

技术图书排版



杜金房 著

INVITE sip:you@xswitch.cn SIP/2.0

Via: SIP/2.0/TLS xswitch.cn:5060;branch=z9hG4bK74bf9

Max-Forwards: 70

From: "Seven Du" <sip:seven@xswitch.cn>;tag=9fxced76sl

To: You <sip:you@xswitch.cn>

Call-ID: 3848276298220188511@xswitch.cn

CSeq: 2 INVITE

小樱桃出品

Contact: <sip:seven@xswitch.cn;transport=tls>

技术图书排版

杜金房

图书不在版编目（NCIP）数据

技术图书排版/杜金房 著/2019.7
ISBN 7-DU-777777-7

本书分享写作FreeSWITCH相关图书过程中总结的一些排版经验，也使用了类似《FreeSWITCH文集》的封面。本书所有源代码都在Github上，书中有相关链接。

技术图书排版

作 者	杜金房
封面设计	杜金房
校 对	杜金房
排 版	杜金房
责任编辑	杜金房
开 本	216 mm × 279 mm
印 张	7.5
印 数	7
版 数	2019年7月第1版 2019年7月第1次发布
电子邮箱	freeswitch@dujinfang.com

前言

我出过两本书——《FreeSWITCH权威指南》和《Kamailio实战》，也写过很多FreeSWITCH相关的电子书。有读者问我是怎么排版的，不揣鄙陋，愿与大家分享。

我最初写博客都是使用的是Markdown¹，后来写书也使用Markdown做简单排版。但一到出版社编辑那儿，就必须用Office了。也跟出版社聊过，是否可以用Latex排版，但出版社的答复是养一个Latex排版师太贵了，因此，大多数情况下，作者们还是需要使用Word/WPS排版。

对于那些写小说或故事的作者，或许用Word排版还是不错的，但是，对于像我这样的技术人员，由于书中有很多代码和图表，用Word排版就比较累，尤其是技术人员好多都在用Mac，与Windows版的Word兼容性还比较差，写起来就更痛苦了。

使用Markdown格式写作，比Word要轻松多了，同时，使用一些辅助工具也能做到比较好的排版。如果万一有一天写出来的书能够正式出版，也希望Word是最后一步，把痛苦留到最后。

Word有一个功能确实不大好替代，那就是『修订』功能，使用它可以让作者和编辑很方便地交互修改文件。但文件传来传去也很烦。不知道一些在线的协作工具如石墨文档等，是否适合这种协作。（2022更新：最近发现有的出版社已经在用WPS在线版了）

当然，其实作者不应该关注排版，而是在保证内容正确的前提下，把章节、强调、引用、代码之类的都标注出来即可，出版社是有专人进行排版的。而做这些标注，Markdown就够了。而且Markdown文件可以很方便地放到Git仓库中。我们不期望出版社所有的编辑也能熟悉Git，但如果真有哪天编辑们也喜欢Git了，那社会就真的进步了。

不过，话又说回来，Git主要是做代码管理的，Git的管理粒度也是基于行的。但在写作时，通常有大段的文字不换行。这样，即使一段（一行）中只修改一个字，或一个标点符号，就会出现一个大的diff。从这一点来说，不如Office的修订功能好用。

关于排版，本书主要讲一下相关的模板，以及一些排版原则：

- 支持标准的Markdown，暂不支持各种扩展；
- 使用开源字体；
- 代码应该有单独的格式，用等宽字体，最好支持语法高亮；
- 应该有移动版，目前大多数A4、16K、32K幅面的PDF并不适合在手机上阅读；

¹<https://en.wikipedia.org/wiki/Markdown>

- 移动版不应该首行缩进，因为页面太窄，缩进反而影响阅读体验；
- 标准版应该首行缩进；
- 印刷版应该奇偶页不同；
- 提供Word版方便与其它人交流；
- 还可以生成EPUB电子书等其它格式，欢迎到Github上给我提pr。
- 如果你发现错别字或别的错误，也欢迎提pr。

本书是可以『自举』的，也就是说你可以通过本书的源代码生成本书的PDF，参见：<https://github.com/seven1240/latex>（**注意**：PDF版不会实时更新，可能会比较旧）。你也可以访问本书生成的HTML：<http://www.freeswitch.org.cn/books/typesetting/>。

最后，本书写作的目的并不是教你成为一个排版专家，相反，希望通过分享我的排版模板，让广大技术人员专注于用自己喜欢的工具写好自己的作品，忘记排版;)。

杜金房/2019/烟台 更新于2023年2月1日

目 录

前 言	III
1 技术图书排版	1
1.1 Makefile	1
1.2 meta.md	4
1.3 diagram-generator.lua	5
1.4 webp.lua	5
1.5 docx-figure-number.lua	5
1.6 cover.tex	5
1.7 cover-std.tex	7
1.8 header.tex	8
2 主模板	10
3 我在Mac上的写作工具链	22
4 插图和公式	29
4.1 插图	29
4.2 公式	33
5 写作点滴	38
6 Pandoc安装与使用	44
6.1 安装Pandoc	44
6.2 安装Latex	44

技术图书排版	目 录
6.2.1 在macOS上安装Pandoc和Latex	44
6.2.2 在Linux上安装Pandoc和Latex	47
6.2.3 在Windowsh安装Pandoc和Latex	47
6.3 使用Docker	47
写在最后	49
作者简介	50
版权声明	51
广告	52
关于广告的广告	52
XSwitch	52
技术支持	52
FreeSWITCH相关图书	53
知识星球	54
	56

第一章 技术图书排版

欢迎来到排版世界。我们用Markdown¹格式写作，用LaTeX模板排版，用Pandoc做格式转换。

Markdown是一个文档格式，它是基于纯文本的，通过简单的格式约定，既做到源文件易读，又做到可以支持一些基本的格式。本文就是用Markdown写成的。

Pandoc²是一个文档转换工具，是一个瑞士军刀。它可以在各种文档格式间转换，在此我们会将我们的Markdown文件转换成PDF。

LaTeX³是世界上最先进的排版系统，在文档转换过程中，我们会用到LaTeX。推荐安装TexLive⁴，但完整版安装包有4个多G，太大了，所以一般安装BasicTex就够了，这个版本比较小，但宏包不全，遇到有缺少的宏包可以后续用 `tlmgr install 包名` 命令安装。

本书是在Mac上编译的，笔者也制作了个Docker镜像方便大家使用，参见下一节。

这是本书的第一章。到此，章节、段落、脚注、链接等写法你都已经看到了（当然，如果你阅读的是PDF的话，你需要看一下源文件）。下面，我们看下本书中用到的一些文件。

1.1 Makefile

关于Pandoc的用法我们不会详细解释，感兴趣的应该去看官方网站上的文档，我们只是解释一下我用到的一些命令。

先看Makefile。好吧，我们需要先学习一下Makefile。

Makefile不是必需的，但作为一名程序员，它是一个很方便使用的工具。

首先定义了一个变量，`PANDOC` 就指向 `pandoc` 可执行文件，如果在不同的系统上使用，可以更新路径。

`all` 定义了一个目标（Target），如果在命令行上执行 `make` 命令，就默认使用这个目标。可以看到，它其实依赖于另外几个目标 `mobile`、`book` 和 `print`，后面我们会讲到这些目标，在命令行上也可以单独 `make` 一个目标，如 `make mobile`。

¹<https://docs.xswitch.cn/xpedia/markdown/>。

²<https://pandoc.org/>

³<https://www.latex-project.org/>

⁴<http://www.tug.org/mactex/morepackages.html>

VER 只是一个版本号。**SRC** 为本书全部的源文件。

```
PANDOC := pandoc
```

```
all: mobile book print docx
```

```
VER=7
```

```
SRC = meta.md \  
      chapter-1.md \  
      chapter-2.md \  
      chapter-3.md \  
      chapter-4.md \  
      postface.md
```

先来个小目标;)，**preface.tex** 是一个小目标，是『前言』部分。因为我们希望前言能在目录的前面，所以我们需要一个 **tex** 文件，但我们还是想用Markdown格式写，所以，我们会把 **README.md** 转换成 **tex** 并插入到文档相应的位置。

其中 **-s** 为Smart的意思（嗯，欲知详情看官方文档），**--variable** 为变量，设置LaTeX的文档格式，后面我们会有模板文件中看到。**--template** 选择一个模板，在此，我们自创了一个空模板，它会生成不带模板的LaTeX文件。好吧，如果听不明白也没关系，你可以看一下生成的这个文件的内容，初步了解一下LaTeX的格式。

```
preface.tex: README.md  
    $(PANDOC) -s --variable documentclass=report \  
    --template template-dummy.tex \  
    -o preface.tex README.md
```

out 是一个目标，这就是Makefile的魔术，如果没有这个目标文件夹，就执行下面的**mkdir** 命令创建它。再次说明，Pandoc并不依赖于Makefile，你可以手工执行命令创建这个文件夹，但是我们使用Makefile只是为了方便。

```
out:  
    mkdir out
```

book 是我们标准的目标，它会生成一个PDF文件（通过 **-o** 指定）。**--toc** 是 Table of Content，即自动生成图书目录。这里的 **template.tex** 是我们的模板文件，后面我们还会详细讲。**--number-sections** 自动生成章节号。**pdf-engine** 我们选 **xelatex**，对中文比较友好，另一个选项是 **pdflatex**，但对中文支持稍差点。**--include-xxx** 表示将相关文件插入到模板文件相应的位置。

```
book: out preface.tex $(SRC)
      $(PANDOC) -s --toc \
      --template template.tex \
      --number-sections \
      --pdf-engine=xelatex \
      --include-in-header=cover-std.tex \
      --include-before=header.tex \
      --include-before-body=preface.tex \
      --lua-filter=diagram-generator.lua \
      -o out/book-标准版-$(VER).pdf \
      $(SRC)
```

