deactivate#退出myvenv虚拟环境
```

4.9.3 爬虫

爬虫不能一直研究只能随缘，属于奇淫技巧，不能耗费过多时间。

4.9.3.1 百度翻译

这个是第一个学习的第一个爬虫项目-百度翻译的单词爬虫，结果是运行爬虫后直接输入单词或汉字可以直接翻译出结果来。

```
import requests

def spider(url,headers,data):

    response = requests.post(url=url, headers=headers,
    data=data).json()  # 对目标url发起post请求
    for key in response['data'][0]:
        print(key,response['data'][0][key])

def main():

```

```
    url = 'https://fanyi.baidu.com/sug'  #需要请求的url
    headers = {  #进行UA伪装
        'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36 Edg/98.0.1108.56'
    }
    while True:  #使程序进入死循环
        kw = input("输入需要查询的单词: ")
        data = {     #post请求携带的参数
            'kw':kw
        }
        spider(url=url,headers=headers,data=data)
    #调用自定义函数spider

main()
```

4.9.3.2 爬取在线视频

这是一个爬取在线视频项目，爬取喜马拉雅音频的失败了，因为 path 都为空，使用了最新的反爬手段

在爬去视频时明白原来网络视频都是一小段一小段，一个 40 分钟电视剧分散成有 700 多个小视频，插入广告非常方便，去除广告也非常方便哦!

4.9.4 自动化办公

Python 的自动化办公可是比较有名的，所以我也试验了下，确实很强大。我的想法是借助 Python 自动生成 Word 和 Excel 文件功能实现服务器端存储数据源文件，需要的时候自动生成需要的格式文件。

```
#这里只简单记录下需要安装的模块，详细的参考网页资料
#值得注意的是只支持指定格式的文件并非所有excel和word文件
pip install openpyxl#安装xlsx文件的模块
pip install python-docx#安装docx文件的模块，导入时是import docx
```

4.9.5 百度网盘

Python 使用 bypy 模块可以操作百度网盘，包括显示文件列表、同步目录、文件上传，不过只支持/apps/bypy 目录。

借助该模块可以搭建自己的 1T 云备份和同步系统。

```
pip install bypy#安装模块
bypy list #显示云盘根目录下文件列表
#首次操作点击终端上方的蓝色链接，复制授权码并回车完成授权
```

```
from bypy import ByPy
bp = ByPy()
print(bp.list())
bp.upload(r"localfile","disfile")#上传文件
bp.download(r"localfile","disfile")#下载文件
bp.syncup(r"localdir","disdir")#上传文件夹
bp.syncdown("remotedir",r"localdir")#下载文件夹
```

4.9.6 Django

4.10 ImageMagick

ImageMagick 是一套功能强大、稳定而且免费的工具集和开发包，它可以单独使用来处理图片而且还可以在 PHP 调用来自动处理图片。

webp 是一种新的图像格式，用于 web 项目，可以大大提高网站访问速度。同样的分辨率，大小比 jpg、png 小 25% 以上。我的网站上上传的图片就是在 PHP 中自动将图片转换为 webp 格式图片，转换后的图片大小小了不止 25%。

4.10.1 ImageMagick 安装

记录在Debian12 下安装 ImageMagick。

```
#安装依赖库
sudo apt-get install build-essential
sudo apt-get install libjpeg-dev libpng-dev libtiff-dev
    libgif-dev libwebp-dev
sudo apt-get install webp
#github下载ImageMagick源文件进行编译安装
tar xf ImageMagick-7.1.1-47.tar.gz
cd ImageMagick-1.1-47
./configure
sudo make install
magick --version#安装完成后，检查版本信息
#如果出现下面错误
magick: error while loading shared libraries:
    libMagickCore-7.Q16HDRI.so.10: cannot open shared object
```

```
    file: No such file or directory
#执行以下命令
echo "/usr/local/lib" >>sudo  /etc/ld.so.conf
sudo ldconfig
magick --version#再次检查版本信息
```

4.10.2 ImageMagick 命令

