[野火]sphinx 文档规范与模版

2020 年 08 月 05 日

为什么使用 sphinx 写文档

CHAPTER 1

关于本项目

本项目的 github 地址：https://github.com/Embdefire/ebf_contribute_guide

本项目的 gitee 地址：https://gitee.com/wildfireteam/ebf_contribute_guide

点击右侧链接可在线阅读本项目文档：《[野火]sphinx 文档规范与模版 》

本文档是关于如何使用 sphinx 编写文档的说明。也可以把本文档作为模版创建其它 sphinx 的文档。

1.1 为什么使用 sphinx 编写文档

使用 shpinx 编写文档有如下优点：

- 使用 sphinx 编写的文档可以方便地制作 html、pdf 等格式，非常方便浏览和转换。
- sphinx 支持 rst 和 markdown 语法，方便共享及开源编辑，使用 git 也方便跟踪。
- 由于 rst 语法比 markdown 语法更强大和方便，我们主要采用 rst 语法编写文档，linux 内核源码文档也是使用 rst 格式编写的。我们的文档也支持 markdown，主要是为了方便不熟悉 rst 的用户参与进来。

CHAPTER 2

关于野火

2.1 开源共享，共同进步

野火在发布第一块 STM32 开发板之初，就喊出 **开源共享，共同进步**的口号，把代码和文档教程都免费提供给用户下载，而我们也一直把这个理念贯穿至今。

目前我们的产品已经包括 *STM32*、*i.MX RT 系列*、*GD32V*、*FPGA*、*Linux*、*emXGUI*、操作系统、网络、下载器等分支，覆盖电子工程应用领域的各种常用技术，其中教学类产品的代码和文档一直保持着开源的姿态发布到网络上，为电子工程师排忧解难，让嵌入式没有难用的技术是我们最大的愿望。

2.2 联系方式

- 官网：http://www.embedfire.com
- 论坛：http://www.firebbs.cn
- github 主页：https://github.com/Embedfire
- gitee 主页：https://gitee.com/wildfireteam
- 淘宝：https://yehuosm.tmall.com
- 邮箱：embedfire@embedfire.com
- 电话：0769-33894118

CHAPTER 3

快速参与本项目（提交 bug 或文档修改）

开源代码和文档最重要的初衷是让大家参与进来，一起来找茬。

无论你是觉得代码写得不够漂亮、有 bug，或是文档小到有错别字，大到有原理性的描述错误，都可以直接通过项目的 github 提交给我们，我们会视具体的情况进行审核处理加入到主线或特性分支。

野火的每个项目的《关于本项目》文件中都包含有项目的 git 仓库地址，通过 git 提交 **issue**、**pull request** 的方式可参与到项目。

我们主要使用 github 仓库进行维护和更新，gitee 仓库用于方便用户快速下载，我们一般只在 gitee 上同步 github 的 master 分支，所以参与项目时请尽量使用 github。

3.1 轻度参与，提交 issue

如果你发现了一些 bug 或小问题，而自己暂时没时间帮助我们修改的话，可以直接在项目的 github 中提交 issue 告诉我们，我们会适时进行处理。

3.2 深度参与，提交 pull request

如果你想深度参与本项目，可以先克隆或 fork 项目的仓库到本地，修改编辑后向我们提交 pull request，我们会审核处理。

CHAPTER 4

TODO 和悬赏任务

本页记录待改进的事项，以后会在此页发布任务清单，完成部分任务可获得 **奖励**，敬请期待！

如果您想参与到项目，可按《*快速参与本项目（提交 bug 或文档修改）*》的说明操作。

4.1 文档任务

此处写具体的任务列表

- PDF 与 readthedocs
- 详细写如何贡献（git issue 和 commit）

4.2 代码开发任务

此处写具体的任务列表

CHAPTER 5

sphinx 的安装与使用

5.1 安装 sphinx

sphinx 官方安装说明：http://www.sphinx-doc.org/en/master/usage/installation.html

readthedoc 官方说明：https://docs.readthedocs.io/en/stable/intro/getting-started-with-sphinx.html

总的来说步骤如下：

- 安装 python3
- 通过 python3 安装 sphinx 包
- 根据项目要求安装项目的 requirements.txt 里的软件包

5.1.1 安装 python3

windows 直接到 python 官网下载安装

Ubuntu 下使用如下指令安装：

```
sudo apt install python3
```

5.1.2 安装 sphinx

windows 下使用如下指令安装：

```
py -3 -m pip install sphinx
# 国内用户推荐使用清华源安装，使用-i 指定源
py -3 -m pip install -i https://pypi.tuna.tsinghua.edu.cn/simple sphinx
```

Ubuntu 下使用如下指令安装

```
python3 -m pip install sphinx
# 国内用户推荐使用清华源安装，使用-i 指定源
python3 -m pip install -i https://pypi.tuna.tsinghua.edu.cn/simple sphinx

#Ubuntu 下可能还需要安装如下软件包
apt-get install python3-sphinx
```

5.2 使用模板创建 sphinx 文档

本项目的 sphinx 已适配好支持 markdown、默认的 readthedoc 主题等内容。

如果是要编写新的书籍，推荐直接使用本项目文档作为模版，修改 conf.py 文件的配置即可改变项目的名字、主题之类的内容。

5.2.1 下载项目源码

先从 github 下载本项目源代码：

```
git clone git@github.com:Embdefire/ebf_contribute_guide.git
```

下载完成后，可看到本项目的结构大致如下：

```
  README                  说明文档或软链接至 documentation 中的说明，方便 github 阅读
  base_code               配套代码
  documentation           配套文档
      faq                 存储常见问题的文档
      about_us.rst        关于我们
      _build              文档编译输出目录
      conf.py             sphinx 配置文件
      contribute          如何参与项目，贡献与投稿的说明
      foreword.rst        前言
      index.rst           目录
      make.bat
      Makefile
```

(下页继续)

(续上页)

```
    media                   文档中使用的图片文件
    requirements.txt        文档的 python 依赖
    _static
    _templates
    TODO.rst                待完成的内容，发布的任务列表
```

特别内容说明如下：

- **base_code**： 该目录通常存放项目的代码，
- **documentation**： 该目录通常存放项目的文档内容，如我们的 rst、markdown 文档。

5.2.2 安装 python 依赖包

使用时，要先根据项目的 **documentation/requirements.txt** 安装依赖的 python 包。

```
# 切换至 requirements.txt 文件的同级目录
cd documentation

#requirements.txt 文件的同级目录下执行后面的命令

#Windows 指令，推荐使用 powershell 运行
py -3 -m pip install -r requirements.txt
#Ubuntu 指令
python3 -m pip install -r  requirements.txt

# 国内用户推荐使用清华源安装
#Windows 指令，推荐使用 powershell 运行
py -3 -m pip install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple
#Ubuntu 指令
python3 -m pip install -r  requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple
```

5.2.3 编写文档

安装好后，在项目的文档目录中添加 rst 或 markdown 文件即可，建议使用 vs code 来编写文档，它非常方便预览文档，可参考《*使用 vscode 编写 rst 文件*》来搭建 vscode 预览 sphinx 文档的环境。

编写文档时的语法和规范可参考以下内容：

- 《*ReST 基础语法*》
- 《*野火 sphinx 文档规范*》
- 《*Markdown 示例*》

5.3 编译文档

如果使用了 vscode 的 rst 插件，可以直接保存 rst 文件后它会自动编译并可预览。

也可以手动编译，到文档源码所在的 makefile 目录，执行如下命令：

```
# 在文档的 makefile 目录下执行

#Windows 指令
make.bat  html
#Ubuntu 指令
make html
```

在设定的 build 或 _build 的 html 目录下会生成静态的 html 文件。可直接使用这些静态的 html 文件制作网站。

5.4 在本地预览文档

vscode 插件预览有时不够完整，可以在本地开启一个 python 服务器来预览。进入到生成的 _build/html 目录，运行如下指令：

```
# 在生成的 html 目录执行如下指令
#Windows
py -3 -m http.server  8000

#Ubuntu
python3 -m http.server 8000
```

运行指令后，在浏览器中打开 http://localhost:8000 即可查看生成的静态网页。

5.5 允许其它主机预览文档

类似地，若要允许其它主机访问自己的文档，执行如下命令即可

```
# 在生成的 html 目录执行如下指令
#Windows
py -3 -m http.server --bind 0.0.0.0 8000

#Ubuntu
python3 -m http.server --bind 0.0.0.0 8000
```

运行指令后，在浏览器中打开 http://主机 IP:8000 即可查看生成的静态网页。

5.6 清除编译输出

有时 html 文件不会完全达到我们修改 rst 后的效果，这可能是因为之前的旧文件影响，这时可以先清除编译输出再重新编译。

```
# 清除编译输出

#Windows 指令
make.bat  clean
#Ubuntu 指令
make clean

# 重新编译

#Windows 指令
make.bat  html
#Ubuntu 指令
make html
```