标准版是A4的纸张，在手机上显示不适合阅读，我们生成个移动版的，设置纸张大小为 9 x 16cm，即对于 16:9 的手机屏幕，刚好显示一页。在此，我们传入了一个 `mobile=true` 参数。

```
mobile: out preface.tex $(SRC)
        $(PANDOC) -s --toc \
        --template template.tex \
        --number-sections \
        --pdf-engine=xelatex \
        --variable mobile=true \
        --include-in-header=cover.tex \
        --include-before=header.tex \
        --include-before-body=preface.tex \
        --lua-filter=diagram-generator.lua \
        -o out/book-移动版-$(VER).pdf \
        $(SRC)
```

图书是要出版的，一般出版的尺寸不会是A4的，所以要设置不同的尺寸，另外，出版图书跟电子阅读的版本还有一个重要的区别是奇偶页不同（页边距和页码位置等），我们加了个 `print` 变量控制打印的格式。看了这么多年书，你有没有注意到这个问题呢？

```
print: out preface.tex $(SRC)
       $(PANDOC) -s --toc \
       --variable print=true \
       --variable fontsize=11pt \
       --template template.tex \
       --number-sections \
       --pdf-engine=xelatex \
       --include-in-header=cover-dummy.tex \
       --include-before=header.tex \
       --include-before-body=preface.tex \
```

```
--lua-filter=diagram-generator.lua \  
-o out/book-印刷版-$(VER).pdf \  
$(SRC)
```

为了照顾那些顽固地想看Word版的人，我们增加了一个 `make docx`，就可以直接生成Word版了。

```
docx: out/preface.tex $(SRC)  
$(PANDOC) -s --toc \  
--number-sections \  
-o out/技术图书排版-$(VER).docx \  
README.md $(SRC)
```

如果你本地没有安装Pandoc以及LaTeX环境，可以使用笔者制作的Docker镜像。在命令行上执行 `make docker` 会进入一个Docker容器中，并把当前目录映射到 `/team` 目录中，然后就可以继续 `make` 生成PDF了。

```
docker:  
docker run --rm -it -v `PWD`:/team ccr.ccs.tencentyun.com/free/pandoc:multiarch bash
```

读到这里，如果你还是不理解Makefile，可以参考笔者的另一篇文章《[Makefile极速入门](#)》。

1.2 meta.md

`meta.md` 里面定义了一些变量，YAML格式。这些变量在主模板文件中会用到。

```
---  
documentclass: report  
title: 技术图书排版  
author: 杜金房  
title-meta: 技术图书排版  
author-meta: 杜金房  
publisher: 版权所有\qqquad 侵权必究  
verbatim-in-note: true  
---
```

1.3 diagram-generator.lua

这是一个Lua脚本，用于将以 `graphviz` 和 `mscgen` 标记的代码块转换成图片。

1.4 webp.lua

这也是一个Lua脚本，由于Latex不支持 `webp` 格式的图片，替换成一段说明。这主要是为了使用同一个Markdown源文件适配PDF和HTML的情况。

```
-- 实现一个函数判断字符串是否以某个字符串结尾
function string:endswith(ending)
    return ending == "" or self:sub(-#ending) == ending
end

-- 对每一个图片，将自动调用如下函数，该函数仅对latex格式生效
function Image(img)
    if FORMAT ~= "latex" then return end

    if img.src:endswith(".webp") then
        return pandoc.Str("!PDF版不支持该类型的图片[webp image]!")
    end
end
```

1.5 docx-figure-number.lua

Lua脚本，仅用于 `docx` 格式的文件，为图片添加编号。

Pandoc虽然有 `native-numbering` 选项，但是对于图片编号不能区分章节。

1.6 cover.tex

我们先从这个文件熟悉一下LaTeX的语法。你不需要精通LaTeX，但学一点总是有好处。

这是本书的封面，如果读到这里，你应该已经看到这个封面了。简单起见，我直接用了《FreeSWITCH文集》的封面。虽然我们可以直接用个PNG或JPEG图片做封面，但是作为一名程序员，我还是喜欢用代码生成封面，尽量少地依赖PhotoShop之类的软件。

我们使用 `tikz` 宏包，嗯，它是在LaTeX里画图用的。类似于程序语言中的模块，LaTeX使用宏包扩展本身的功能。其中 `%` 是注释。如果把 `texcoord` 那行注释去掉，可以看到一些参考线。

```

\usepackage{tikz}
\usepackage[absolute,overlay]{textpos}
% uncomment to see grid system
% \usepackage[teXcoord,grid,gridcolor=red!10,subgridcolor=green!10,gridunit=cm]{eso-pic}

```

`shadowtext` 给『技术图书排版』画上阴影。

```

\usepackage{shadowtext}
\shadowcolor{black}
\shadowoffset{1pt}

```

下面定义了一个新命令 `\cover`，用于在正文中『画』封面。从 `yellow` 到 `Orange` 做个渐变的颜色背景。然后把颜色再切换成白色。

```

\newcommand{\cover}{
\begin{tikzpicture}[remember picture,overlay]
\path [top color = yellow, bottom color = Orange] (current page.south west) rectangle (current page.north
↪ east);
\end{tikzpicture}
\color{white}
}

```

设置这是一个『空』（`empty`）页面（告诉LaTeX不需要自动生成页码之类的），居中，写上『FreeSWITCH』，加上 `wenji.png`，这是一个图片，然后写上作者。

```

\thispagestyle{empty}
\pagenumbering{gobble}
% \chapter*{}
\bigskip
\begin{center}
\textbf{\fontsize{36}{48}\selectfont\shadowtext{技术图书排版}}
\\[2em]
\oldincludegraphics[width=0.5\paperwidth]{wenji.png}
\\[2em]
\textbf{\large \theauthors \quad 著}\\[2em]
\end{center}

```

找一个位置，放上『小樱桃出品』，使用绝对坐标。

```

\begin{textblock*}{4cm}[0.5,0.5](0.5\paperwidth, 13.8cm)
\small
\color{white}
\center
\textbf{小樱桃出品}
\color{black}
\end{textblock*}

```

`bigskip` 会自动填充中间的空间，然后在下面画一个文本框放上一些装饰文本（笔者是做SIP通信的，因此放了一段SIP消息）。

```

\bigskip
\vfill
\begin{adjustwidth}{-2.3mm}{ }
\mbox{\vbox{
\color{yellow}
\small
INVITE sip:you@xswitch.cn SIP/2.0\newline
Via: SIP/2.0/TLS xswitch.cn:5060;branch=z9hG4bK74bf9\newline
Max-Forwards: 70\newline
From: {\color{yellow!60!white}\Oldtexttt{}}Seven Du\Oldtexttt{}}
↔ <sip:seven@xswitch.cn>;tag=9fxced76sl\newline
To: You <sip:you@xswitch.cn>\newline
Call-ID: 3848276298220188511@xswitch.cn\newline
CSeq: 2 INVITE\newline
Contact: <sip:seven@xswitch.cn;transport=tls>
}}
\end{adjustwidth}

\color{black}
}

```

LaTeX的语法比较奇怪，还是那句话，懂不懂没关系（因为模板我已经写好了啊，除非你要做自己的封面;))。

不过瘾？下面再来一个。

1.7 cover-std.tex

这个文件其实跟 `cover.tex` 差不多，只是调整了一些尺寸，看看 `diff` 吧：

```

$ diff cover.tex cover-std.tex
8c8
< \shadowoffset{1pt}
---
> \shadowoffset{2pt}
21c21
< \textbf{\fontsize{36}{48}\selectfont\shadowtext{技术图书排版}}
---
> \textbf{\fontsize{72}{96}\selectfont\shadowtext{技术图书排版}}
23c23
< \ldincludegraphics[width=0.5\paperwidth]{img/wenji.png}
---
> \ldincludegraphics[width=0.4\paperwidth]{img/wenji.png}
25c25
< \textbf{\large \theauthors \quad 著\[\[2em]]}
---
> \textbf{\huge \theauthors \quad 著\[\[2em]]}
28,29c28,29
< \begin{textblock*}{4cm}[0.5,0.5](0.5\paperwidth, 13.8cm)
< \small
---
> \begin{textblock*}{4cm}[0.5,0.5](0.5\paperwidth, 24cm)
> \Large
38d37
< \begin{adjustwidth}{-2.3mm}{}}
41c40
< \small
---
> \Large
51d49
< \end{adjustwidth}

```

1.8 header.tex

嗯，标准的书上都有个『图书在版编目（CIP）数据』，我们的书还没有出版，就加上个『不』吧。

其中 `tabular` 是表格，其它的不多解释了吧，对照效果图自己看，哈哈。

```

\newpage
\pagecolor{white}
\thispagestyle{empty}
% \vspace*{0.5cm}