```
#将png图片格式转换为webp格式
/usr/local/imagemagick/bin/convert file1.png file3.webp
#将图片裁减为指定大小
/usr/local/imagemagick/bin/convert -sample 768x1024 file3 file3
#在图片上添加水印
/usr/local/imagemagick/bin/convert -fill red -pointsize 60
    -draw 'text␣300,80␣"dklovelich"' file3 file3
```

4.10.3 PHP 下安装 ImageMagick 扩展

安装 PHP 下的 ImageMagick 扩展模块后就可以在 PHP 中使用 ImageMagick 处理图片了。但是安装未能成功。

4.11 Nginx

4.12 Apache

4.12.1 Apache 安装配置

4.12.2 apache2 根目录修改

Apache2 的根目录默认为/var/www，如果需要修改到自定义地址，涉及两个关键配置文件的调整。

1. 修改/etc/apache2/apache2.conf, 把文件里面的/var/www 改成你的目标地址
2. 修改/etc/apache2/sites-enabled/000-default.conf,把文件里面的/var/www 改成你的目标地址

4.13 Docker

docker 是一种虚拟化技术，和之前接触比较类似的就是虚拟机，区别就是虚拟机需要安装操作系统然后安装要使用的应用程序，消耗资源比较多，docker 就是在本机上构建一个独立的虚拟环境来运行应用程序比较节省资源。Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可抑制的容器中，然后发布到任何流行的 Linux 机器上，也可以实现虚拟化。容器完全使用沙盒机制，相互之间不会存在任何接口。几乎没有性能开销，可以很容易的在机器和数据中心运行。最重要的是，他们不依赖于任何语言、框架或者包装系统。

4.13.1 Docker 安装部署

在 CSDN 上找了很多篇帖子也没有在 Debian11 上安装成功，后来在博客园上找到一篇安装成功了。安装成功后无法正常拉取镜像，后来修改镜像后拉取成功了。

```
sudo apt-get update#更新软件包索引
sudo apt-get install apt-transport-https ca-certificates curl
    gnupg2 software-properties-common#安装必要的软件包
curl -fsSL
    https://mirrors.aliyun.com/docker-ce/linux/debian/gpg |
    sudo apt-key add -#添加阿里云的Docker官方GPG密钥
sudo add-apt-repository "deb␣[arch=amd64]␣
    https://mirrors.aliyun.com/docker-ce/linux/debian␣
    $(lsb_release␣-cs)␣stable"#添加Docker仓库地址源
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli
    containerd.io#安装Docker CE
sudo systemctl status docker
sudo usermod -aG docker $USER#设置非root用户也能运行Docker
#更新Docker源，否则无法正常拉取镜像
cd /etc/docker/
vim daemon.json#没有就创建该文件
{
 "registry-mirrors": ["https://docker.registry.cyou",
"https://docker-cf.registry.cyou",
"https://dockercf.jsdelivr.fyi",
"https://docker.jsdelivr.fyi",
"https://dockertest.jsdelivr.fyi",
"https://mirror.aliyuncs.com",
"https://dockerproxy.com",
"https://mirror.baidubce.com",
```

```
"https://docker.m.daocloud.io",
"https://docker.nju.edu.cn",
"https://docker.mirrors.sjtug.sjtu.edu.cn",
"https://docker.mirrors.ustc.edu.cn",
"https://mirror.iscas.ac.cn",
"https://docker.rainbond.cc"]
}
systemctl daemon-reload
systemctl restart docker
```

4.13.2 Docker 使用

```
docker pull mysql:5.7#拉取mysql的镜像
docker images#查看本地镜像
```

4.14 Grafana

Grafana 是一款开源的数据可视化工具，主要用于大规模指标数据的可视化展示。这是我在搭建 Pyscada 平台时接触到的一款软件，它可以用于历史曲线显示和监控画面显示，很厉害。可以在官网下载该软件的 linux 和 windows 版本。

4.14.1 安装配置 Grafana

Grafana 的安装比较简单，在 windows 下甚至不需要安装直接解压就可以使用了。