5.7 创建全新的 sphinx 文档

若不想使用本工程模版，可以使用如下指令创建全新的文档。

```
sphinx-quickstart
```

按照提示回答问题即可。推荐使用默认的 _build 目录，与 vscode 保持一致。其中提示语言时可以使用这个中文代码：zh_CN

sphinx 默认不支持 markdown 语法，要支持的话请参考本模版的 conf.py 文件配置。

CHAPTER 6

使用 vscode 编写 rst 文件

使用 vscode 编写 sphinx 文档非常方便，可以即时预览和自动编译。

rst vscode 插件说明页：https://docs.restructuredtext.net/index.html

另外，vscode 默认支持 markdown 单个文件的预览。

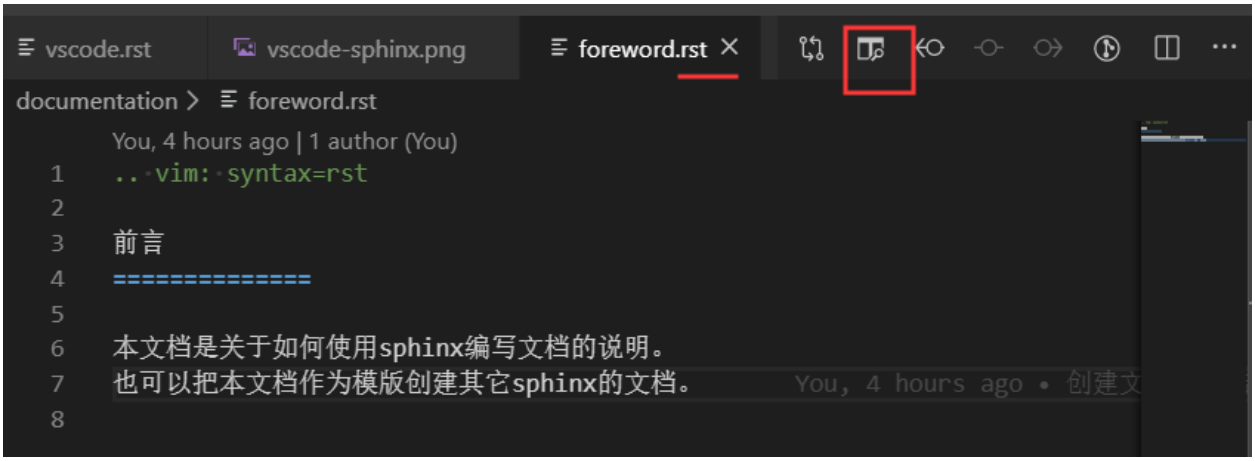
6.1 reStructuredText 插件

需要安装的 vscode 插件：

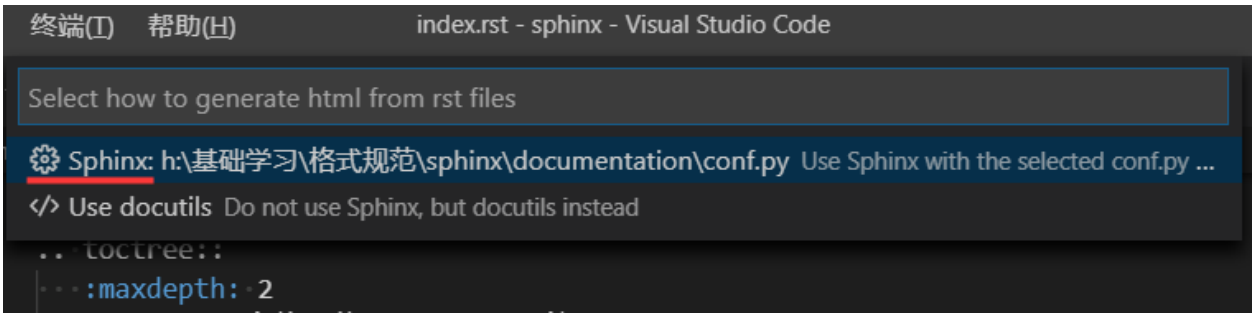
- python
- reStructuredText

安装 python 插件后，要选择使用 python3 的解释器，不要使用 python2。

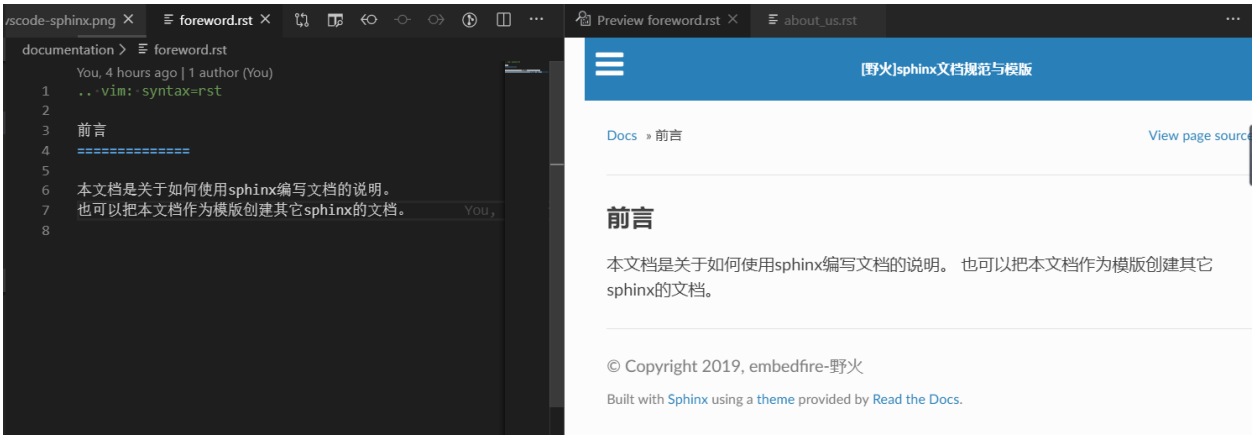
安装 reStructuredText 插件后，打开 rst 文件，点击预览按钮。



会提示使用 sphinx 还是 doctuil 工具，我们选择使用 sphinx。



若配置正确，稍等一会即可在右侧看到预览效果。



6.2 Table Formater 插件

使用 Table Formater 可以方便格式化表格，支持 markdown 和 rst https://marketplace.visualstudio.com/items?itemName=shuworks.vscode-table-formatter