```

```
\noindent\textbf{图书不在版编目（NCIP）数据}
```

```
\vspace{1em}
```

```
\noindent\thetitle/\theauthor\quad 著/2019.7
```

```
\\
```

```
\noindent ISBN 7-DU-777777-7
```

```
% \vspace*{2cm}
```

```
\noindent {本书分享写作FreeSWITCH相关图书过程中总结的一些排版经验，也使用了类似《FreeSWITCH文集》的封面。}
```

```
\bigskip
```

```
\vfill
```

```
\noindent {\bf \thetitle}
```

```
\begin{adjustwidth}{-2.3mm}{}
```

```
\begin{tabular}{cl}
```

```
\hline
```

```
{\bf 作\qquad 者} & \theauthor\\
```

```
{\bf 封面设计 } & \theauthor\\
```

```
{\bf 校\qquad 对} & \theauthor\\
```

```
{\bf 排\qquad 版} & \theauthor\\
```

```
{\bf 责任编辑} & \theauthor\\
```

```
{\bf 开\qquad 本} & \printlen[0][mm]{\paperwidth} \times \printlen[0][mm]{\paperheight}\\
```

```
{\bf 印\qquad 张} & 7.5\\
```

```
{\bf 印\qquad 数} & 7\\
```

```
{\bf 版\qquad 数} & 2019年7月第1版\quad 2019年7月第1次发布\\
```

```
{\bf 电子邮箱} & freeswitch@dujinfang.com \\
```

```
\hline
```

```
\end{tabular}
```

```
\end{adjustwidth}
```

```
\vfill
```

```
\begin{center}
```

```
{\bf \thepublisher}
```

```
\end{center}
```


第二章 主模板

`template.tex` 是我们的主模板。之所以我们将封面等模板放到其它的文件里，是因为这样会便于用一套模板做不同的书。

Latex文档的开头叫导言区（Preamble），定义了纸张及其它排版的参数。`if-else` 之类的条件判断是Pandoc加上去的，它会根据命令行上输入的参数不同选择不同的设置，比如一般的图书用 `report`，而一篇文章则用 `article`。你也可以看到，如果有 `print` 参数生成可以印刷图书的话，那就会加上 `twoside,openright` 以支持双面打印，以及奇偶而不同之类。

```
\documentclass[ $\if(fontsize)$$fontsize$, $\endif$$\if(lang)$$lang$, $\endif$CJKutf8 $\if(print)$ 
↪ $,twoside,openright $\endif$]{ $\documentclass$}$$$$$ 
```

中文支持设定。

```
\XeTeXlinebreaklocale "zh"
\XeTeXlinebreakskip = 0pt plus 1pt minus 0.1pt
\hyphenation{FreeSWITCH}
\usepackage{xprintlen} % print paper length in mm
```

根据变量设置纸张大小。

```
 $\if(mobile)$$ 
% \usepackage[showframe=true,papersize={9cm, 16cm}, text={8.4cm, 14cm}]{geometry}
\usepackage[papersize={9cm, 16cm}, text={8.4cm, 14.1cm}]{geometry}
 $\else$$ 
 $\if(print)$$ 
\usepackage[papersize={19cm, 23.6cm},text={15cm, 19.6cm}]{geometry}
 $\else$$ 
\usepackage[top=1in,bottom=1in,left=1.25in,right=1.25in]{geometry}
 $\endif$$ 
 $\endif$$ 
```

跟字体相关的设置。我们使用了谷歌的思源CJK（中日韩文）字体¹。嗯，因为该字体是开源的。如果你使用其它字体嵌入到PDF中，要注意字体的版权问题。

`mainfont` 是主字体，`monofont` 是等宽字体，`romanfont` 就是英文相关的代码相关的字体，我们实验性地使用了Adobe的Source Code Pro，因为它有斜体（Italic），注意中文是没有斜体的，我们用楷体或德意黑体²代替。所有用到的字体都已经打包到了Docker镜像里，如果你不使用Docker，你需要下载并安装这些字体。可以从它们的官方网站上找找，或者到<https://github.com/seven1240/font>上面找。

```
\usepackage{changepage}
\usepackage{float}
\usepackage{fontspec}
\newcommand\mainfont{Noto Sans CJK SC DemiLight}
\newcommand\boldfont{Noto Sans CJK SC Bold}
\newcommand\itfont{Smiley Sans}
\newcommand\kaifont{Adobe Kaiti Std}
\newcommand\fangsong{Adobe Kaiti Std}
\setmainfont[BoldFont=\boldfont,ItalicFont={\kaifont}]{\mainfont}
\newfontfamily\kai{\kaifont}
\newfontfamily\fs{\fangsong}
\newfontfamily\zhfont[BoldFont=\boldfont,ItalicFont={\kaifont}]{\mainfont}
\newfontfamily\zhpunctfont[BoldFont=\boldfont]{\mainfont}
\setromanfont[Mapping=tex-text,BoldFont=\boldfont,ItalicFont=\itfont]{\mainfont}
\setmonofont{Noto Sans Mono CJK SC}
```

设置中文间距等（最新版的可能不需要了）。

```
\usepackage{zhspacing}
\zhspacing
```

`longtable` 支持跨页的表格，普通表格不能跨页。

```
\usepackage{longtable}
```

首行缩进。如果是手机端，就不缩进，因为手机屏幕比较小，缩进会不好看。

¹<https://www.google.com/get/noto/>

²德意黑体是2022年发布的免费开源字体，本身就是按斜体设计的，参见 <https://github.com/atelier-anchor/smiley-sans>

。

```
\usepackage{indentfirst}
```

```
$if(mobile)$
\setlength{\parindent}{0em}
$else$
\setlength{\parindent}{2em}
$endif$
```

`fancyvrb` 是设置每页的页眉和页脚格式。注意 `\leftmark` 和 `rightmark` 是设成奇偶页不同的。

```
\usepackage{fancyvrb}
\DefineVerbatimEnvironment{verbatim}{Verbatim}
{frame=lines,
framerule=0.4pt,
baselinestretch=1,
fontfamily="Source Code Pro",
fontsize=\tiny,
xleftmargin=0pt,
xrightmargin=0pt,
rulecolor=\color{grey},
framesep=3mm,
numbers=left,
samepage=true
}
\fvset{frame=lines,framerule=0.4pt,rulecolor=\color{cyan},framesep=3mm}
```

```
\usepackage{fancyhdr}
\pagestyle{fancy}
\fancyhead{}
\fancyfoot{}
\fancyhead[L0,LE]{\$title$}
\fancyfoot[C]{\small ----- \thepublisher -----}
\fancyfoot[R0, LE]{\thepage}
$if(article)$
$else$
% \fancyhead[LE,R0]{\chaptermark}
$endif$
% \fancyhead[L0,RE]{\sectionmark}
$if(mobile)$
$else$
\fancyhead[R0]{\nouppercase{\leftmark}}
\fancyhead[RE]{\nouppercase{\rightmark}}
\headheight 25pt
\headsep 10pt
$endif$
```

默认章节号是英文的，翻译成中文。

```

\renewcommand{\contentsname}{目\quad 录}
\renewcommand\listfigurename{插图目录}
\renewcommand\listtablename{表格目录}
% \renewcommand\refname{参考文献}
\renewcommand\indexname{索引}
\renewcommand\figurename{图}
\renewcommand\tablename{表}
\renewcommand\abstractname{摘要}
\renewcommand\partname{第\,\the part\,部分}
\renewcommand\appendixname{附录}
\renewcommand\today{\number\year 年\number\month 月\number\day 日}
\providecommand{\CJKnumber}[1]
→ {\ifcase#1\or{一}\or{二}\or{三}\or{四}\or{五}\or{六}\or{七}\or{八}\or{九}\or{十}\or{十一}\or{十二}\or{十}
→ 三}\or{十四}\or{十五}\or{十六}\or{十七}\or{十八}\or{十九}\or{二十}\or{二十一}\or{二十二}\or{二十}
→ 三}\or{二十四}\or{二十五}\or{二十六}\or{二十七}\or{二十八}\or{二十九}\or{三十}\fi}

\usepackage{titlesec}
\titleformat{\chapter}{\centering\LARGE\bfseries}{\textbf{第\CJKnumber{\thechapter}章}}{0.5em}{}
$if(article)$
$else$
\renewcommand{\chaptername}{第\CJKnumber{\thechapter}章}
$endif$

\titleformat{\part}{\centering\Huge}{第\,\the part\,部分}{1em}{}
\renewcommand\partname{第\,\the part\,部分}

```

这一段设置不知道是否起作用，还是页眉页脚相关的。

```

\usepackage{fancyvrb}
\fvset{fontsize=\footnotesize}
% \fvset{xleftmargin=0.8cm}
\fvset{frame=lines,framerule=0.4pt,rulecolor=\color{cyan},framesep=3mm}
% \fvset{commandchars=\\\{\}}

```

Verbatim是代码段的环境。

```

\RecustomVerbatimEnvironment{verbatim}{Verbatim}{}

```

设置一些颜色。

```
\usepackage[usenames, dvipsnames]{xcolor}
\definecolor{mygray}{gray}{0.9}
\definecolor{darkblue}{rgb}{0.0, 0.0, 0.61}
\definecolor{indigo}{rgb}{0.29, 0.0, 0.51}
\definecolor{navyblue}{rgb}{0.0, 0.0, 0.5}
\definecolor{NAVYBLUE}{rgb}{0.0, 0.0, 0.5}
\definecolor{myyellow}{RGB}{255,255,0}
\definecolor{thegray}{RGB}{60,60,60}
\definecolor{darkblue}{RGB}{0,0,139}
\definecolor{lime}{RGB}{0,255,0}
\definecolor{wireshark}{RGB}{0,153,204}
\definecolor{wireshark1}{RGB}{102,204,255}
\definecolor{BLACK}{rgb}{0.0, 0.0, 0.0}
\definecolor{darkgreen}{RGB}{0,100,0}
```