```
#Debian下安装
wget
    https://dl.grafana.com/oss/release/grafana_10.2.3_amd64.deb
sudo dpkg -i grafana_10.2.3_amd64.deb
sudo systemctl start grafana-server
sudo systemctl enable grafana-server
```

4.14.1.1 使用 Grafana

Grafana 的使用比较复杂,安装完成后在浏览器使用 http://localhost:3000 就可以登录页面，首次登录用户名和密码都是 admin，登录后需要修改密码。

配置数据源，需要首先增加用户并授权指定数据库的 select 权限才能添加成功，否则会添加失败，用 root 用户无法设置权限不足。

4.15 GoldenDict

GoldenDict是一款免费的 linux 预装的字典，它可以导入下载好的字典离线查询，也可以设置好网址后在线查询。.

4.16 OpenTTD

OpenTTD 是一款运输模拟游戏，安装很简单，直接从官网上下载解压即可，无需安装，但是它的玩法门槛很高而且容易上头。

西文中一般采用上述的斜体强调方式而不是粗体，例如在说明书的时候可能就会使用以上命令。关于字体更多内容参考字体这一节。

4.16.1 下划线与删除线

LaTeX 原生提供的`\underline`命令简直烂的可以，建议你使用`ulem`宏包下的 `uline` 命令代替，它还支持换行文本。`ulem`宏包还提供了一些实用命令：

项目平台 | 5

这一章主要记录一些小的项目搭建，这些小的项目需要安装操作系统和一些软件组合并经过一些配置搭建起一个工作平台，然后在这个平台下做一些事情。比如 LAMP、SIS 系统、Smart PLC 培训。

5.1 我的个人网站

我的个人网站是我一直在学习中搭建的一个网站，其中使用了很多技术，在此记录下。

5.1.1 php 生成 PDF 文件

5.1.2 远程操作 latex 生成 pdf 文件

5.1.3 远程操作 python 生成 doc 文件

5.1.4 动态更新 svg 画面

5.1.5 Dygraph 生成历史曲线

5.2 LAMP 环境

LAMP(Linux Apache Mysql Php)是指一组通常一起使用来运行动态网站或者服务器的自由软件名称首字母缩写;Linux 系统下 Apache+MySQL+PHP 这种网站服务器架构，LAMP 环境主要是给 WEB 端应用程序 (各种类型的网站项目)，提供了一个部署安装和使用的平台。

L: Linux 操作系统，提供了项目部署时所需要的操作系统环境

A: Apache 服务器：WEB 应用程序的服务器，提供软件源文件的存放地，提供了程序访问时所需要的端口 (接口)

M: MySQL 数据库，提供项目或者程序在使用时数据的存储与解析的工作

P: PHP/Python 开发语言，提供软件或者项目程序部署时所需要的开发环境的支持

只要把这四个软件安装完成，就形成了 LAMP 环境，环境有了之后，只需要把 WEB 应用程序对应的源文件，部署在 apache 服务器上即可，这样用户就可以直接访问该网站。

5.2.1 LAMP 环境搭建

LAMP 环境搭建的指定就是在 Linux 系统上安装这三款软件然后配置相关文件后协同工作来建设一个 WEB 服务器。这几款软件的安装前边已经记录过了，这里主要记录软件之间相互配置调用和搭建过程中注意事项。

主要是 PHP 版本必须是 PHP7，而且随着 linux 系统升级 PHP7 安装过程中会有各种问题，这里记录在 Debian12 环境下的注意事项，安装完成后记得重启 apache2 服务，不然部分 php 模块无法正常加载。

5.3 LNMP

LNMP(Linux Nginx Mysql Php)是指一组通常一起使用来运行动态网站或者服务器的自由软件名称首字母缩写;Linux 系统下 Nginx+MySQL+PHP 这种网站服务器架构，LNMP 环境主要是给 WEB 端应用程序 (各种类型的网站项目)，提供了一个部署安装和使用的平台。

5.3.1 LNMP 环境搭建

我最初接触的是 LAMP 环境，在搭建 Pysacada 环境时因为需要才开始接触 LNMP 环境，相比较而言 LNMP 就是把 WEB 服务由 Apache 更换为 Nginx。

5.4 PyScada

pyScada 是一款基于 Python 的开源 SCADA 系统，可以在它的开源项目和官方网站上了解和下载该系统。

5.4.1 PyScadaa 安装部署

现将PyScada 系统安装记录记录在此，值得注意的是 PyScada 系统是以 NGINX 作为 WEB 服务器的，所以得先停运 Apache2 服务器，否则无法正常启动 NGINX，因为它两使用同一端口，不过可以重新配置端口让两种服务器同时运行，甚至经过配置还可以让两种服务器配合运行各取所长，这也是我下一个目标。

官网安装指导部分有这么一句话 This installation guide covers the installation of PyScada for Debian 10/11 , Raspberry Pi OS based Linux systems using MariaDB as Database, Gunicorn as WSGI HTTP Server and nginx as HTTP Server。

1. 获取安装包，值得注意的是解压缩需要使用命令行工具 uzip，否则安装最后后出现文件权限问题
2. 停运 Apache2 服务,最好禁止 Apache2 开机启动否则下次开机,PyScada 还是不能正常启动

3. 安装 MariaDB 并初始化完成，安装过程中要使用数据库并创建专门的数据库
4. 可以选择本机或 docker 安装，我们选着本机，因为 docker 还未安装成功...
5. 安装 Python 是必须的，设置 pip 国内源否则安装过程中下载速度过慢甚至报错失败
6. 不需要安装虚拟环境，因为安装第一步就是创建 PyScada 文件夹并创建.env 虚拟环境