插件名：Table Formatter

安装后通过 Ctrl-Shift-P 调用 Table: Format Current 或 Table: Format All

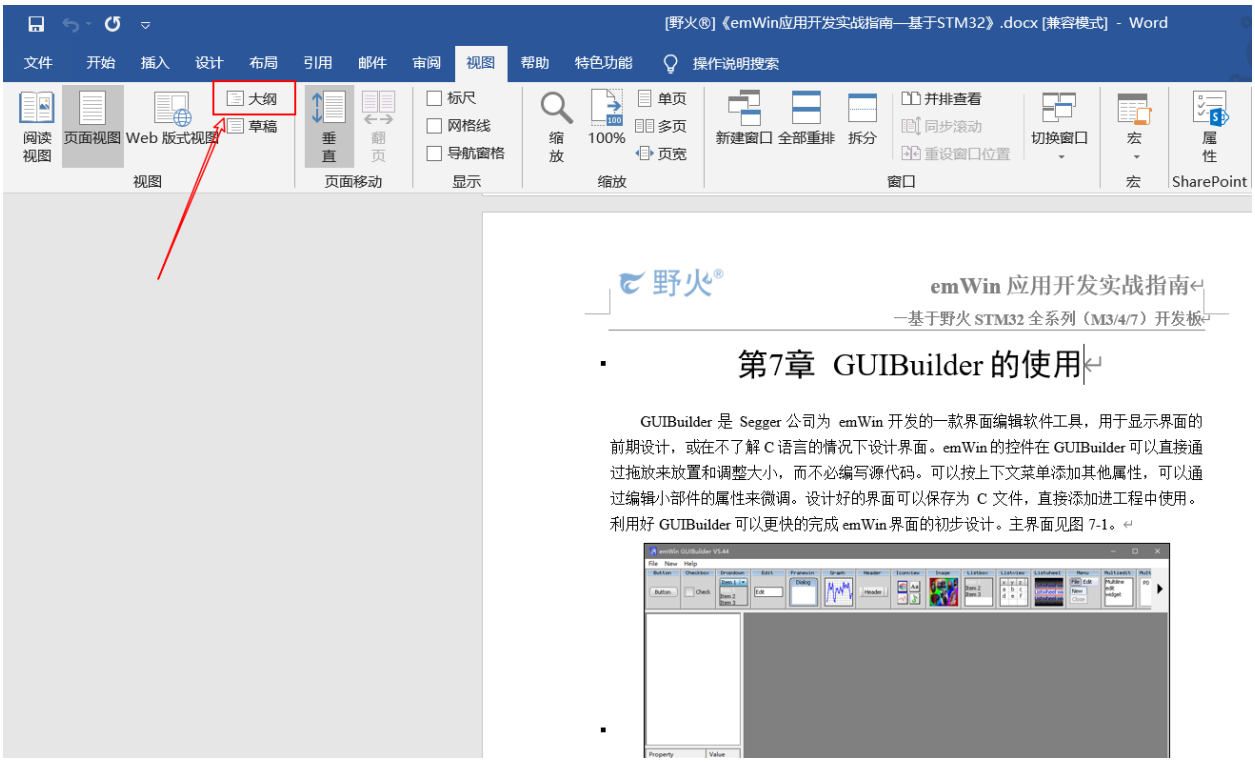
CHAPTER 7

docx 批量转 rst 方法

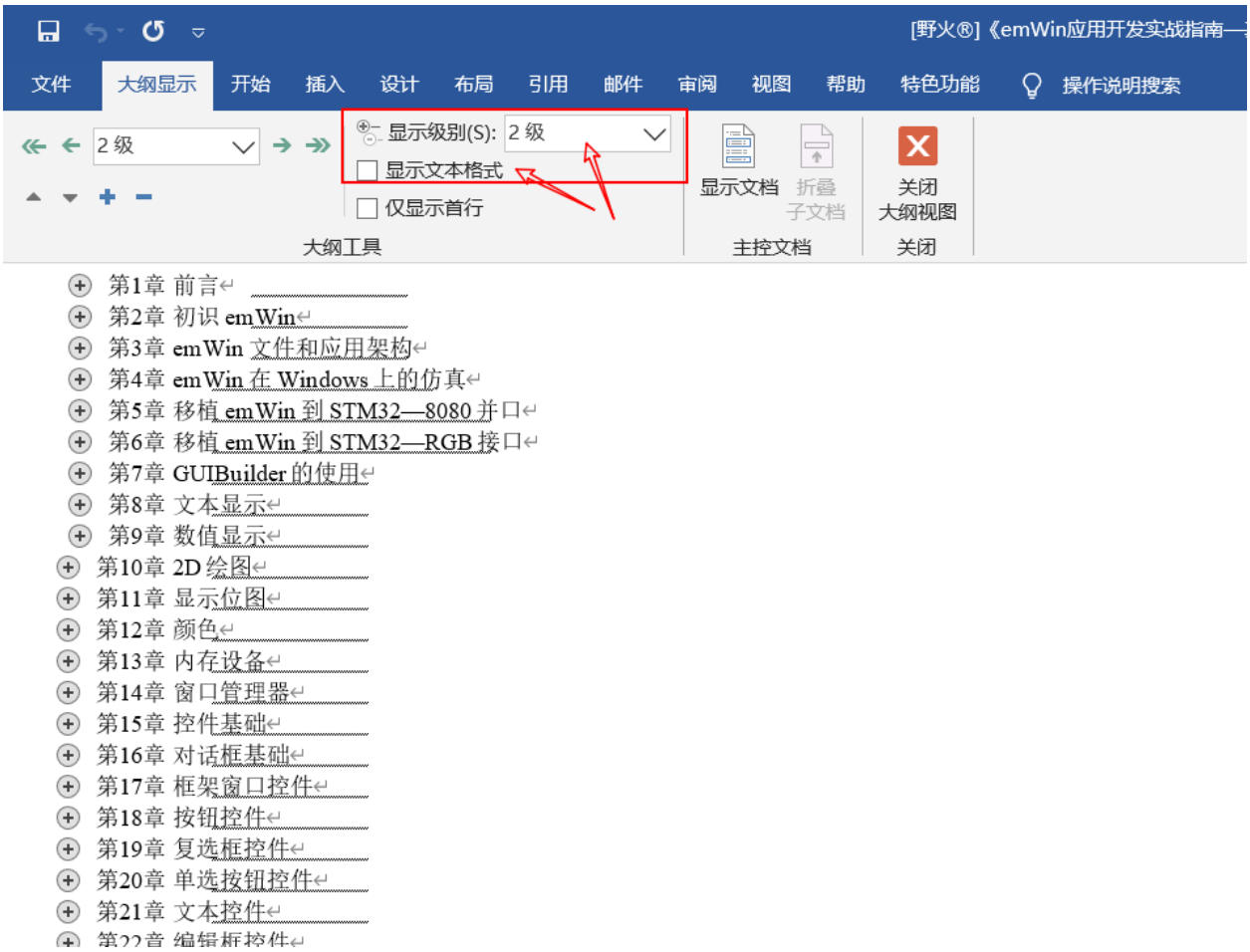
7.1 按章分割 word 文档

1. 将所需转换的文档放入一个新的文件夹中

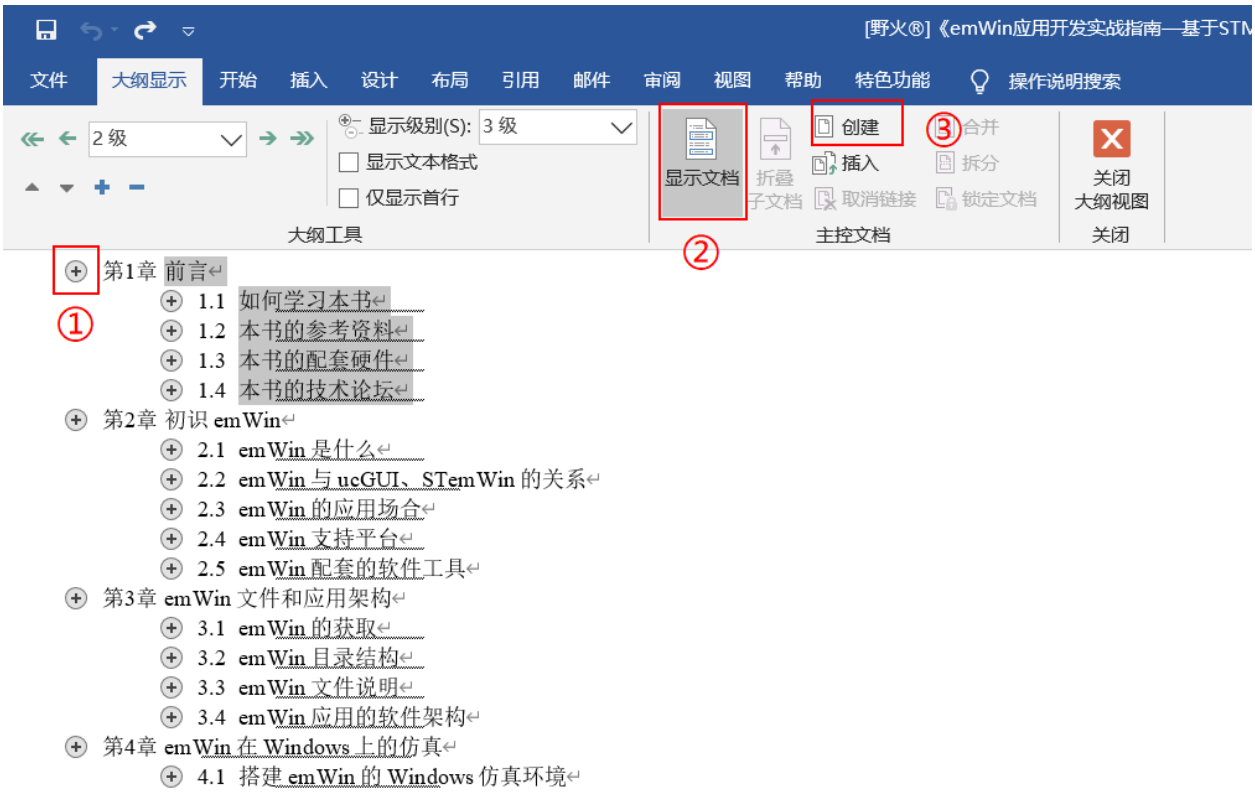
2. 打开文档，进入大纲视图



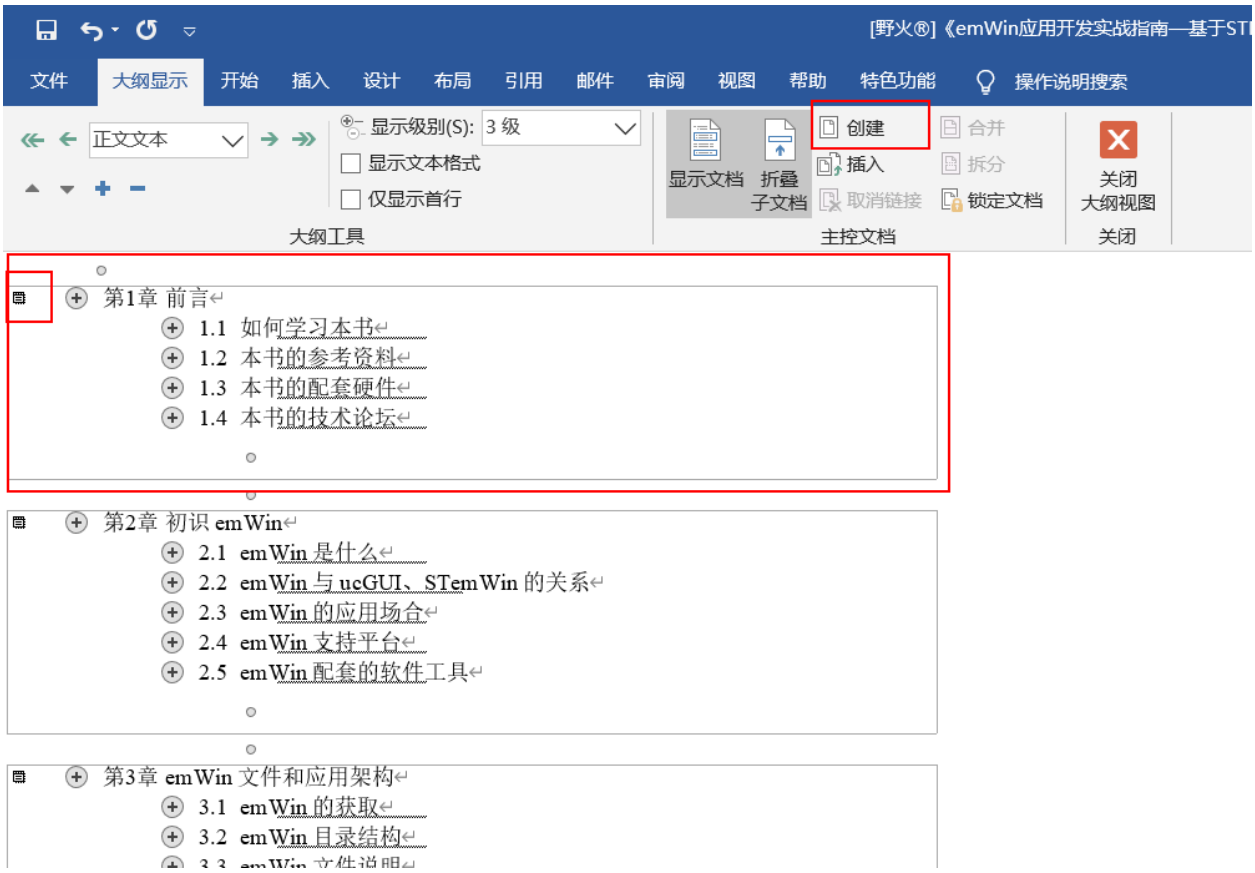
3. 合理选择显示级别，取消显示文本格式的选项后将显示本文档所有的章节名



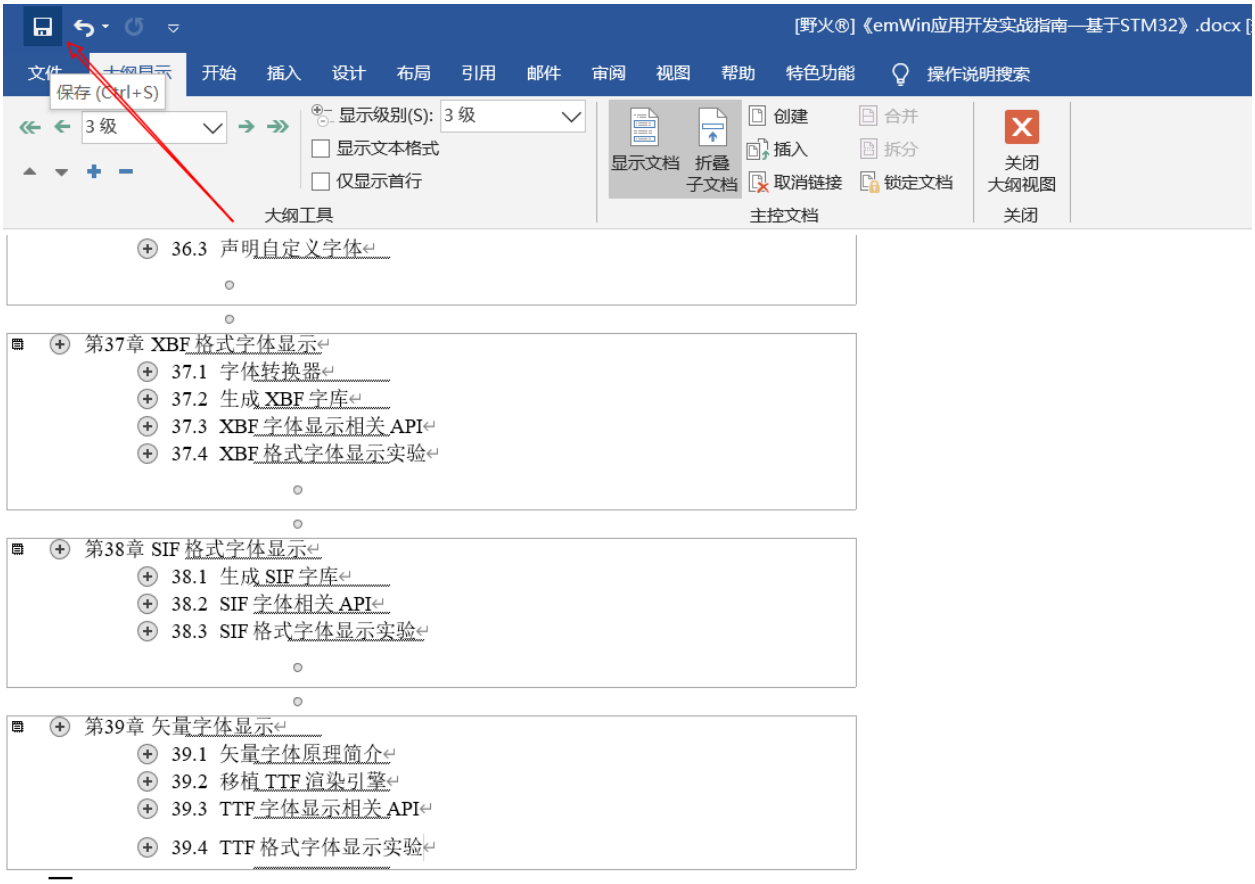
4. 点击第 n 章前的加号按钮，选中本章，点击显示文档按钮，出现创建选项



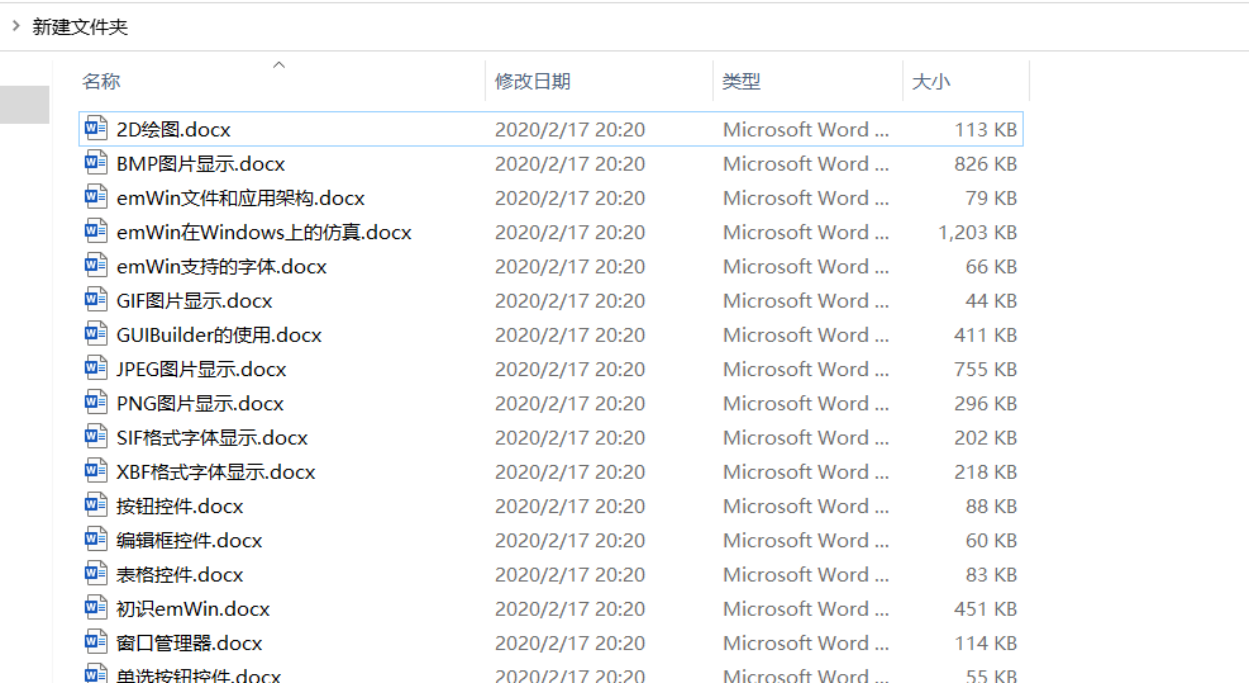
5. 点击创建按钮之后如图所示



6. 所有章都按上述方法操作，点击保存

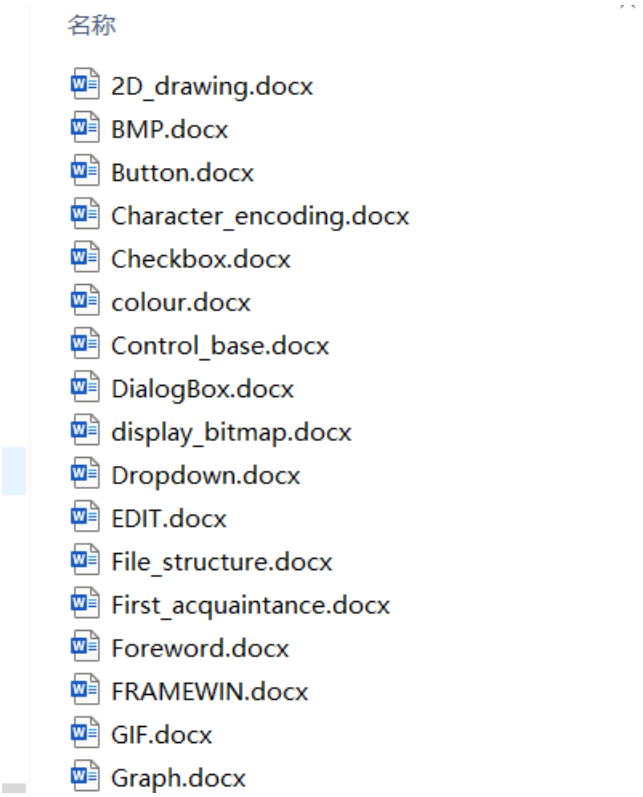


7. 文件夹中就出现了分割好的单独 docx 格式的文件，删除原来整本的文档



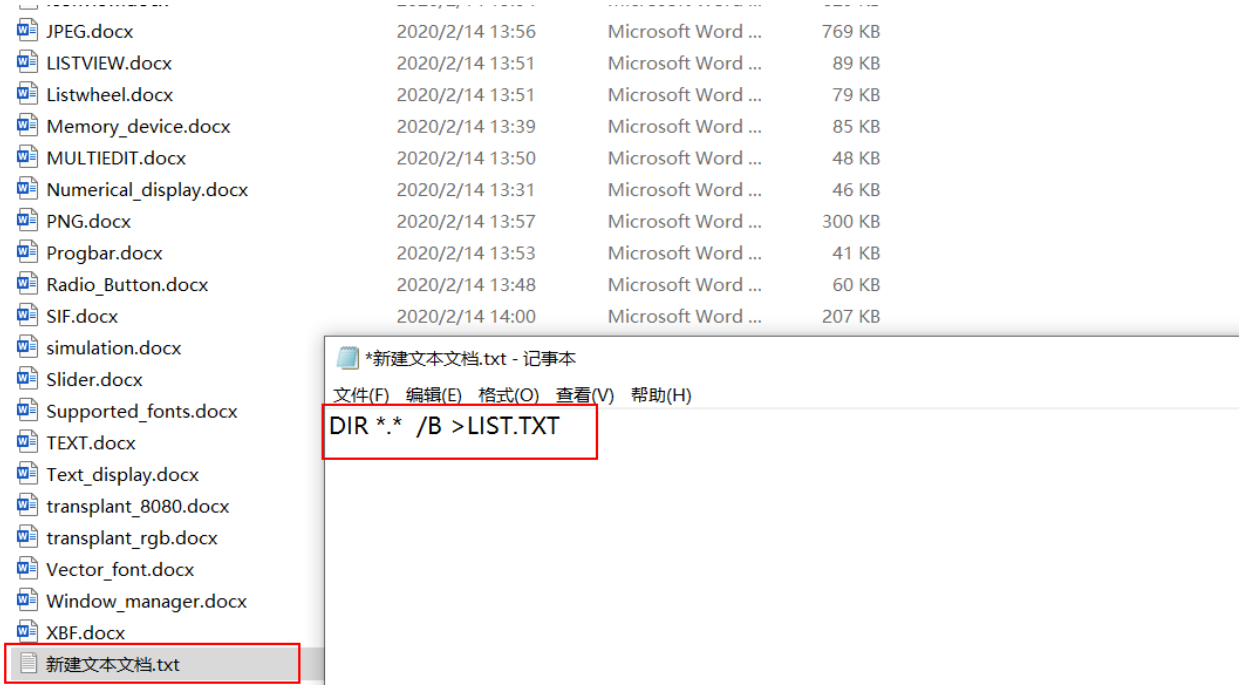
7.2 批量将分割后的 docx 转换为 rst

1. 将分割后的文件改成对应的英文名以方便转换

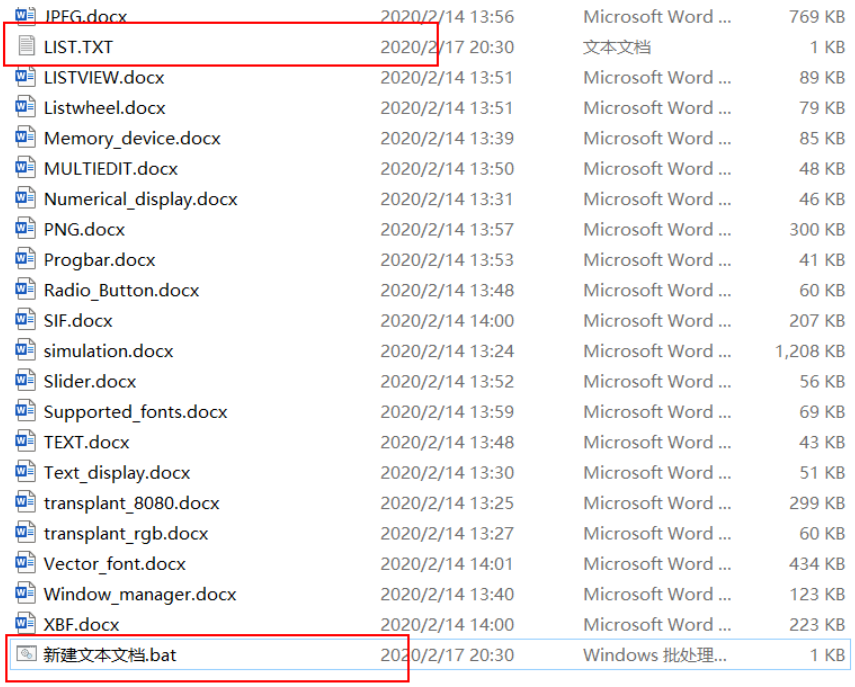


2. 新建一个文本文档，内容如下，获取当前列表中的文件名

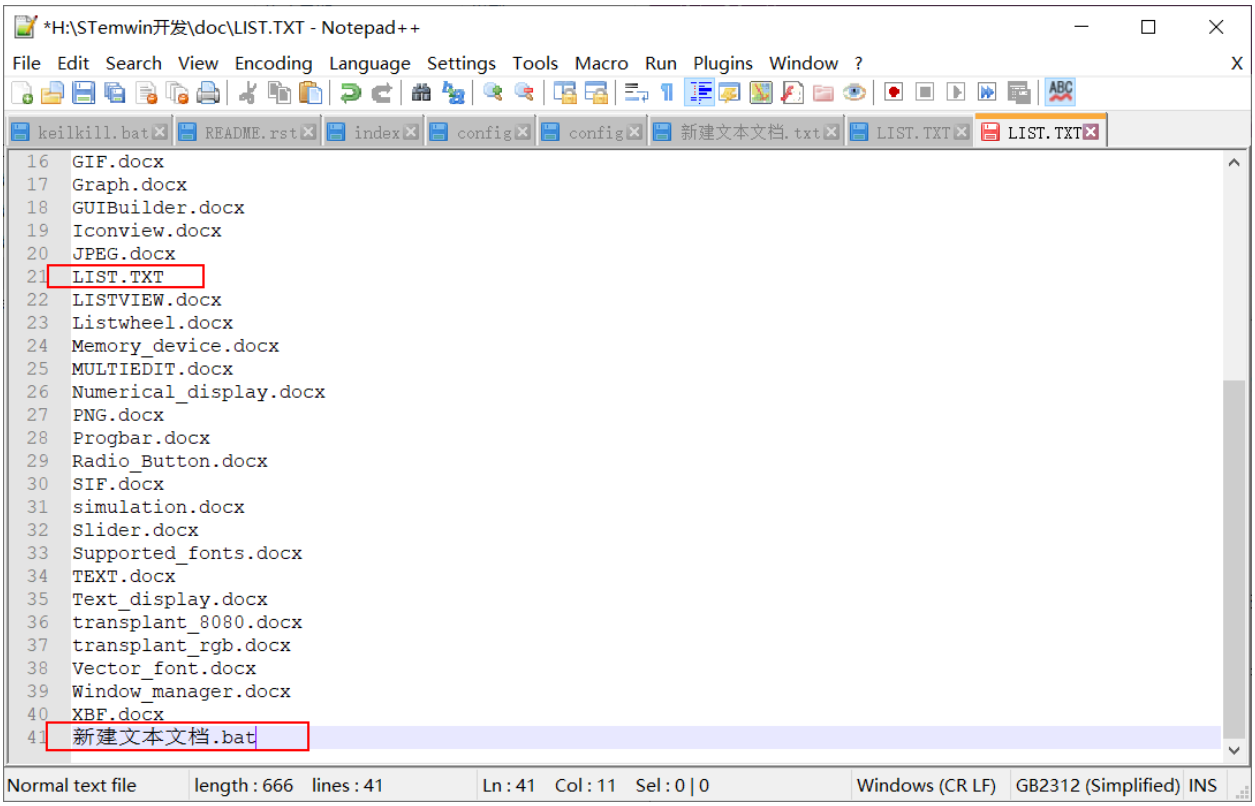
```
DIR *.* /B >LIST.TXT
```



3. 将新建文本文档的后缀名改为.bat 后双击运行得到了 LIST.TXT ，

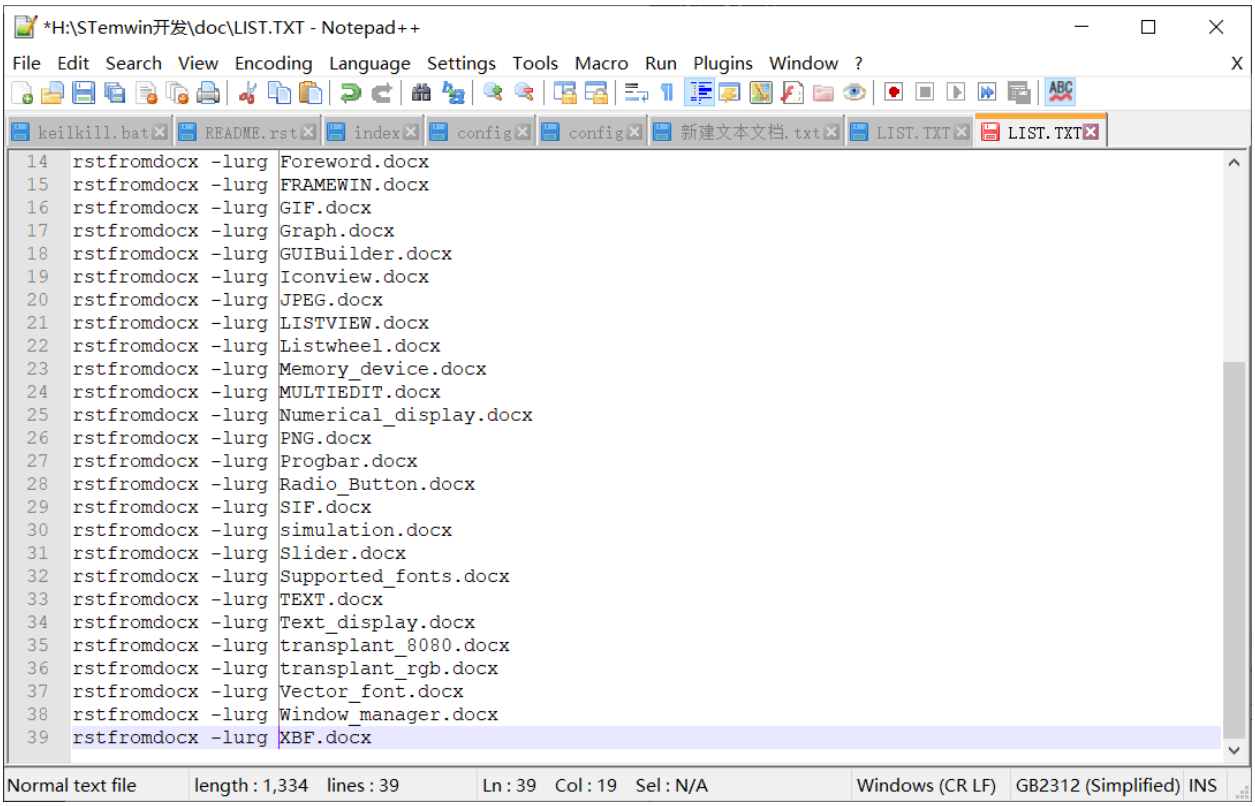


4. 用 notepad++ 打开 LIST.TXT, 删除图中的两行

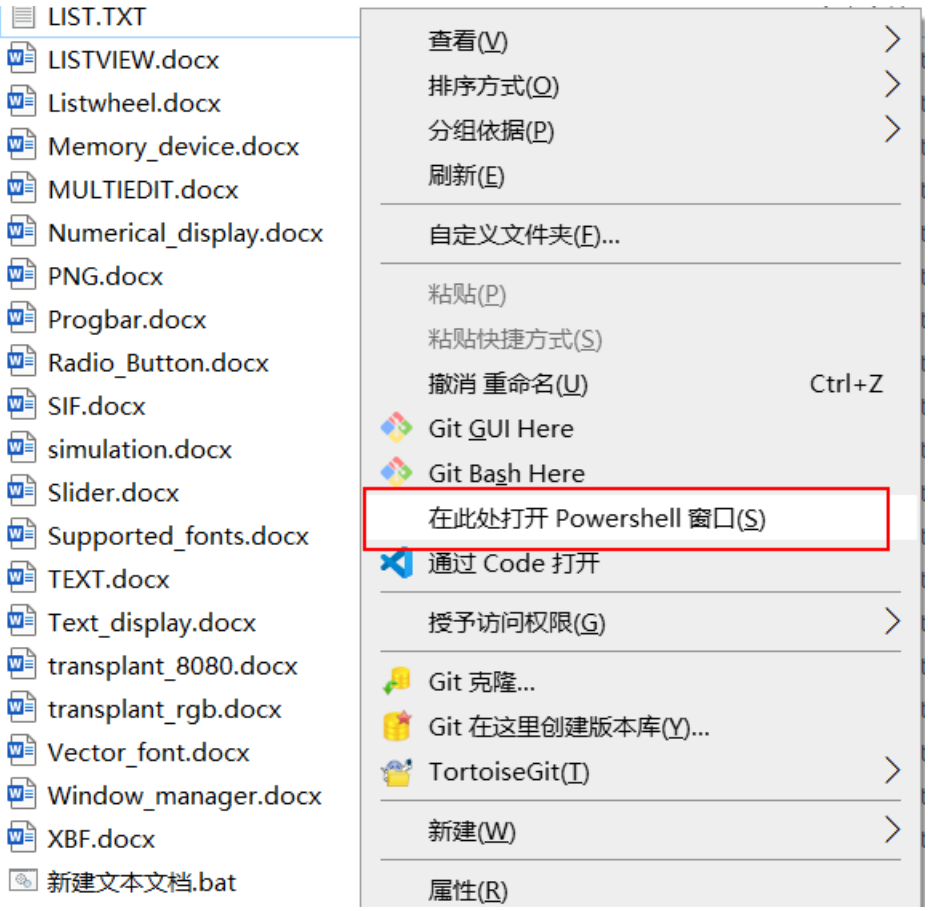


5. 鼠标：alt+ 鼠标左键选择中所有的行。键盘：alt+shift+ 方向鍵将位置调整到行首。输入 rstfromdocx -lurg 后保存。

```
rstfromdocx -lurg
```



6. 在文档所在的文件夹按住 shift 键和点击鼠标右键，选中打开 powershell，



7. 全选上面编辑好的文档列表然后复制到 powershell 中，鼠标右键复制，文档开始转换

```
Windows PowerShell
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg 2D_drawing.docx
D:\Users\H7819\Desktop\新建文件夹\2D_drawing\2D_drawing.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg BMP.docx
D:\Users\H7819\Desktop\新建文件夹\BMP\BMP.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Button.docx