设置引用相关的字体和颜色，比如本段就是一段引用，源文件中以 > 开头，注意到了吗？

```
\newfontfamily\quote font{STKaiti}
\let\quoteOLD\quote
\def\quote{\quoteOLD\color{thegray}\small\quote font\selectfont}
```

奇偶页不同的设置。

```
$if(print)$
\let\tmp\oddsidemargin
\let\oddsidemargin\evensidemargin
\let\evensidemargin\tmp
\reversemarginpar
$endif$
```

还是代码段环境的一些设置。

```
% for inline code
\let\Oldtexttt\texttt
\renewcommand{\texttt}[1]{\Oldtexttt{\,\color{navyblue}#1\color{black}\,}}
```

数学公式相关。

```
% \usepackage{lmodern}
\usepackage{amssymb,amsmath}
```

这是Pandoc提供的模板的一些默认设置，没改，也没有深入研究。

```
\usepackage{ifxetex,ifluatex}
\usepackage{fixltx2e} % provides \textsubscript
% use microtype if available
\IfFileExists{microtype.sty}{\usepackage{microtype}}{}
\ifnum 0\ifxetex 1\fi\ifluatex 1\fi=0 % if pdftex
  \usepackage[utf8]{inputenc}
$if(euro)$
  \usepackage{eurosym}
$endif$
\else % if luatex or xelatex
  \usepackage{fontspec}
  \ifxetex
    \usepackage{xltxtra,xunicode}
  \fi
  \defaultfontfeatures{Mapping=tex-text,Scale=MatchLowercase}
  \newcommand{\euro}{€}
$if(mainfont)$
  \setmainfont{$mainfont$}
$endif$
$if(sansfont)$
  \setsansfont{$sansfont$}
$endif$
$if(monofont)$
  \setmonofont{$monofont$}
$endif$
$if(mathfont)$
  \setmathfont{$mathfont$}
$endif$
\fi
$if(geometry)$
\usepackage[$for(geometry)$geometry$$sep$, $endfor$]{geometry}
$endif$
$if(natbib)$
\usepackage{natbib}
\bibliographystyle{plainnat}
$endif$
$if(biblatex)$
\usepackage{biblatex}
```

```

$if(biblio-files)$
\ bibliography{$biblio-files$}
$endif$
$endif$

```

`listings` 也是代码段环境，但我们好像没有用到（或者是以前用到会有什么问题），如果在命令行参数中使用 `--listings` 就可以启用它。

```

$if(listings)$
\usepackage{listings}
\lstset{ % General setup for the package
  language=perl,
  basicstyle=\small\sffamily,
  numbers=left,
  numberstyle=\tiny,
  frame=tb,
  tabsize=4,
  columns=fixed,
  showstringspaces=false,
  showtabs=false,
  keepspaces,
  commentstyle=\color{red},
  keywordstyle=\color{blue},
  backgroundcolor=\color{mygray},
  rulecolor=\color{cyan},
  % fancyvrb=true,
  breaklines=true
}
$endif$

$if(lhs)$
\lstnewenvironment{code}{\lstset{language=Haskell,basicstyle=\small\ttfamily}}{}
$endif$

$if(highlighting-macros)$
$highlighting-macros$
$endif$

$if(verbatim-in-note)$
\usepackage{fancyvrb}
$endif$

$if(fancy-enums)$
% Redefine labelwidth for lists; otherwise, the enumerate package will cause
% markers to extend beyond the left margin.
\makeatletter\AtBeginDocument{%
  \renewcommand{\@listi}{
    {\setlength{\labelwidth}{4em}}
  }
\makeatother

```

```
\usepackage{enumerate}
$endif$
```

设置图片格式默认为图片大小，如果超出页面则自动缩放。

```
$if(tables)$
\usepackage{ctable}
\usepackage{float} % provides the H option for float placement
$endif$
\let\Oldincludegraphics\includegraphics
$if(graphics)$
\usepackage{graphicx}
% We will generate all images so they have a width \maxwidth. This means
% that they will get their normal width if they fit onto the page, but
% are scaled down if they would overflow the margins.
\makeatletter
\def\maxwidth{\ifdim\Gin@nat@width>\linewidth\linewidth
\else\Gin@nat@width\fi}
\makeatother
\let\Oldincludegraphics\includegraphics
\renewcommand{\includegraphics}[1]{\Oldincludegraphics[width=\maxwidth]{#1}}
% hack figure to position figure at Here!
\makeatletter
\renewcommand\fps@figure{H}
\makeatletter
$endif$
```

URL相关的处理，以及其它。

```
\usepackage[hyphens]{url}
\ifxetex
  \usepackage[setpagesize=false, % page size defined by xetex
    unicode=false, % unicode breaks when used with xetex
    xetex]{hyperref}
\else
  \usepackage[unicode=true]{hyperref}
\fi
\hypersetup{breaklinks=true,
  bookmarks=true,
  pdfauthor={\$author-meta$},
  pdftitle={\$title-meta$},
  colorlinks=true,
  urlcolor=\if(urcolor)\$urlcolor\else$blue$endif$,
  linkcolor=\if(linkcolor)\$linkcolor\else$magenta$endif$,
```

```

pdfborder={0 0 0}}
$if(links-as-notes)$
% Make links footnotes instead of hotlinks:
\renewcommand{\href}[2]{#2\footnote{\url{#1}}}
$endif$
$if(strikeout)$
\usepackage[normalem]{ulem}
% avoid problems with \sout in headers with hyperref:
\pdfstringdefDisableCommands{\renewcommand{\sout}{} }
$endif$
% \setlength{\parindent}{2em}
\setlength{\parskip}{6pt plus 2pt minus 1pt}
\renewcommand{\baselinestretch}{1.4}
\setlength{\emergencystretch}{3em} % prevent overfull lines
$if(numbersections)$
$else$
\setcounter{secnumdepth}{0}
$endif$
$if(verbatim-in-note)$
\VerbatimFootnotes % allows verbatim text in footnotes
$endif$
$if(lang)$
\ifxetex
\usepackage{polyglossia}
\setmainlanguage{$mainlang$}
\else
\usepackage[$lang$]{babel}
\fi
$endif$

```

代码高亮。

```

\usepackage{fvextra}
\DefineVerbatimEnvironment{Highlighting}{Verbatim}{breaklines,commandchars=\\\{\}}

```

其它，将变量变成命令以便在正文中使用。

```

\usepackage{afterpage}

\newcommand{\thetitle}{$title$}
\newcommand{\theauthor}{$author$}
\newcommand{\theauthors}{$author$}
\newcommand{\thepublisher}{$publisher$}

```

图片和表格的编号默认显示为 `m.n`，改为 `m-n` 格式。

```

\renewcommand {\thetable} {\thechapter{}-\arabic{table}}
\renewcommand {\thefigure} {\thechapter{}-\arabic{figure}}
% \renewcommand {\thelisting} {\thechapter{}-\arabic{listing}}
\renewcommand {\theequation} {\thechapter{}-\arabic{equation}}

% fix --listing missing passthrough https://github.com/laboon/ebook/issues/139
% \newcommand{\passthrough}[1]{\lstset{mathescape=false}\color{red}#1\color{black}\lstset{mathescape=true}}

```

命令行中指定的文件可以插入到这些位置。

```

$for(header-includes)$
$header-includes$
$endfor$

$if(title)$
\title{$title$}
$endif$

\author{$for(author)$author$$sep$ \and $endfor$}
$if(date)$
\date{$date$}
$else$
\date{\today}
$endif$

```

文档开始。 `\cover` 画封面，还记得 `cover.tex` 吗？

```

\begin{document}
\cover

```

封面后的一页，书名和作者、出版社等。

```

\newpage
\thispagestyle{empty}
\pagenumbering{gobble}
\begin{center}
\vspace*{2cm}
$if(mobile)$
\textbf{\huge \thetitle\[\[1em]}

```

```
\textbf{\Large \theauthors}
$else$
\textbf{\Huge \thetitle\}[1em]}
\textbf{\LARGE \theauthors}
$endif$
\end{center}

\bigskip
\vfill
\begin{center}
{\color{blue}\textbf\thepublisher}
\end{center}
```

用罗马数字标记页号。

```
\pagenumbering{Roman}
```

这里插入前言。

```
$for(include-before)$
$include-before$
$endfor$
```

如果有 `--toc`，则生成目录。

```
$if(toc)$
{
\cleardoublepage
\hypersetup{linkcolor=blue}
\tableofcontents
}
$endif$
```

书的正文开始，将页号改为阿拉伯数字。

```
\cleardoublepage
\pagenumbering{arabic}
```

正文，Markdown的内容全部会转换成Latex格式放到这儿。

`$body$`

后面的参考资料之类的，我们这里没有用到。

```
$if(natbib)$
$if(biblio-files)$
$if(biblio-title)$
$if(book-class)$
\renewcommand\bibname{$biblio-title$}
$else$
\renewcommand\refname{$biblio-title$}
$endif$
$endif$
\ bibliography{$biblio-files$}
$endif$
$endif$
$if(biblatex)$
\printbibliography$if(biblio-title)$[title=$biblio-title$]$endif$
$endif$

$for(include-after)$
$include-after$
$endfor$

\end{document}
```