```
#在Debian11下安装PyScada记录
sudo apt install wget
wget
    https://github.com/pyscada/PyScada/archive/refs/heads/main.zip
    -O PyScada-main.zip
sudo apt install unzip
unzip ./PyScada-main.zip
rm ./PyScada-main.zip
cd PyScada-main

sudo ./install.sh
```

5.4.2 PyScada 安装插件

刚安装完成的 PyScada 中 Devices 中驱动只有一个 generic，我们需要安装我们使用的 OPCUA 驱动。官网的例子是安装 Modbus 驱动，我们照猫画虎安装 OPCUA 驱动，在这里可以看到和下载所需要的驱动。

```
#安装PyScada-OPCUA驱动
sudo apt install git
cd /home/pyscada
sudo -u pyscada git clone
    https://github.com/pyscada/PyScada-OPCUA.git
cd PyScada-OPCUA

# 激活PyScada虚拟环境
source /home/pyscada/.venv/bin/activate
#安装驱动
sudo -u pyscada -E env PATH=${PATH} pip3 install
    .#特别注意这个地方有个点表示当前目录
# run migrations
sudo -u pyscada -E env PATH=${PATH} python3
    /var/www/pyscada/PyScadaServer/manage.py migrate
```

```
# copy static files
sudo -u pyscada -E env PATH=${PATH} python3 /var/www/pyscada/PyScadaServer/manage.py collectstatic --no-input
#重新启动gunicorn和PyScada服务
sudo systemctl restart gunicorn pyscada

pip3 list | grep cada#查看驱动是否安装成功
sudo -u pyscada -E env PATH=\${PATH} pip3 uninstall yourPlugin#卸载驱动
```

5.4.3 PyScadaa 使用

在本机浏览器输入 127.0.0.1 会出现登录界面，输入你安装过程中创建的账号密码就可以进入系统，在安装成功后也会提示这一步并显示登录账号密码。点击右上角 Admin 后出现后台管理界面

5.4.3.1 编写一个 OPCUA 服务器

这里有个插入一个小插曲，因为后边需连接 OPCUA 服务器读取变量进行测试，如果安装软件或者是从其他主机联机读取的话比较费劲，这里使用 Python 编写一个简单的 OPCUA 服务器，很方便的，代码如下