D:\Users\H7819\Desktop\新建文件夹\Button\Button.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Character_encoding.docx
D:\Users\H7819\Desktop\新建文件夹\Character_encoding\Character_encoding.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Checkbox.docx
D:\Users\H7819\Desktop\新建文件夹\Checkbox\Checkbox.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg colour.docx
D:\Users\H7819\Desktop\新建文件夹\colour\colour.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Control_base.docx
D:\Users\H7819\Desktop\新建文件夹\Control_base\Control_base.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg DialogBox.docx
D:\Users\H7819\Desktop\新建文件夹\DialogBox\DialogBox.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg display_bitmap.docx
D:\Users\H7819\Desktop\新建文件夹\display_bitmap\display_bitmap.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Dropdown.docx
D:\Users\H7819\Desktop\新建文件夹\Dropdown\Dropdown.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg EDIT.docx
D:\Users\H7819\Desktop\新建文件夹\EDIT\EDIT.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg File_structure.docx
D:\Users\H7819\Desktop\新建文件夹\File_structure\File_structure.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg First_acquaintance.docx
D:\Users\H7819\Desktop\新建文件夹\First_acquaintance\First_acquaintance.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Foreword.docx
D:\Users\H7819\Desktop\新建文件夹\Foreword\Foreword.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg FRAMEWIN.docx
D:\Users\H7819\Desktop\新建文件夹\FRAMEWIN\FRAMEWIN.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg GIF.docx