好玩不？

第三章 我在Mac上的写作工具链

杜金房/2015.08.13/来自FreeSWITCH-CN公众号

有读者问我在写书时是如何画插图又如何生成PDF的，今天，就跟大家分享一下。

先讲个故事。

大约09年底10年初的时候，我就想写点关于FreeSWITCH的东西，也希望最后能汇集成书。因此，我从最开始就研究怎么排版。

最开始，是从第二章写起的。当时直接使用Textmate写，使用Latex语法，用PdfLatex转成PDF。事实证明，写的很痛苦。因为，跟那些写小说的人不同，我的书中有好多代码。大段的代码还好说，直接嵌入段落中的代码就比较麻烦，必须用一堆`$$`括起来，而且，如果代码中有`$`就得用`\$`转义。不幸的是，我最开始写的那一章中有很多的`$`。

后来，发现写书是很遥远的事，就开始把内容放到我的博客上。博客是自己建的，使用简单的Markdown格式。Markdown写起来很方便，而且兼容HTML，一切都好。

但我一直忘不了寻求如何生成PDF的方案。也尝试过RestructuredText等方案，但都不完美。

再后来，一个自称是Tim Yang的网友鼓励我把书印出来。排版的事就又提上了日程。由于我时间不多，一个老朋友答应帮忙。后来，还真让她整理出了一个Word版。但是，问题是Word版里有很多地方需要改，而修改Word版，不仅是很累的活，而且，无法同步回Markdown，并且我也不怎么喜欢（极其讨厌）Word。所以，我只好把那么精心打造的Word版给废了。再次寻找从Markdown生成PDF的方案。很幸运，我找到了Pandoc。

Pandoc是用Haskell编写的软件，能从各种文件格式中互转。比较典型的就是从Markdown转成PDF、HTML和Word。

当然，它还是先将Markdown转成Latex，再从Latex转成PDF。通过精心调校以及一些Shell脚本的帮助（用于调整Latex），终于能生成比较美观的PDF了。书也印了出来，名为《FreeSWITCH：VoIP实战》。在第一届FreeSWITCH沙龙上卖出去几本，后来又间或有人来买，书也一直在更新。

但，PDF比较美观，跟完美还差一个数量级。所以，我也一直在改进。当然，在过去的几年里，Pandoc也一直在升级。

后来，我从PdfLatex转成了XeLatex，效果要好很多（具体差别细节不记得了）。我的书中有很多脚注，写起来非常方便。而且，在使用中，我也慢慢学会了如何通过Markdown中适当地嵌入一

些Latex标记来做交叉链接。现在，除了还有下列两个问题外，已经几近完美了。

1) 我还没有搞定字体（我希望某些引用使用楷体，不知道为什么使用楷体后有些副作用），好在我可以通过改变字号变相的区别。

2) 图文混排控制力较弱。如果不在Markdown中增加Latex标记（我不想增加），就只能实现简单的图文混排，即图片永远占一行，而无法使用文字环绕。这种情况的一个最大问题就是一个竖版的图片，往往会占一整页。如果不想让它占一整页，只好修改图片，用白色或无色背景加宽。好在我会点PS。

同样的Markdown也可以生成ePub格式，从ePub可以继续生成Kindle格式。这两种格式与PDF格式比起来，还不是很完美（因为PDF有Latex帮助，而ePub版不能利用Latex标签，如果需要更好的ePub，需要一些相关的插件才行），但是，一般来说也够用了，在移动设备上有比较好的阅读体验。

不过，即使你的Latex和PDF再精美，出版社也不会用的。后来，出书的时机成熟后我发现，我的出版社只接受Word格式的稿子。他们说中国的环境不足以养一个会Latex的编辑。另外，Word的修改和批注功能确实没有什么很好的替代品。我虽然跟他们说『我们可以使用Git啊』，但他们只是笑笑。其实我自己也知道，Git是基于行的，不适合diff大段的文本（更别提中文单词和标点之间没有空格了）。

所以，我只好将我的Markdown通过Pandoc转成Word格式。再跟编辑们一遍遍地校对和修改。当然，这些修改就只存在于Word版了。除非手工同步修改Markdown，不可能再从Word版转回原来的Markdown。好在，书出版以后，我不需要再继续打印了。因此，也没有必要再保持Markdown跟最终的版本同步。而且，出版社给出了一个Word模版，通过使用一些快捷键，可以很方便地格式化一级标题二级标题以及代码块等，手工按这些快捷键处理一章也基本上就是几分钟的事（Pandoc似乎也支持模板，但我没有试过，因为反正在编辑的时候还是要再次阅读和修改的）。

书于去年6月出版，出版时的名字叫《FreeSWITCH权威指南》。

后来，在读了《Producter》一书（电子书）后，我也深受启发，在SelfStore上发了几本我的电子书《FreeSWITCH互联互通》、《FreeSWITCH实例解析》等。

当然，如果我今天只讲故事，没有干货的话，你可能觉得我是在做广告了。所以，接下来干货。

在写书时，要有很多的插图。据说Mac上最好的画图工具是OmniGraffle，然而，我始终没有冲动试用它。而是尝试了很多不同的软件：Skitch、Dia、yEd、XMind、Keynote等。这些软件的问题是画得不好看，而且风格不统一。所以，最后，我又回到了程序员神器——Graphviz。

通过Graphviz，可以使用很简单的dot语言画出很复杂的图，比如今天的题图，对应的dot源文件如下（虽然长，但很直观）：

```
graph G {
    ranksep = 3;
    ratio = auto;

    A [Label = ""];
```

```
B [Label = ""];
C [Label = ""];
D [Label = ""];
E [Label = ""];
F [Label = ""];
G [Label = ""];
H [Label = ""];
I [Label = ""];
J [Label = "];
```

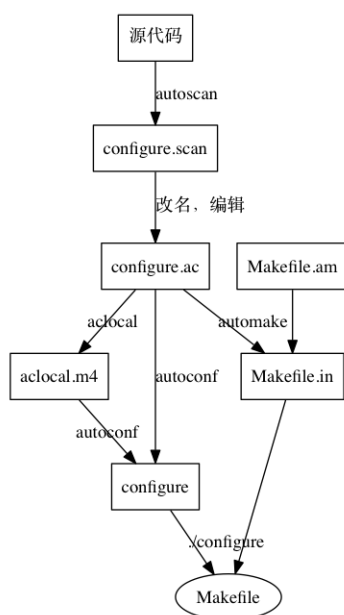
```
A -- B;
A -- C;
A -- D;
A -- E;
A -- F;
A -- G;
A -- H;
A -- I;
A -- J;
B -- C;
B -- D;
B -- E;
B -- F;
B -- G;
B -- H;
B -- I;
B -- J;
C -- D;
C -- E;
C -- F;
C -- G;
C -- H;
C -- I;
C -- J;
D -- E;
D -- F;
D -- G;
D -- H;
D -- I;
D -- J;
E -- F;
E -- G;
E -- H;
E -- I;
E -- J;
F -- G;
F -- H;
F -- I;
F -- J;
```

```

G -- H;
G -- I;
G -- J;
H -- I;
H -- J;
I -- J;
}

```

又比如下图：



源文件加上空行也只有22行（为了在手机上减少换行，取消了缩进）：

```

digraph G {
splines = false;

S [label="源代码" shape = "box"];
SCAN [label = "configure.scan" shape = "box"];
IN [label = "configure.ac" shape = "box"];
ACLOCAL [label = "aclocal.m4" shape = "box"];
configure [shape = "box"];
MakefileAm [label = "Makefile.am" shape = "box"];
MakefileIn [label = "Makefile.in" shape = "box"];

S->SCAN [label = "autoscan"];
SCAN -> IN [label = "改名, 编辑"];
IN -> ACLOCAL [label = "aclocal"];
ACLOCAL -> configure [label = "autoconf"];
AC -> MFI [label = "automake"];
MA[Makefile.am] -> MFI[Makefile.in];
CONF[configure] -> MF((Makefile)) [label = "xconfigure"];
MFI -> MF

```



```
IN -> configure [ label = "autoconf"];
MakefileAm -> MakefileIn;
IN -> MakefileIn [ label = "automake"];

MakefileIn -> Makefile;
configure -> Makefile [label = "./configure"];
}
```