```
import sys
sys.path.insert(0, "..")
import time
from opcua import ua, Server
if __name__ == "__main__":
    # setup our server
    server = Server()

    #server.set_endpoint("opc.tcp://127.0.0.1:4840/freeopcua/server/")

    server.set_endpoint("opc.tcp://192.168.1.5:4840/freeopcua/server/")
    # setup our own namespace, not really necessary but should as spec
    uri = "http://automan.freeopcua.github.io"
    idx = server.register_namespace(uri)
    # get Objects node, this is where we should put our nodes
    objects = server.get_objects_node()
    # populating our address space
    myobj = objects.add_object(idx, "MyObject")
    myvar = myobj.add_variable(idx, "MyVariable", 6.7)
```

```
18      myvar.set_writable()    # Set MyVariable to be writable by
        clients
19      # starting!
20      server.start()
21      try:
22          count = 0
23          while True:
24              time.sleep(1)
25              count += 0.1
26              myvar.set_value(count)
27      finally:
28          #close connection, remove subcsriptions, etc
29          server.stop()
```

5.4.3.2 添加第一个 OPCUA 变量

首先添加 Devie，然后添加变量，选择刚刚添加的 deveic，重要的是变量最下方的 s 和 n 需要按照 UA 服务器变量的 ID 输入。

5.5 SIS 系统

第一款软件无疑就是 VIM 了，那第二款肯定是 Latex 了，这玩意就是用他两鼓捣出来的。编辑器的配置大概是需要讲解一下的，毕竟对于初学者来说是很头疼的事情。本手册就以 TEX studio 为例进行配置。首先你应该安装一个 TEX Live，他是完全免费的，网址：http://tug.org/texlive/。

5.5.1 单向网闸配置

我接触单向网闸是在 2024 年 9 月份，记得当时是公司 SIS 系统按照等保测评要求在 DCS 系统和 SIS 系统中间使用了电力系统专用的单向隔离网闸，它的特点是数据只能从内网向外网正常传输，外网向内网传输数据只能是单 Bit，所以就导致外网安装的软件 KepWare 无法正常发送连接请求到 DCS 系统 OPC DA 服务器，最后还是我使用 Python 语言 DCOM 编成从 DCS OPC DA 服务器读取出数据。

5.5.1.1 平台搭建

在 Win7 SP1 64 位操作系统安装 NR 软件后，配置主机 IP 地址为，用网线连接电脑和网闸内网管理口，ping 测试正常，将 2 和 1 转换器插到内网 console 口，并在 2 和 1 转换器插入其中一个操作员秘匙。

5.5.1.2 设备激活

初次登陆需要创建系统管理员，账户：rekongroot 密码：rekong1314root! pin 码：Nari6702 初始化用户信息，ukey 账户信息被删除，生成 ukey 证书请求 (内容随便填写)，使用证书签发系统签发后，将签发证书上传装置。出现登录界面用系统管理员账户登录后开始生成激活申请文件，其中最终用户名称必须是购买合同中的单位名称，按照规定格式发送邮件并将激活申请文件作为附件上传，等待邮件回复。

虽然它体积较大，但是却是最一劳永逸、最不需要花时间去配置的方法，同时它大概也是功能支持最强的 LaTeX 发行版。

打开 TeX Studio 后，选择选项 → 设置 TeX Studio → 构建 → 默认编译器，选择 XeLaTeX。这主要是基于中文文档编译的考虑，同时 XeLaTeX 也能很好的编译英文文档。我建议始终使用它作为默认编译器。

之后你可以在窗口输入一篇小文档，并保存为 tex 扩展名的文件进行测试：

```
\documentclass{ctexart}
\begin{document}
Hello,world!
你好，世界!
\end{document}
```

点击编译按钮生成，F7 查看。生成 pdf 在你的 tex 文件保存目录中。具体各行的含义我们后在后文介绍。

5.6 Smart PLC 培训

这个主要是针对 2024 年厂里的一套氨水控制系统做培训时搭建的过程记录 Windows 7 旗舰版 Service Pack 1 64 位操作系统 STEP7-MicroWIN-SMAT-V2.4 WinCC V7.3 PC_ACCESS_V2.3

5.6.1 VIM 插件管理 Plum-vim

LaTeX 中的**命令**通常是由一个反斜杠加上命令名称，再加上花括号内的参数构成的（有的命令不带参数，例如：`\TeX`）。