D:\Users\H7819\Desktop\新建文件夹\GIF\GIF.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Graph.docx
D:\Users\H7819\Desktop\新建文件夹\Graph\Graph.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg GUIBuilder.docx
D:\Users\H7819\Desktop\新建文件夹\GUIBuilder\GUIBuilder.rest
PS D:\Users\H7819\Desktop\新建文件夹> rstfromdocx -lurg Iconview.docx
D:\Users\H7819\Desktop\新建文件夹\Iconview\Iconview.rest
```

8. 完成后就得到了转换好的文件，将转换好的文件复制到一个新文件夹中，防止接下来的操作失败，注意备份

7.3 批量将.rest 修改为.rst 并删除不需要的文件

1. 在新文件夹中新建一个.bat 文件，并复制以下内容保存后运行

```
del *.py /s
del index.rest /s
del Makefile /s

for /f "tokens=* delims=" %%i in ('dir /b /a-d /s "*.rest"') do (move "%%i" "%%~dpi./../
↪")
for /f "tokens=* delims=" %%i in ('dir /b /a-d /s "*.png"') do (move "%%i" "%%~dpi./../")
for /f "tokens=* delims=" %%i in ('dir /b /a-d /s "*.jpeg"') do (move "%%i" "%%~dpi./../
↪")
```

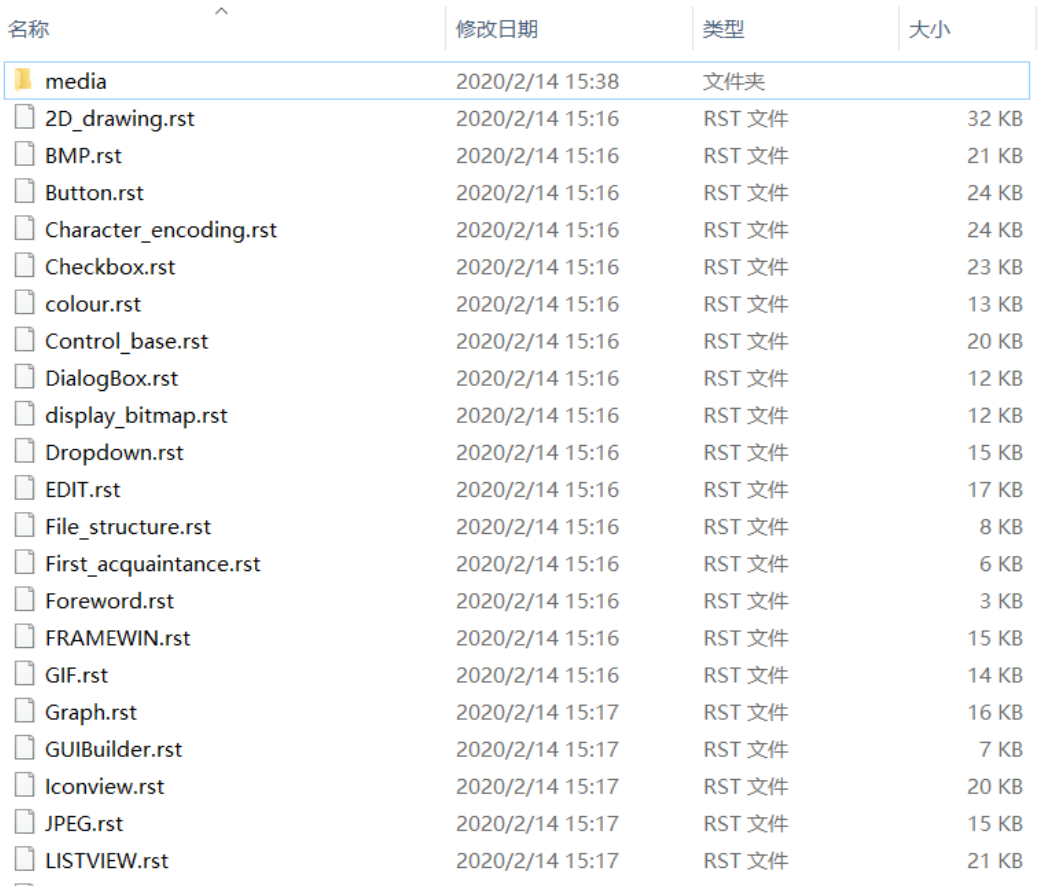
(下页继续)

(续上页)

```
for /f "tokens=* delims=" %%i in ('dir /b /a-d /s "*.jpg"') do (move "%%i" "%%~dpi./../")
for /f "tokens=* delims=" %%i in ('dir /b /a-d /s "*.bmp"') do (move "%%i" "%%~dpi./../")

echo.
echo 正在删除当前目录及子目录中所有的空文件夹，请稍后......
echo ------------------------------------------------------------
cd. > listnull.txt
for /f "delims=" %%i in ('dir /ad /b /s') do (
dir /b "%%i" | findstr .>nul || echo %%i >> listnull.txt
)

set /a sum=0
for /f "tokens=*" %%i in (listnull.txt) do (
rd /q "%%i"
echo 成功删除空目录: %%i
set /a sum=sum+1
)

echo ------------------------------------------------------------
echo 共成功删除%cd% 目录及其子目录下%sum% 个空文件夹!
echo.
set sum=

ren *.rest *.rst

del listnull.txt

exit
```