而所有的dot文件，只是靠一个简单的Makefile和一个make命令就会部都转成PNG图片了。同时，dot文件还可以存在Git仓库里。

谁能告诉我，像下面这样的图（如图3-1）用OmniGraffle或Windows上的Visio画累不累呢？

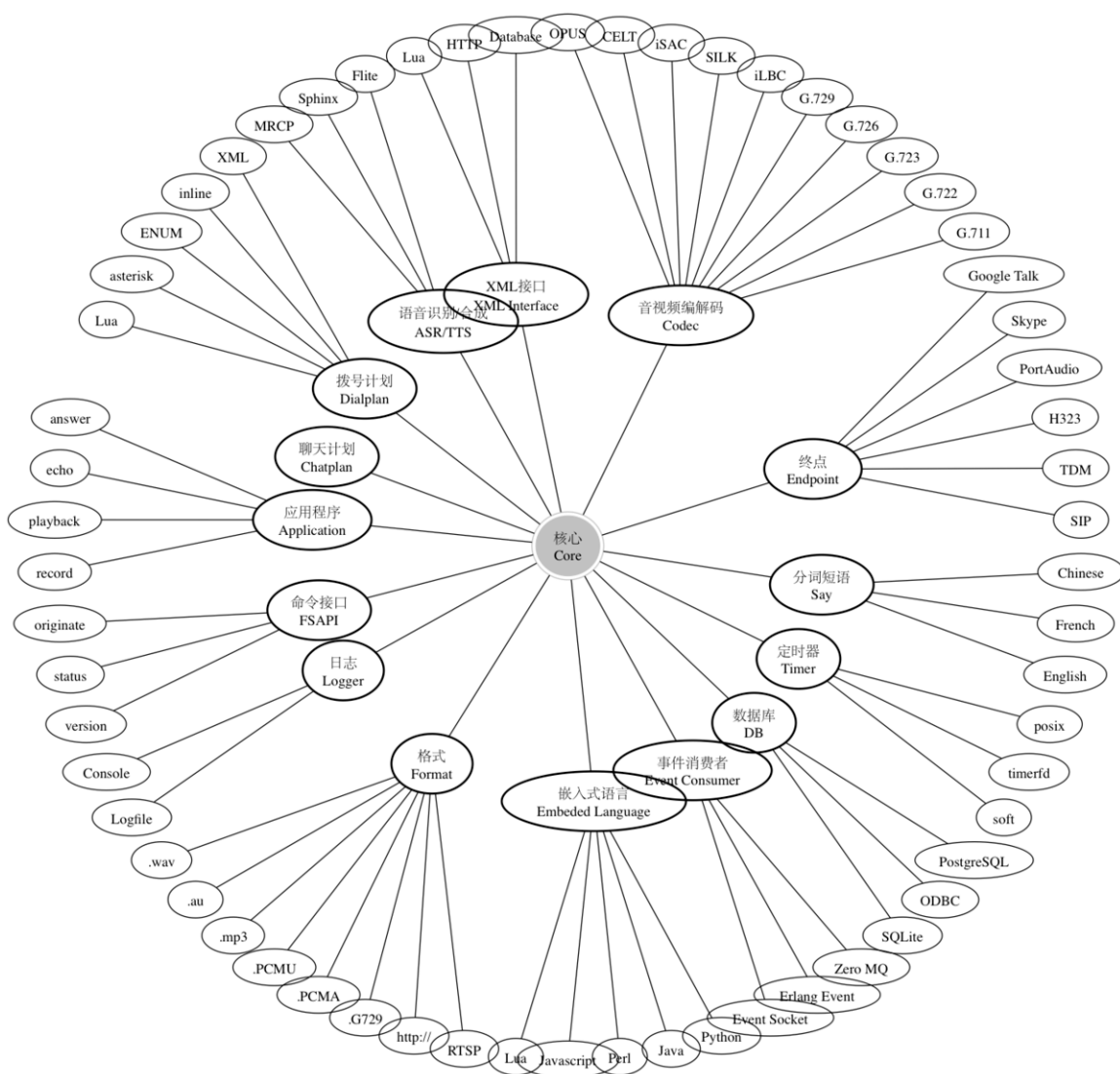


图 3-1: FreeSWITCH模块结构

当然，并不是所有插图都适合用Graphviz。所以，除此之外，我还使用Keynote。Keynote比较适合画这样的图（是的，Internet也是我用弧线画出来的，如图3-2）：

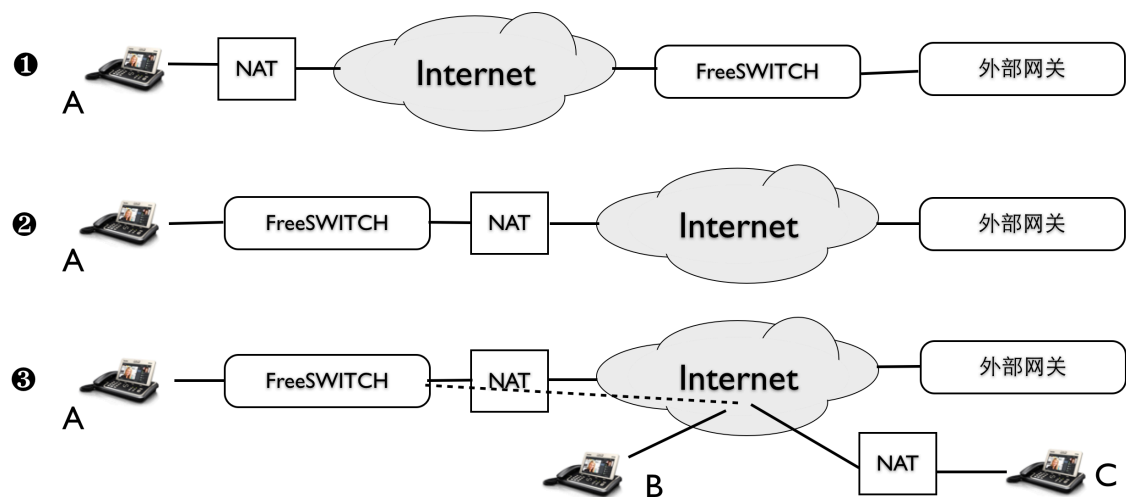


图 3-2: 用Keynote画的NAT示意图

还有一个问题。那就是，以上工具都不适合画时序图。因此，我不得不又搜遍互联网，找到一个类似Graphviz的工具——mscgen。是的，跟我期望的一样，它可以使用类似dot的语法画时序图，大概就是下面这个样子，如图3-3：

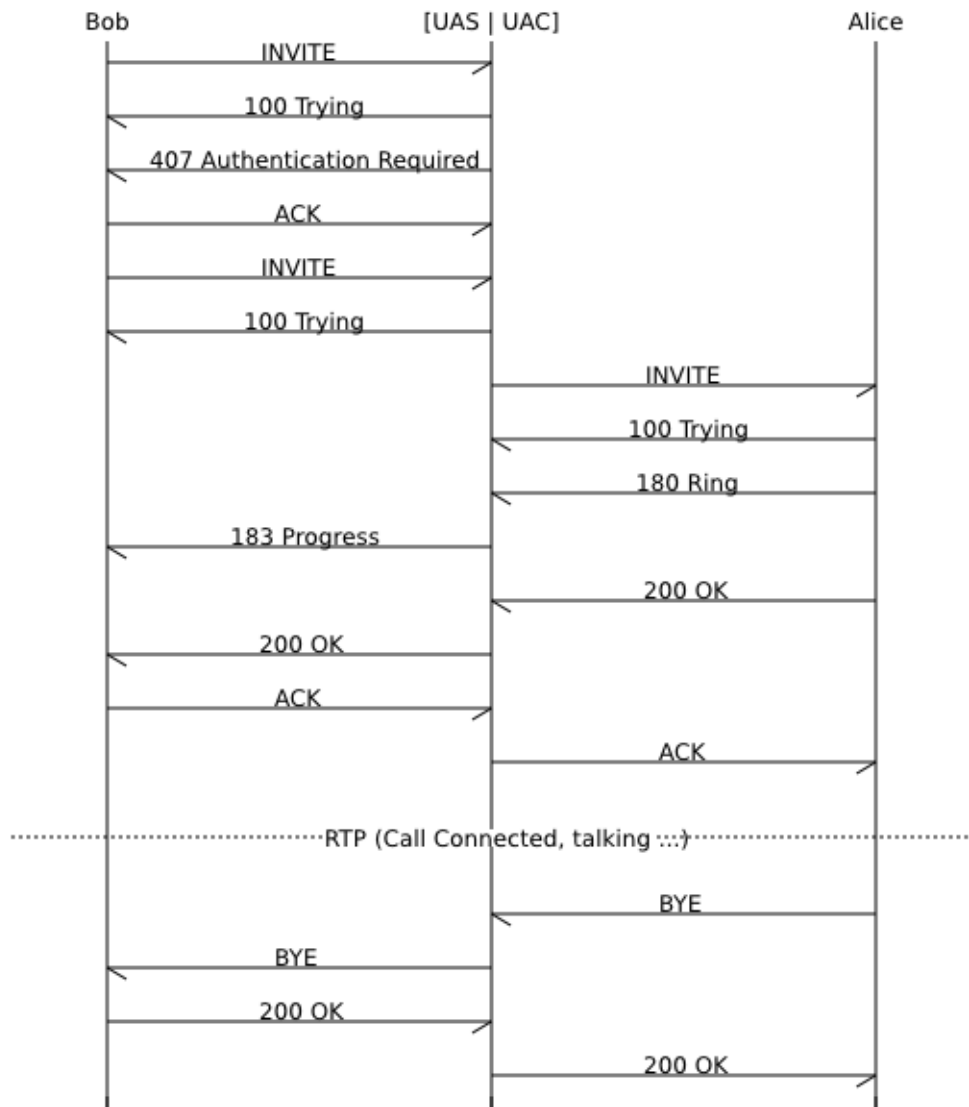


图 3-3: SIP信令时序图

搞过通信的人应该对上面那种图都非常熟悉。当然mscgen在Mac上生成图时对中文的处理有点问题。我只好首先用mscgen生成svg，然后又用Inksape将svg转换成PNG。

你还在用Word写API文档和产品说明文档吗？反正我们已经全部是Markdown了。虽然Word也可以进行『文件比较』，但怎么也不如git diff来得顺手吧？

好了。这基本上就是全部的秘密了。如果大家感兴趣，我以后也可以跟大家分享一些相关的脚本和源文件。程序员万岁。

第四章 插图和公式

Markdown支持原生的插图和公式。也可以针对不同的输出格式，使用不同的插图和公式语法。

4.1 插图

在Markdown中，插入一张图片很简单，语法如下：

```

```



图片默认会以页面大小100%宽度显示，可以使用如下方法调整图片的宽度（这在长图片排版时非常有用，可以防止图片显示过大）：

```
{ width=50% }
```