```
\documentclass{ctexart}
```

如果有一些选项是备选的，那么通常会在花括号前用方括号标出。比如：

```
\documentclass[a4paper]{ctexart}
```

还有一种重要指令叫做**环境**。它被定义与控制命令`\begin{environment}`和`\end{environment}`间的内容。比如：

```
\begin{document}
...内容...
\end{document}
```

环境如果有备选参数，只需要写在`\begin[...]{name}`这里就行。

注意：不带花括号的命令后面如果想打印空格，请加上一对内部为空的花括号再键入空格。否则空格会被忽略。例如：`\LaTex{} Studio`。

5.6.2 VIM 常用操作

LaTeX 中有许多字符有着特殊的含义，在你生成文档时不会直接打印。例如每个命令的第一个字符：反斜杠。单独输入一个反斜杠在你的行文中不会有任何帮助，甚至可能产生错误。LaTeX 中的保留字符有：

```
# $ % & _ { } \
```

它们的作用分别是：
#：自定义命令时，用于标明参数序号。
§：数学环境命令符。
以上除了反斜杠外，均能用前加反斜杠的形式输出。即你只需要键入：

```
\# \$ \% \^ \& \_ \{ \}
```

唯独反斜杠的输出比较头痛，你可以尝试：

```
$\backslash$ \textbackslash
\texttt{\char92}
```

\ \\

其中命令`\char[num]`是一个特殊的命令

`\texttt{\char`~}`%输出一个波浪线

5.6.3 导言区

任何一份 LaTeX 文档都应该包含以下结构：