上述代码的作用是删除不需要的文件和空文件夹，并将图片文件移动到上一级目录下，然后将 rest 更名为 rst

2. 新建一个 media 文件夹，并将存放图片的文件夹移动至 media 文件夹，至此，批量转换完成



名称	修改日期	类型	大小
media	2020/2/14 15:38	文件夹	
2D_drawing.rst	2020/2/14 15:16	RST 文件	32 KB
BMP.rst	2020/2/14 15:16	RST 文件	21 KB
Button.rst	2020/2/14 15:16	RST 文件	24 KB
Character_encoding.rst	2020/2/14 15:16	RST 文件	24 KB
Checkbox.rst	2020/2/14 15:16	RST 文件	23 KB
colour.rst	2020/2/14 15:16	RST 文件	13 KB
Control_base.rst	2020/2/14 15:16	RST 文件	20 KB
DialogBox.rst	2020/2/14 15:16	RST 文件	12 KB
display_bitmap.rst	2020/2/14 15:16	RST 文件	12 KB
Dropdown.rst	2020/2/14 15:16	RST 文件	15 KB
EDIT.rst	2020/2/14 15:16	RST 文件	17 KB
File_structure.rst	2020/2/14 15:16	RST 文件	8 KB
First_acquaintance.rst	2020/2/14 15:16	RST 文件	6 KB
Foreword.rst	2020/2/14 15:16	RST 文件	3 KB
FRAMEWIN.rst	2020/2/14 15:16	RST 文件	15 KB
GIF.rst	2020/2/14 15:16	RST 文件	14 KB
Graph.rst	2020/2/14 15:17	RST 文件	16 KB
GUIBuilder.rst	2020/2/14 15:17	RST 文件	7 KB
Iconview.rst	2020/2/14 15:17	RST 文件	20 KB
JPEG.rst	2020/2/14 15:17	RST 文件	15 KB
LISTVIEW.rst	2020/2/14 15:17	RST 文件	21 KB

CHAPTER 8

ReST 基础语法

rst 的基础语法和 markdown 差不多，

可以使用这个在线的 rst 编辑器了解相关语法：rst 在线编辑器

- sphinx 语法官网：http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
- restruct 语法官网：http://docutils.sourceforge.net/rst.html

下面是常用语法：

语法：

```
一级标题
==============================

二级标题
~~~~~~~~~~~~~~~

三级标题
------------------

四级标题
^^^^^^^^^^^^^^^^^^^^^

五级标题
""""""""""""""""""
```

(下页继续)

(续上页)

```
六级标题
*****************

** 强调 **

* 斜体 *

``monospace，会变色，具体作用不清楚``

无序列表
--------------------------
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh

有序列表
------------------------
支持数字、大小写字母和罗马数字

1. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh

a. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
#. hhhhhhhh
```

效果：

强调

斜体

`monospace，会变色，具体作用不清楚`

8.1 无序列表

- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh
- hhhhhhhh

8.2 有序列表

支持数字、大小写字母和罗马数字

1. hhhhhhhh
2. hhhhhhhh
3. hhhhhhhh
4. hhhhhhhh
5. hhhhhhhh
6. hhhhhhhh

a. hhhhhhhh

b. hhhhhhhh

c. hhhhhhhh

d. hhhhhhhh

e. hhhhhhhh

f. hhhhhhhh

8.3 表格

表格语法说明：http://docutils.sourceforge.net/docs/ref/rst/directives.html#csv-table

推荐使用列表式表格，修改比较方便

8.3.1 列表式表格

语法：

```
.. list-table:: Frozen Delights!
    :widths: 15 10 30
    :header-rows: 1

    * - Treat
      - Quantity
      - Description
    * - Albatross
      - 2.99
      - On a stick!
    * - Crunchy Frog
      - 1.49
      - If we took the bones out, it wouldn't be
        crunchy, now would it?
    * - Gannet Ripple
      - 1.99
      - On a stick!
```

效果：

表 1: Frozen Delights!

Treat	Quantity	Description
Albatross	2.99	On a stick!
Crunchy Frog	1.49	If we took the bones out, it wouldn’ t be crunchy, now would it?
Gannet Ripple	1.99	On a stick!

8.3.2 普通表格

表格使用 == 号制作

语法示例：

```
=====  =====  =======
A      B      A and B
=====  =====  =======
False  False  False
```

(下页继续)

(续上页)

```
True   False  False
False  True   False
True   True   True
=====  =====  =======
```

效果:

A	B	A and B
False	False	False
True	False	False
False	True	False
True	True	True

在 vscode 可安装插件方便格式化表格: https://marketplace.visualstudio.com/items?itemName=shuworks.vscode-table-formatter

安装后通过 Ctrl-Shift-P 调用 Table: Format Current 或 Table: Format All

8.3.3 CSV 表格

使用 CSV 编写

```
.. csv-table:: Frozen Delights!
    :header: "Treat", "Quantity", "Description"
    :widths: 15, 10, 30

    "Albatross", 2.99, "On a stick!"
    "Crunchy Frog", 1.49, "If we took the bones out, it wouldn't be
    crunchy, now would it?"
    "Gannet Ripple", 1.99, "On a stick!"
```

效果:

表 2: Frozen Delights!

Treat	Quantity	Description
Albatross	2.99	On a stick!
Crunchy Frog	1.49	If we took the bones out, it wouldn’ t be crunchy, now would it?
Gannet Ripple	1.99	On a stick!

CHAPTER 9

超链接与文件引用

参考说明：http://www.sphinx-doc.org/en/master/usage/restructuredtext/roles.html#ref-role

9.1 超链接

语法：

```
直接嵌入网址：
`野火公司官网 <http://www.embedfire.com>`_

使用引用的方式把具体网址定义在其它地方： `野火公司官网 `_

.. _ 野火公司官网: http://www.embedfire.com
```

效果：

直接嵌入网址：

野火公司官网

使用引用的方式把具体网址定义在其它地方：野火公司官网

9.2 引用

9.2.1 引用图片、表格

在图片、表格上面加一个引用标签，然后通过 ref 指令引用。语法：

```
.. _my-reference-label 支持中文:

.. figure: ../media/rest-syntax/pic-video/logo.png
   :alt: 野火 logo
   :align: center

引用方式 :ref:`my-reference-label 支持中文 ` 。
```

效果：

图 1: 必须写这个图片名

引用方式必须写这个图片名 。

表格引用：

```
.. _ 拨码开关启动配置表:

.. table:: 拨码开关启动配置表

==== ====== ========== ==== == ===
编号 名称   NAND FLASH eMMC SD USB
==== ====== ========== ==== == ===
1    MODE0  0          0    0  1
2    MODE1  1          1    1  0
3    CFG1-4 1          0    0  X
4    CFG1-5 0          1    0  X
5    CFG1-6 0          1    1  X
6    CFG1-7 1          0    0  X
7    CFG2-3 0          1    0  X
8    CFG2-5 0          0    1  X
```

(下页继续)

(续上页)

```
==== ====== ========== ==== == ===


引用示例 :ref:`拨码开关启动配置表 ` 。
自定义名称引用 :ref:`自定义名称 < 拨码开关启动配置表>` 。
```

效果:

表 1: 拨码开关启动配置表

编号	名称	NAND FLASH	eMMC	SD	USB
1	MODE0	0	0	0	1
2	MODE1	1	1	1	0
3	CFG1-4	1	0	0	X
4	CFG1-5	0	1	0	X
5	CFG1-6	0	1	1	X
6	CFG1-7	1	0	0	X
7	CFG2-3	0	1	0	X
8	CFG2-5	0	0	1	X

引用示例拨码开关启动配置表 。自定义名称引用自定义名称 。

9.2.2 引用文档

语法:

```
自定义引用文字


:doc:`引用本地的其它 rst 文档,rst 后缀要省略, 如 about_us <../about_us>`

使用标题文字
:doc:`../about_us`
```

效果:

自定义引用文字

引用本地的其它 rst 文档,rst 后缀要省略, 如 about_us

使用标题文字关于野火

9.2.3 使用标签引用文档

要在被引用的文件头定义标签，如 about_us.rst 文件头写“about_embedfire”的标签，具体请查看该文档的源码

语法：

```
:ref:`自定义链接文字 <about_embedfire>`

:ref:`about_embedfire`
```

效果：

about_embedfire

关于野火

若要跨文档引用标题，可以使用自动切片扩展插件，它的使用方式如下：

某个文档：

```
=============
Some Document
=============



Internal Headline
=================
```

然后在另一个文档引用：

```
===============
Some Other Doc
===============



A link-  :ref:`Internal Headline`
```

此扩展程序是内置的，因此您只需编辑即可 conf.py

```
extensions = [
    .
    . other
    . extensions
    . already
```

(下页继续)

(续上页)

```
    . listed
    .
    'sphinx.ext.autosectionlabel',
]
```

您唯一需要注意的是，现在您无法在整个文档集合中复制内部标题。

9.2.4 引用非 rst 文档

会呈现出点击后下载文件的效果。注意这种引用方式在生成 pdf 文件时链接会无效。

语法：

```
:download:`引用非 rst 的本地文档 <../requirements.txt>`.
```

效果：

引用非 rst 的本地文档.

CHAPTER 10

代码高亮

参考说明：http://www.sphinx-doc.org/en/master/usage/restructuredtext/directives.html#toctree-directive

支持的高亮语言：https://pygments.org/docs/lexers#lexers-for-various-shells

10.1 快速定义代码块

使用简便的预定义高亮语法高亮缩进，默认不带语言说明的都使用 highlight 定义的语言高亮，然后可以直接使用“::”两个冒号代替“code-block”指令快速定义其它代码块，直到下一个 highlight 指令，才会改变语言：

```
.. highlight:: sh

此指令后如下的“::”定义的块都会以 sh 语法进行高亮，直到遇到下一条 highlight 指令。

::

   # 此命令在主机执行
   sudo apt install python
   echo "helloworld,this is a script test!"
```

效果：

```
# 此命令在主机执行
sudo apt install python
echo "helloworld,this is a script test!"
```

10.2 code-block 代码高亮

10.2.1 shell 高亮测试

高亮语法:

```
.. code-block:: sh
   :caption: test
   :name: test333
   :emphasize-lines: 2
   :linenos:

   # 此命令在主机执行
   sudo apt install python
   echo "helloworld,this is a script test!"
```

效果:

列表 1: sh test

```
1 # 此命令在主机执行
2 sudo apt install python
3 echo "helloworld,this is a script test!"
```

10.2.2 C 高亮测试

语法:

```
.. code-block:: c
   :caption: c test
   :emphasize-lines: 4,5
   :linenos:

   #include <stdio.h>
```

(下页继续)

(续上页)

```
   int main()
   {
      printf("hello, world! This is a C program.\n");
      for(int i=0;i<10;i++ ){
         printf("output i=%d\n",i);
      }

      return 0;
   }
```

效果:

列表 2: c test

```c
#include <stdio.h>

int main()
{
   printf("hello, world! This is a C program.\n");
   for(int i=0;i<10;i++ ){
      printf("output i=%d\n",i);
   }

   return 0;
}
```

10.2.3 verilog 高亮测试

语法:

使用 verilog 或 v 进行高亮

```
.. code-block:: v
   :caption: verilog test
   :emphasize-lines: 4,5
   :linenos:

   module  key_filter
   #(
      parameter CNT_MAX = 20'd999_999 //计数器计数最大值
   )
```

(下页继续)

(续上页)

```
   (
      input   wire    sys_clk     ,   //系统时钟 50Mhz
      input   wire    sys_rst_n   ,   //全局复位
      input   wire    key_in      ,   //按键输入信号

      output  reg     key_flag        //key_flag 为 1 时表示消抖后检测到按键被按下
                                       //key_flag 为 0 时表示没有检测到按键被按下
   );
```

效果:

列表 3: verilog test