也可以插入带标题（caption）的图片，图片将会自动生成编号：

```
![这是一个有标题的图片](img/typesetting2.png){ width=50% }
```

技术图书排版

图 4-1: 这是一个有标题的图片

可以在正文中引用图片，如图4-2所示（注意，该方法在docx格式中不好用）。

```
![引用该图片\label{fig:typesetting}](img/typesetting2.png){ width=50% }
```

技术图书排版

图 4-2: 引用该图片

通过 `diagram-generator.lua`，可以直接在Markdown中内嵌流程图。如下面的代码可以生成一个有向图：

```
```graphviz
digraph G {
 rankdir=LR
 r[color="red"]
 g[color="green"]
 b[color="blue"]
 r -> g -> b
}
```
```

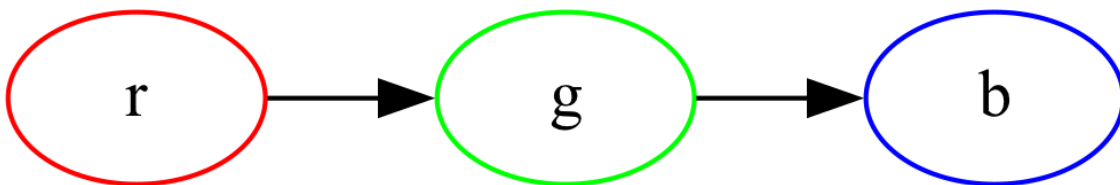


图 4-3

也可以使用Mscgen画图，代码和图如下：

```

```msc
msc {
 alice, bob;

 alice -> bob[label="Hello World!"];
 bob -> alice[label="你好，世界! "];
}
```

```



图 4-4

可以给图片源加标题，如：

```

```{.graphviz caption="这是一张有标题的图片"}
digraph G {
 rankdir=LR
 a -> b -> c
}
```

```

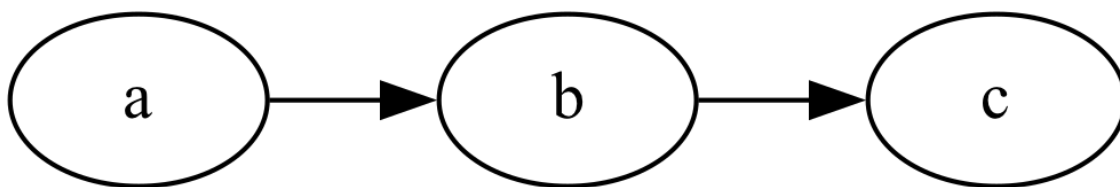


图 4-5: 这是一张有标题的图片

注意：只有具有标题的图片才会自动生成图片编号。

也可以这样引用图片，如图4-6所示。

```

```{.graphviz caption="\label{fig:example-1}这是另一张有标题的图片"}
graph G {

```

```

rankdir=LR
a -- b -- c
}
...

```

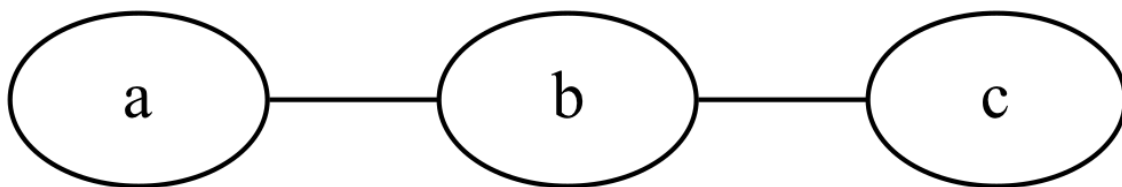


图 4-6: 这是另一张有标题的图片

自Pandoc 3.0版（2023年初发布）起，支持如下格式的语法（这样看起来更一致，可以与上面的写法对比其异同）：

```

```graphviz {caption="\label{fig:example-1}这是另一张有标题的图片"}
graph G {
    rankdir=LR
    a -- b -- c
}
...

```

我们使用自己搭的Gitea¹服务器管理我们的Markdown。为了能直接在Web界面上显示上述图片，我们写了一个浏览器插件：<https://git.xswitch.cn/xswitch/giteaBar>，供大家参考。当然，团队中每个人都装插件比较麻烦，因此，我们弃用了上述插件，并[自己定制了一版](#)。

Github已经支持使用Mermaid²画图。直接使用如下语法即可。

```

```mermaid
graph TD;
 A-->B;
 A-->C;
 B-->D;
 C-->D;
...

```

但我们的PDF中暂时还不支持这个语法，主要是Docker镜像已经很大了，如果再加上Mermaid，就会更大，而且，比起来，Mermaid来的图并不怎么好看。

<sup>1</sup>参见 <https://gitea.io/>。

<sup>2</sup>参见 <https://mermaid-js.github.io/mermaid/#/>。

后来，我还学会了使用Gnuplot<sup>3</sup>画图。下面的图是我画的圆与正弦波的图。以后有了时间，我也会讲讲我是怎么画的。

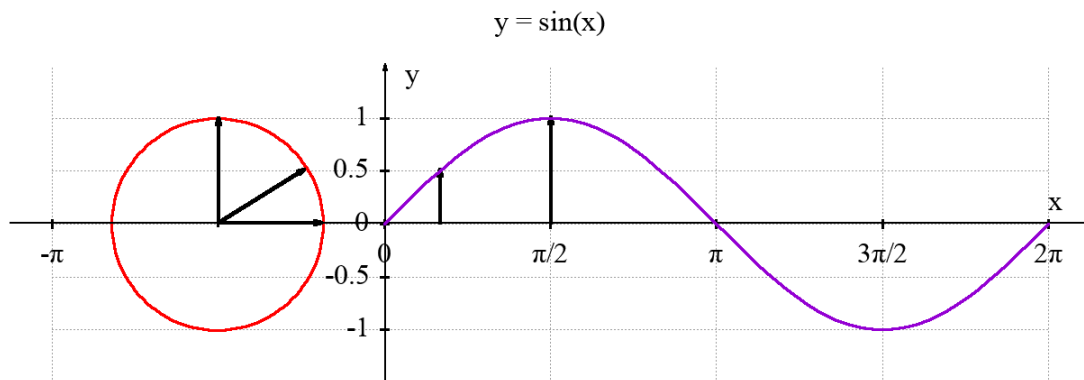


图 4-7: 圆与正弦波

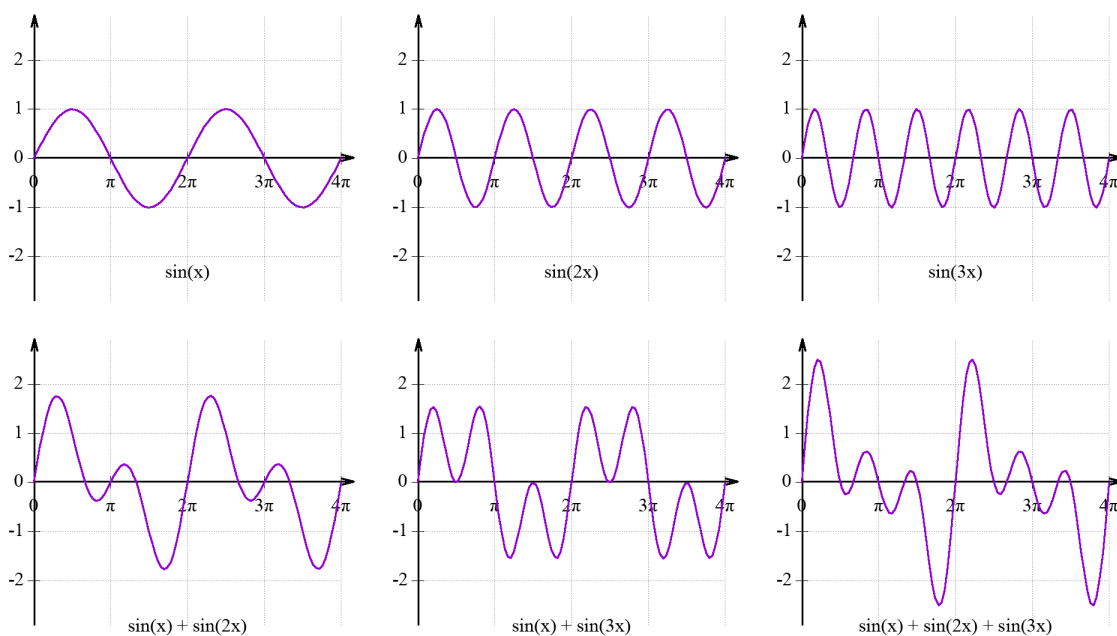


图 4-8: 正弦波叠加

## 4.2 公式

Markdown支持行内公式，简单的公式可以使用原生的上下标格式写。上标写为  $x^2$ ，显示为  $x^2$ ，下标写为  $x_2$ ，如著名的质能方程  $E = mc^2$  可以写为  $E = mc^2$ 。这种原生方式支持

<sup>3</sup><http://www.gnuplot.info/>。



的输出格式比较多，如HTML、PDF、**docx**等。如果公式比较复杂，也可以使用Latex语法的公式<sup>4</sup>，语法是`$公式语法$`，注意前`$`后面不要有空格，后`$`前面不要有空格。如`$x^2 + y^2 = z^2$`显示为 $x^2 + y^2 = z^2$ 、`$E = mc^2$`显示为 $E = mc^2$ 、欧拉公式`$e^{i\pi} + 1 = 0$`显示为 $e^{i\pi} + 1 = 0$ 等。Latex格式的公式对PDF支持比较好，**docx**中也可以正常显示，在HTML中有些能正常显示，有些需要配合MathML或MathJax生成公式。

也可以使用如下语法显示独立的公式（单独在一个段落中）：

---