```
\documentclass['\itshape options']{doc-class}%没有斜体option
\begin{document}
...
\end{document}
```

其中，在语句`\begin{document}`之前的内容成为**导言区**。导言区可以留空，也可以进行一些、文档的准备操作。你可以粗浅地理解为：导演区即模板定义。

文档类的参数 doc-class 和可选选项 *options* 有取值：

表 5.1: 文档类和选项

doc-class 文档类[1]	
article	- 科学期刊，演示文稿，短报告，邀请函。
proc	- 基于 article 的会议论文集。
report	- 多章节的长报告、博士论文、短篇书。
book	- 书籍。
slides	- 幻灯片，使用了大号 Scans Serif 字体。
options	
字体	- 默认 10pt，可选 11pt 和 12pt。

在本文中，多数的文档类提及的均为 report/book 类。如果有 article 类将会特别指明。其余的文档类不与说明。本手册排版即使用了 report 类。

在导言区最常见的是**宏包**的加载工作，命令形式如：`\usepackage{package}`。通俗地讲，宏包是指一系列已经制作好的功能“模块”，在你需要使用一些原生 LaTeX 不带有的功能时，只需要调用这些宏包就可以了。比如文本的代码就是利用`listings`宏包实现的。

宏包的具体使用将参在个部分内容说明中进行讲解。如果你想学习一个宏包的使用，按 Win+R 组合键呼出运行对话框，输入 texdoc 加上宏包名称即可打开宏包帮助 pdf 文档。例如：`texdoc xeCJK`。

5.6.4 错误的排查

在编辑器界面上，下方的日志是显示编译过程的地方。在你编译通过后，会出现这样的字样：/

- **Errors 错误**：严重的错误。一般地，编译若通过了，该项是零。

[1] 此外还有`beamer`宏包定义的 beamer 文档类，常用于创建幻灯片。

- **Warnings 警告**：一些不影响生成文档的瑕疵。
- **Bad Boxes 坏箱**[2]：指排版中出现的长度问题，比如长度超出（Overfull）等。后面的 Badness 表示错误的严重程度，程度越高数值越大。这类问题需要检查，排除 Badness 高的选项。

你可以向上翻越日志记录（即.log 文件），来找到 Warning 开头的记录，或者 Overfull/ Underfull 开头的记录。这些记录会指出你的问题出在哪一行（比如 line 1-2）或者在 pdf 的哪一页（比如 active[12]。注意，这个 12 表示计数器技术页码，而不是文件打印出来的真实页数）。此外你还需要了解：/

此处注解在后续章节，目前未链接

- 值得指出的是，由于 LaTeX 的编译原理（第一次生成 aux 文件，第二次再引用它），目录想要合理显示**需要连续编译两次**。在连续编译两此后，你会发现一些 Warnings 会在第二次编译后消失。在 TeX Studio 中，你可以只单击一次“构建并查看”，他会检测到文章的变化并自动决定是否需要编译两次。
- 对于大型文档，寻找行号十分痛苦。你需要学会合理地拆分 tex 文件，参阅内容。

这里也推荐宏包`syntonly`，在导言区加入它支持的`\syntaxonly`命令，会只排查语法错误而不生成任何文档，这可以使你更快地编译。不过他似乎不太稳定，例如本文档可以正常编译，但是使用该命令时则会出错。

5.6.5 文件输出

LaTeX 的输出一般推荐 pdf 格式，有 LaTeX 直接生成 dvi 的方法并不推荐。

你在 tex 文档的文件夹下可能看到的其他文件类型：

.sty	宏包文件
.cls	文档类文件。
.aux	用于存储交叉引用信息的文件。
.out	宏包生成的 pdf 书签记录。

5.7 latex

英文符号一般用于数学 | <> + = 一般用于数学环境中，如果在文本中使用，请在它们两侧加上“$”。如果你在 LaTeX 中直接输入大于、小于号而不是把它们、放在数学环境中，它们并不会被正确打印。你应该使用`\textgreater`，`\texless`命令。

[2]Box 是 LaTeX 中的一个特殊概念，具体将在进行讲解。

在部分科技文章中，中文的句号可能使用全角原点“．”[3]，而不是平常的“。”，也不是正常的英文句点“.”。这个符号很难正常输入；你可以先输入正常句点，最后再替换。

5.7.1 latex 安装与配置

英文单引号并不使用两个'符号组合。左单引号是重音符`（键盘上数字 1 左侧），而右单引号是常用的引号符。英文中，左双引号就是连续两个重音符号。英文下的引号嵌套需要借助`\thinspace`命令分隔，比如：

```
%方括号里的是啥意思?
``\thinspace`Max' is here.''
```

“‘Max’ is here.”

中文下的单引号和双引号你可以用中文输入法直接输入。

5.7.2 latex 常用命令

英文的短横分为三种：/

- 连字符：输入一个短横：-，效果如 daughter-in-law
- 数字起止符：输入两个短横：--，效果如：page 1–2
- 破折号：输入三个短横：---，效果如：Listen—I’m serious.

中文的破折号你也许可以直接使用日常的输入方式。中文的省略号同样。但是注意，英文的省略号使用`\ldots`这个命令而不是三个句点。

5.7.3 Tizk

LaTeX 中专门有个叫做`\emph{text}`的命令，可以强调文本。对于通常的西文文本，上述命令的作用就是斜体。如果你对一段已经这样转换为斜体的文本再使用这个命令，它就会取消斜体，而成为正体。

西文中一般采用上述的斜体强调方式而不是粗体，例如在说明书的时候可能就会使用以上命令。关于字体更多内容参考字体这一节。

5.7.4 下划线与删除线

LaTeX 原生提供的`\underline`命令简直烂的可以，建议你使用`ulem`宏包下的 `uline` 命令代替，它还支持换行文本。`ulem`宏包还提供了一些实用命令：

[3]这个标点是 u+FF0E，称为 FULLWIDTH FULL STOP。

索引 6

7 参考文献

[1] K.L Wu. 简单粗暴 LaTeX, 2021.

[2] 刘海洋. LaTeX 入门. 电子工业出版社, 2013.