```
module  key_filter
#(
   parameter CNT_MAX = 20'd999_999 //计数器计数最大值
)
(
   input   wire    sys_clk     ,   //系统时钟 50Mhz
   input   wire    sys_rst_n   ,   //全局复位
   input   wire    key_in      ,   //按键输入信号

   output  reg     key_flag        //key_flag 为 1 时表示消抖后检测到按键被按下
                                   //key_flag 为 0 时表示没有检测到按键被按下
);
```

10.3 literalinclude 直接嵌入本地文件并高亮

10.3.1 嵌入整个文件

直接嵌入文件，包含标题、代码语言、高亮、带编号以及名称方便引用。

插入 C 代码

```
.. literalinclude:: ../../base_code/hello.c
   :caption: ../../base_code/hello.c
   :language: c
   :emphasize-lines: 5,7-12
```

(下页继续)

(续上页)

```
   :linenos:
   :name: hello.c
```

效果:

列表 4: ../../base_code/hello.c

```c
/* $begin hello */
#include <stdio.h>

int main()
{
        printf("hello, world! This is a C program.\n");
        for(int i=0;i<10;i++ ){
                printf("output i=%d\n",i);
        }

        return 0;
}
/* $end hello */

```

插入 shell 代码

语法:

```
.. literalinclude:: ../../base_code/hello_world.sh
   :caption: ../../base_code/hello_world.sh
   :language: sh
   :linenos:
```

效果:

列表 5: ../../base_code/hello_world.sh

```sh
echo "helloworld,this is a script test!"
```

插入 Makefile 代码

语法:

```
.. literalinclude:: ../../base_code/Makefile
   :caption: ../../base_code/Makefile
   :language: makefile
   :linenos:
```

效果:

列表 6: ../../base_code/Makefile

```
1  # 生成的可执行文件名
2
3  hello:hello.o
4          gcc -o hello hello.o
5
6  # 生成规则
7  hello.o:hello.c
8          gcc -c hello.c -o hello.o
9
10 # 伪目标
11 .PHONY:clean
12 clean:
13         rm -f *.o hello
```

10.3.2 嵌入文件的某部分

通过 lines 指定嵌入文件的某些行。

语法:

```
.. literalinclude:: ../../base_code/hello.c
   :caption: ../../base_code/hello.c
   :language: c
   :linenos:
   :lines: 1,3,5-8
```

效果:

列表 7: ../../base_code/hello.c

```
1  /* $begin hello */
2
3  {
```

(下页继续)

(续上页)

```
4         printf("hello, world! This is a C program.\n");
5         for(int i=0;i<10;i++ ){
6                 printf("output i=%d\n",i);
```

10.3.3 文件对比

语法:

```
.. literalinclude:: ../../base_code/hello.c
:diff: ../../base_code/hello_diff.c
```

效果:

```
--- /home/docs/checkouts/readthedocs.org/user_builds/ebf-contribute-guide/checkouts/
↪latest/base_code/hello_diff.c
+++ /home/docs/checkouts/readthedocs.org/user_builds/ebf-contribute-guide/checkouts/
↪latest/base_code/hello.c
@@ -5,7 +5,7 @@
 {
         printf("hello, world! This is a C program.\n");
         for(int i=0;i<10;i++ ){
-                printf("diff output i=%d\n",i);
+                printf("output i=%d\n",i);
         }

         return 0;
```

CHAPTER 11

图片与视频

添加图片的 sphinx 说明：http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html 添加图片的 rst 官方说明：

11.1 图片

图片原文件统一存储在引用文档所在的同级目录的 **media** 文件夹下。

显示图片直接使用 image 或 figure 指令，无特殊情况的话我们书籍图片要求使用居中方式显示，还需要添加“alt”选项指定图片的描述（类似 doc 中的题注），以便图片加载失败时显示文字

不要使用 bmp 图片，bmp 图片在生成 pdf 的时候会丢失，所以不要使用 bmp 格式的图片。

11.1.1 figure 命令

语法：

```
.. figure: ../media/rest-syntax/pic-video/logo.png
   :alt: 野火 logo
   :align: center
   :caption: 野火 logo
```

效果：

11.1.2 image 命令:

语法:

```
.. image:: ../media/rest-syntax/pic-video/logo.png
   :align: center
   :alt: 野火 logo
```

效果:

以下的图片显示方式是 word 自动转换的结果，使用这种方式无法以居中形式显示图片，所以我们要修改。

它的原理是使用“||”类似宏的方式替换指令，使 rst 源码看起来更简洁，但不能居中显示。

语法:

```
|logo|, 使用"| 宏名 |" 的形式替换文字。

.. |logo| image:: ../media/rest-syntax/pic-video/logo.png
```

效果:

，使用”|”宏名”|”的形式替换文字。

图片还可以使用 width、heigh、scale 等参数，但不推荐使用，设置过 width、height 参数的图片，在页面可以点击查看原图。

语法:

```
.. image:: ../media/rest-syntax/pic-video/logo.png
   :align: center
```

(下页继续)

(续上页)

```
   :width: 5.63529in
   :height: 0.97222in
```

效果：

11.2 视频

使用 raw 指令插入 html 代码。直接粘贴视频网站的通用 html 代码，不过貌似不能放大。

在 vscode 可能无法预览，直接在浏览器中看即可。

注解： TODO：目前还不知道如何放大，以及点击后到具体的视频网站播放。

语法：

```
.. raw:: html

<iframe src="//player.bilibili.com/player.html?aid=70961112&cid=122951107&page=1"␣
↪scrolling="no" border="0" frameborder="no" framespacing="0" allowfullscreen="true"> </
↪iframe>
```

效果：

语法：

```
.. raw:: html

<iframe height=498 width=510 src='http://player.youku.com/embed/XMzk2MzQxNTQ3Ng==='␣
↪frameborder=0 'allowfullscreen'></iframe>
```

CHAPTER 12

数学符号和公式

数学符号和公式

12.1 上标

使用 sub 表示平方等符号:

语法

```
I\ :sup:`2`\ C
```

效果:

I^2C

CHAPTER 13

特殊提示

特殊提示支持警告、重要、提示、注意等标签，适合做显眼的用途。

- attention
- caution
- danger
- error
- hint
- important
- note
- tip
- warning

语法：

```
.. note:: This is a note admonition.
 This is the second line of the first paragraph.

 - The note contains all indented body elements
   following.
 - It includes this bullet list.
```

(下页继续)

(续上页)

```
.. hint:: This is a hint admonition.

.. important:: This is a important admonition.

.. tip:: This is a tip admonition.

.. warning:: This is a warning admonition.

.. caution:: This is a caution admonition.

.. attention:: This is a attention admonition.

.. error:: This is a error admonition.

.. danger:: This is a danger admonition.
```

效果:

注解: This is a note admonition. This is the second line of the first paragraph.

- The note contains all indented body elements following.
- It includes this bullet list.

提示: This is a hint admonition.

重要: This is a important admonition.

小技巧: This is a tip admonition.

警告: This is a warning admonition.

警告: This is a caution admonition.

注意： This is a attention admonition.

错误： This is a error admonition.

危险： This is a danger admonition.

CHAPTER 14

rst 的指令和角色（directive and role）

- directive 说明：http://docutils.sourceforge.net/docs/ref/rst/directives.html
- role 说明：http://docutils.sourceforge.net/docs/ref/rst/roles.html

rst 使用 directive 和 role 进行特殊操作。

14.1 directive

directive 的格式为：

```
.. directive 名:: 参数
   : 参数:
   : 参数:
```

14.2 role

role 的格式为：

```
:role 名: 内容
```

CHAPTER 15

野火 sphinx 文档规范

开源项目的整体目录：

```
README                  说明文档或软链接至 documentation 中的说明，方便 github 阅读
base_code               配套代码
documentation           配套文档
    faq                 存储常见问题的文档
    about_us.rst        关于我们
    _build              文档编译输出目录
    conf.py             sphinx 配置文件
    contribute          如何参与项目，贡献与投稿的说明
    foreword.rst        前言
    index.rst           目录
    make.bat
    Makefile
    media               文档中使用的图片文件
    requirements.txt    文档的 python 依赖
    _static
    _templates
    TODO.rst            待完成的内容，发布的任务列表
```

注意：每次新添加了 rst 文档，必须要修改 documentation 文件夹下的 index.rst 文件，将新增的 rst 文件，添加到对应的地方。

15.1 图片

文档引用的图片存储在文档源码目录下的 media 文件夹中，按部分、章节及文档分目录存储。

图片要直接使用如下形式, 方便居中：

```
.. image:: media/logo.png
   :align: center
```

不要使用如下形式，如下形式是 docx 转换后的格式，它不支持居中，见到要把它改好。

```
|logo|

.. |logo| image:: media/logo.png
```

15.2 rst 格式检查

使用 vscode 的 rst 插件在编写时就会有格式检查。编写文档时应尽量满足该格式检查的规范。对于 docx 转 rst 后的不规范文档，做到见一个改一个。

make html 时，编译会有提示输出，尽量让它不输出的 warning。

15.3 文档编码和回车

文档编码统一用“utf-8”，回车使用“LF”，不要使用“CRLF”，文档编码和回车可在 VS-Code 的右下角设置。

15.4 代码引用

超过 3 行的代码要加上行号、并用 caption 名指明代码片段的名，对于引用的代码文件，使用 caption 指明引用的路径名。

如以下语法：

```
.. literalinclude:: ../../base_code/hello.c
   :caption: ../../base_code/hello.c
   :language: c
   :linenos:
```

15.5 类似 docx 的题注引用

rst 可通过链接实现类似 doc 的题注引用，它通过给内容添加 :name: 属性来实现，代码、图片、表格等都可以使用这种方式，使用 :name: 属性来定义引用名，通过引用名 + 下划线来实现引用：

语法：

```
.. literalinclude:: ../../base_code/hello.c
   :caption: ../../base_code/hello.c
   :language: c
   :name: 代码清单或自己起的引用名字
   :linenos:

引用的方式是使用 ":name:" 定义的名字加下划线 "_", 如：

代码清单或自己起的引用名字_
```

效果：

列表 1: ../../base_code/hello.c

```c
/* $begin hello */
#include <stdio.h>

int main()
{
        printf("hello, world! This is a C program.\n");
        for(int i=0;i<10;i++ ){
                printf("output i=%d\n",i);
        }

        return 0;
}
/* $end hello */

```

引用的方式是使用“:name:”定义的名字加下划线“_”，如：

代码清单或自己起的引用名字

语法：

```
.. image:: ../media/rest-syntax/pic-video/logo.png
   :align: center
```

(下页继续)

(续上页)

```
   :name: 野火 logo

引用的方式是使用 ":name:" 定义的名字加下划线 "_", 如:

野火 logo_
```

效果:

引用的方式是使用“:name:”定义的名字加下划线“_”, 如:

野火 *logo*

语法:

```
.. list-table:: Frozen Delights!
    :widths: 15 10 30
    :header-rows: 1
    :name: 测试表格

    * - Treat
      - Quantity
      - Description
    * - Albatross
      - 2.99
      - On a stick!
    * - Crunchy Frog
      - 1.49
      - If we took the bones out, it wouldn't be
        crunchy, now would it?
    * - Gannet Ripple
      - 1.99
      - On a stick!

引用的方式是使用 ":name:" 定义的名字加下划线 "_", 如:

测试表格_
```

效果:

表 1: Frozen Delights!

Treat	Quantity	Description
Albatross	2.99	On a stick!
Crunchy Frog	1.49	If we took the bones out, it wouldn’ t be crunchy, now would it?
Gannet Ripple	1.99	On a stick!

引用的方式是使用“:name:”定义的名字加下划线“_”, 如:

测试表格 __

CHAPTER 16

使 sphinx 支持 markdown

sphinx 默认是不支持 markdown 语法的，但可以通过插件来支持。

16.1 markdown 基础支持

recommonmark 的 PyPi 说明：https://pypi.org/project/sphinx-markdown-tables/

recommonmark 的 sphinx 说明：http://www.sphinx-doc.org/en/master/usage/markdown.html

recommonmark 的使用说明：https://recommonmark.readthedocs.io/en/latest/index.html

```
pip install recommonmark
```

在 conf.py 添加扩支持：

```
extensions = ['recommonmark']
```

配置后就可以在 sphinx 中使用 markdown 文件了

16.2 markdown 表格支持

要支持 markdown 的表格语法，还需要安装 sphinx-markdown-tables

markdown 表格支持：https://pypi.org/project/sphinx-markdown-tables/

安装：

```
pip install sphinx-markdown-tables
```

使用:

在 conf.py 添加扩展支持:

```
extensions = [
   'sphinx_markdown_tables',
]
```

CHAPTER 17

Markdown 示例

本文件的源码是一个 markdown 文件, 也就是说在本工程中直接添加 markdown 即可嵌入到 sphinx 文档中。

关于使 sphinx 支持 markdown 的详细配置说明，请参考文档markdown-sphinx。

markdown 的公式语法在 sphinx 可能不支持。

17.1 以下是 markdown 的语法使用示例

文档来源：https://www.mdeditor.com/

17.2 欢迎使用 Markdown 在线编辑器 MdEditor

Markdown 是一种轻量级的「标记语言」

markdown

Markdown 是一种可以使用普通文本编辑器编写的标记语言，通过简单的标记语法，它可以使普通文本内容具有一定的格式。它允许人们使用易读易写的纯文本格式编写文档，然后转换成格式丰富的 HTML 页面，Markdown 文件的后缀名便是 “.md”

17.3 MdEditor 是一个在线编辑 Markdown 文档的编辑器

MdEditor 扩展了 *Markdown* 的功能（如表格、脚注、内嵌 *HTML* 等等），以使让 *Markdown* 转换成更多的格式，和更丰富的展示效果，这些功能原初的 *Markdown* 尚不具备。

> Markdown 增强版中比较有名的有 Markdown Extra、MultiMarkdown、Maruku 等。这些衍生版本要么基于工具，如 ~~Pandoc~~，Pandao；要么基于网站，如 GitHub 和 Wikipedia，在语法上基本兼容，但在一些语法和渲染效果上有改动。

MdEditor 源于 Pandao 的 JavaScript 开源项目，开源地址Editor.md，并在 MIT 开源协议的许可范围内进行了优化，以适应广大用户群体的需求。向优秀的 markdown 开源编辑器原作者 Pandao 致敬。

Pandao editor.md

17.4 MdEditor 的功能列表演示

17.5 标题 H2

17.5.1 标题 H3

标题 H4

标题 H5

标题 H5

17.5.2 字符效果和横线等

~~ 删除线 ~~ 删除线（开启识别 HTML 标签时）

斜体字 *斜体字*

粗体 **粗体**

粗斜体 ***粗斜体***

上标：X2，下标：O2

缩写 (同 HTML 的 abbr 标签)

即更长的单词或短语的缩写形式，前提是开启识别 HTML 标签时，已默认开启

The HTML specification is maintained by the W3C.

17.5.3 引用 Blockquotes

> 引用文本 Blockquotes

引用的行内混合 Blockquotes

> 引用：如果想要插入空白换行即 `<br />` 标签，在插入处先键入两个以上的空格然后回车即可，
> 普通链接。

17.5.4 锚点与链接 Links

普通链接 普通链接带标题 直接链接：https://www.mdeditor.com [锚点链接][anchor-id] [anchor-id]: https://www.mdeditor.com/ mailto:test.test@gmail.com GFM a-tail link @pandao 邮箱地址自动链接 test.test@gmail.com www@vip.qq.com

> @pandao

17.5.5 多语言代码高亮 Codes

行内代码 Inline code

执行命令：`npm install marked`

缩进风格

即缩进四个空格，也做为实现类似 `<pre>` 预格式化文本（Preformatted Text）的功能。

```
<?php
    echo "Hello world!";
?>
```

预格式化文本：

JS 代码

```
function test() {
        console.log("Hello world!");
}
```

HTML 代码 HTML codes

```
<!DOCTYPE html>
<html>
    <head>
        <mate charest="utf-8" />
        <meta name="keywords" content="Editor.md, Markdown, Editor" />
        <title>Hello world!</title>
        <style type="text/css">
            body{font-size:14px;color:#444;font-family: "Microsoft Yahei", Tahoma,
↪"Hiragino Sans GB", Arial;background:#fff;}
            ul{list-style: none;}
            img{border:none;vertical-align: middle;}
        </style>
    </head>
    <body>
        <h1 class="text-xxl">Hello world!</h1>
        <p class="text-green">Plain text</p>
    </body>
</html>
```

17.5.6 图片 Images

图片加链接 (Image + Link):

Follow your heart.

17.5.7 列表 Lists

无序列表（减号）Unordered Lists (-)

- 列表一
- 列表二

- 列表三

无序列表（星号）Unordered Lists (*)

- 列表一
- 列表二
- 列表三

无序列表（加号和嵌套）Unordered Lists (+)

- 列表一
- 列表二
  - 列表二-1
  - 列表二-2
  - 列表二-3
- 列表三
  - 列表一
  - 列表二
  - 列表三

有序列表 Ordered Lists (-)

1. 第一行
2. 第二行
3. 第三行

GFM task list

- [x] GFM task list 1
- [x] GFM task list 2
- [ ] GFM task list 3
  - [ ] GFM task list 3-1
  - [ ] GFM task list 3-2
  - [ ] GFM task list 3-3

- [ ] GFM task list 4
  - [ ] GFM task list 4-1
  - [ ] GFM task list 4-2

17.5.8 绘制表格 Tables

特殊符号 HTML Entities Codes

© & ¨ ™ ¡ £ & < > ¥ € ® ± ¶ § ¦ ¯ « ·

X^2 Y^3 ¾ ¼ × ÷ »

18ºC “ ‘

[========]

17.5.9 Emoji 表情:smiley:

> Blockquotes :star:

GFM task lists & Emoji & fontAwesome icon emoji & editormd logo emoji :editormd-logo-5x:

- [x] :smiley: @mentions, :smiley: #refs, links, **formatting**, and tags supported :editormd-logo:;
- [x] list syntax required (any unordered or ordered list supported) :editormd-logo-3x:;
- [x] [ ] :smiley: this is a complete item :smiley:;
- [ ] []this is an incomplete item *test link* :fa-star: @pandao;
- [ ] [ ]this is an incomplete item :fa-star: :fa-gear:;
  - [ ] :smiley: this is an incomplete item *test link* :fa-star: :fa-gear:;
  - [ ] :smiley: this is :fa-star: :fa-gear: an incomplete item *test link*;

反斜杠 Escape

literal asterisks

[========]

17.5.10 科学公式 TeX(KaTeX)

$$E=mc^2$$

行内的公式 $$E=mc^2$$ 行内的公式，行内的 $$E=mc^2$$ 公式。

$$x > y$$

$$(\sqrt{3x-1}+(1+x)^2)$$

$$\sin(\alpha)^{\theta}=\sum_{i=0}^{n}(x^i + \cos(f))$$

多行公式:

```
\displaystyle
\left( \sum\_{k=1}^n a\_k b\_k \right)^2
\leq
\left( \sum\_{k=1}^n a\_k^2 \right)
\left( \sum\_{k=1}^n b\_k^2 \right)
```

```
\displaystyle
    \frac{1}{
        \Bigl(\sqrt{\phi \sqrt{5}}-\phi\Bigr) e^{
        \frac25 \pi}} = 1+\frac{e^{-2\pi}} {1+\frac{e^{-4\pi}} {
        1+\frac{e^{-6\pi}}
        {1+\frac{e^{-8\pi}}
         {1+\cdots} }
        }
    }
```

```
f(x) = \int_{-\infty}^\infty
    \hat f(\xi)\,e^{2 \pi i \xi x}
    \,d\xi
```

17.5.11 分页符 Page break

Print Test: Ctrl + P

[========]

17.5.12 绘制流程图 Flowchart

```
st=>start: 用户登陆
op=>operation: 登陆操作
cond=>condition: 登陆成功 Yes or No?
e=>end: 进入后台

st->op->cond
cond(yes)->e
cond(no)->op
```

[========]

17.5.13 绘制序列图 Sequence Diagram

```
Andrew->China: Says Hello
Note right of China: China thinks\nabout it
China-->Andrew: How are you?
Andrew->>China: I am good thanks!
```

17.5.14 End

CHAPTER 18

使用 pandoc 进行文档转换

使用 pandoc 可以方便地对文档的格式进行相互转换。

18.1 安装

pandoc 安装说明：https://github.com/jgm/pandoc/blob/master/INSTALL.md

pandoc 用户手册：https://pandoc.org/MANUAL.html

根据系统按它的说明下载对应的安装包即可，ubuntu 下不要直接用 apt 安装，apt 安装的版本太旧。

在 ubuntu 下可以使用如下命令查看 pandoc 的说明：

```
man pandoc
```

18.2 docx 转 rst

使用 python 安装 rstdoc 扩展包，方便转换：https://pypi.org/project/rstdoc/

docx 转 rst 操作步骤：

- 先把 docx 文档按章节分开，一个章节一个 docx 文档，不然整个文档转换可能太大，导致错误，而且整个文档转换图片或内容不方便处理。
- 使用 rstfromdocx 命令一个个文档转换成 rst

使用如下指令转换

```
#doc1 为要转换的文档
rstfromdocx -lurg doc1.docx
```

命令执行后会生成与文档名相同的目录，目录下是 sphinx 形式的 rest 文档，把 rest 文档复制至新的书籍目录并把后缀改为 rst，media 下的图片也放到目标书籍的同级目录即可。

小技巧: 在 Windows 转换后，生成的文件目录引用使用的路径是 “”，而 sphinx 路径只支持 “/”，所以图片之类的引用路径要注意修改。

18.3 pandoc rst 转 docx

rstdocx 插件据说可以更完美地把 rst 转成 docx，但还不清楚 rstdocx 插件如何把 rst 转成 docx，但用 pandoc 可满足基本要求。

```
pandoc rstfile.rst  -f   rst  -t docx  -o targetdocfile.docx  --data-dir=sourcedir  --
↪reference-doc=module.docx
```

- 其中 rstfile.rst 是源文件，-f 指源格式，-f 指目标格式
- -o 指定的是输出文件
- –data-dir= 指定的是依赖目录
- –reference-doc= 是 docx 模版文件，使用模版文件转换可以把标题之类的格式搞得更规范

reference-doc 模版及说明：https://pandoc.org/MANUAL.html#option–reference-doc

18.4 pandoc docx 转 rst

与 rst 转 docx 类似，输出的图片目录可以使用这个参数设置：

–extract-media=DIR

CHAPTER 19

常见问题

CHAPTER 20

版权说明

野火电子保留本项目的所有版权。

公司组织有生存的压力，因为开源而导致生存不下去，是有 GEEK 精神的人不愿看到的事情。由于我们还不熟悉各种版权条例，所以关于版权的问题还在选择中。

目前我们保留本项目的所有版权，但我们秉承开源的心是不变的。如果你使用本项目不是用于商业目的，基本不需要考虑版权问题。

我们主要担忧的商业目的如下，列出来的意思是如果用于以下目的，我们极有可能会追究版权。

- 同业竞争，目前开发板是我们主要的盈利来源，禁止把本项目用于其它开发板项目。
- 文档出版，项目中包含的文档我们随时会提交至出版社出版，所以我们保留文档出版的权利。如果你只是在自己的文章中使用了本项目文档中的内容，需要注明来源。