```
$$
x^2 + y^2 = z^2
$$
```

---

$$x^2 + y^2 = z^2 \quad (4-1)$$

在PDF中，配合`diagram-generator.lua`会对公式自动编号。

下面是一些公式示例，可以自行观察在不同输出格式（文件类型）中的效果。

---

```
\begin{equation}
\begin{aligned}
a &= b \\
a1 &= b1
\end{aligned}
\end{equation}
```

---

$$\begin{aligned} a &= b \\ a1 &= b1 \end{aligned} \quad (4-2)$$

只有以下方式能出现在docx中，且公式在docx和pdf中都没有编号：

---

```
$$
\begin{aligned}
a &= b \\
a2 &= b2
\end{aligned}
$$
```

---



---

<sup>4</sup>参见 <https://zh.wikipedia.org/wiki/Help:数学公式>。

$$\begin{aligned} a &= b \\ a^2 &= b^2 \end{aligned} \tag{4-3}$$

下列公式在PDF中正常，在docx和HTML中不显示：

---

```
\begin{align}
a &= b \\
a^3 &= b^3
\end{align}
```

---

$$a = b \tag{4-4}$$

$$a^3 = b^3 \tag{4-5}$$

---

```
\begin{align}
\begin{split}
a &= b \\
a^4 &= b^4
\end{split}
\end{align}
```

---

$$\begin{aligned} a &= b \\ a^4 &= b^4 \end{aligned} \tag{4-6}$$

下面是一些有趣的公式：

---

```
$$
f(x)=\left\{\begin{align}
1,x>0\\
0,x=0\\
-1,x<0
\end{align}\right.
$$
```

---

分段函数：

---

```

$$
f(x)=\left\{\begin{aligned}
&1, x>0\\
&0, x=0\\
&-1, x<0
\end{aligned}\right.
$

```

---

$$f(x) = \begin{cases} 1, x > 0 \\ 0, x = 0 \\ -1, x < 0 \end{cases} \quad (4-7)$$

麦克斯韦方程组<sup>5</sup>:

---

```

$$
\begin{aligned}
&\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0} \\
&\nabla \cdot \mathbf{B} = 0 \\
&\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \\
&\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t}
\end{aligned}
$

```

---

$$\begin{aligned}
 \nabla \cdot \mathbf{E} &= \frac{\rho}{\epsilon_0} && \text{高斯定律} \\
 \nabla \cdot \mathbf{B} &= 0 && \text{高斯磁定律} \\
 \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} && \text{法拉第电磁感应定律} \\
 \nabla \times \mathbf{B} &= \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} && \text{麦克斯韦 - 安培定律}
 \end{aligned} \quad (4-8)$$

其中, 如果把 `\frac` 替换为 `\cfrac`, 则在浏览器中不能正常显示。

麦克斯韦方程组的积分形式如下:

---

```

$$
\begin{aligned}
&\oiint_S \mathbf{D} \cdot d\mathbf{s} = Q_f \\
&\oiint_S \mathbf{B} \cdot d\mathbf{s} = 0
\end{aligned}
$

```

---

<sup>5</sup>参见 <https://www.zhihu.com/question/25121612> 及 <https://zh.wikipedia.org/wiki/馬克士威方程組>。

```

\oint_{L} \, E \cdot dl &= -\frac{d \Phi_B}{dt} \\
\oint_{L} \, H \cdot dl &= I_f + \frac{d \Phi_D}{dt} \\
\end{aligned}

```

$$\begin{aligned}
 \oiint_S D \cdot ds &= Q_f \\
 \oiint_S B \cdot ds &= 0 \\
 \oint_L E \cdot dl &= -\frac{d\Phi_B}{dt} \\
 \oint_L H \cdot dl &= I_f + \frac{d\Phi_D}{dt}
 \end{aligned} \tag{4-9}$$

矩阵：

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \left\{ \begin{matrix} a & b \\ c & d \end{matrix} \right\} \quad \begin{vmatrix} a & b \\ c & d \end{vmatrix} \quad \left\| \begin{matrix} a & b \\ c & d \end{matrix} \right\| \tag{4-10}$$

求和：

$$f(x) = \sum_{n=0}^{\infty} a_n x^n = a_0 + a_1 x + a_2 x^2 + \cdots \tag{4-11}$$

付里叶级数：

$$F(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} [a_n \cos(n\omega t) + b_n \sin(n\omega t)] \tag{4-12}$$

付里叶级数的复数形式：

$$\begin{cases} \cos(n\omega t) = \frac{e^{jn\omega t} + e^{-jn\omega t}}{2} \\ \sin(n\omega t) = \frac{e^{jn\omega t} - e^{-jn\omega t}}{2} \end{cases} \tag{4-13}$$

**注：**本节中的公式在不同的输出格式中不一定能正常显示，可以对比本书不同的版本如HTML、PDF、[docx](#)等查看区别。

## 第五章 写作点滴

我写过两本书《FreeSWITCH权威指南》和《Kamailio实战》，也有一些新书在写作中。今天，就来跟大家分享一下我写作的一些经验、方法和工具等，希望对大家有所帮助。

首先是写作方法。写作的方法有千千万，每个人都有自己的写作方法，我使用的技巧其实并不多，主要就是平时注意收集素材，然后当真正写的时候就有素材了，整理一下，就知道要写那些东西。

比如说我在平时的工作中，尤其是在处理一些问题的时候、在对客户进行支持的时候，经常会遇到各种各样的问题，这时我就会把一些案例记下来。如果有时间，就简单做一下整理，如果没有时间，那就先把原始记录、日志等记下来，等以后有时间再去整理。

这样到了真正写作的时候就可以先设计一下章节结构，把那些案例套进去。基本就是一篇文章或者一本书的草稿也就完成了。

总之就是要多积累素材，仓中有粮，心里不慌。

我最开始写《FreeSWITCH权威指南》的时候，其实那时候还没有《权威指南》。当时主要是写博客，然后就是把工作中遇到一些案例，整理成博客文章，发到网站上。这样后来就慢慢积累了很多的博客，当然在积累的过程中我也有意识地按一本书的样子添加相关的内容。然后大家看了说不错，就有人鼓动我写一本书。然后我就自己排了排版，印了一些，还卖了不少。

后来就对接了出版社。最早找了图灵，但图灵想先出电子版，而我比较想先出纸质版，后来华章同意出纸质版。然后跟出版社的编辑讨论了具体的章节目录，正式写作就开始了。不过真正写书跟写博客完全不一样。写博客比较简单、比较随意，但是写书的话就需要比较严谨，逻辑结构也得清晰、注意前后呼应。最开始想得很简单，觉得博客都写得差不多了，顶多是再补充一些章节，后来发现还差了一大半，又加上编辑排版，着实花了不少心力。虽然那时出版社的编辑没有天天催稿，但是心里的压力还是比较大的。但最终还是完成了，书出来后大家反响还不错。

当我写第二本书《Kamailio实战》的时候就有些经验了，我等到基本上写完了之后又拿给编辑看的，他们还说杜老师你看你连排版都做好了，把我们的很多工作都做了（实际上他们还是要重新排版的）。

当然，不管你准备地多么完善，其实还是差很多，后面也还校对修改了很多次。总起来，这本书前前后后还是花了七个多月的时间。图书的出版流程本来比较慢，尤其是今年加上疫情等各种原

因，我们基本上在各个环节都遇到了一些问题，结果导致书出来就比较慢，又过了五个多月大家才拿到书，这样总起来就是一年多。

下面主要是跟大家分享一下，我写作中使用工具之类的，希望对大家有所帮助。之前我也写过一篇文章《我在Mac上的写作工具链》，感兴趣的同学可以看一下，这里相当于再做一些补充。

我现在写作基本是使用VSCode和Markdown。首先我本身是个程序员，VSCode是常用的代码编辑器，Markdown是一个简洁的文档格式，我平常积累素材也都是它们。提交到Git仓库里，内容也不会丢，还有版本控制。另外，做为技术书，我的书里有大量的代码，而Markdown对程序员非常友好。

但是出版社还是不会使用Markdown，还是需要Word版的，所以说我一般都是写得差不多了，先拿PDF给编辑看，然后在沟通差不多之后再转成Word格式。Word中有一个“修订”功能非常有用，便于与编辑的沟通。在这一点上Git还是不方便。但Word文档要传来传去，有时我们也用WPS，但我使用macOS，编辑使用Windows，总有些小的格式和字体问题无法很好地解决。好在格式不是重点，出版之前所有格式也都还会重排。现在出版社也开始在使用WPS之类的在线文档方便协同，减少传来传去的麻烦。

不过文档一旦转成Word格式，再转回Markdown就比较困难了，因此，我的Markdown还是停留在草稿状态，因为有一些最终的修改也不会再同步回来。

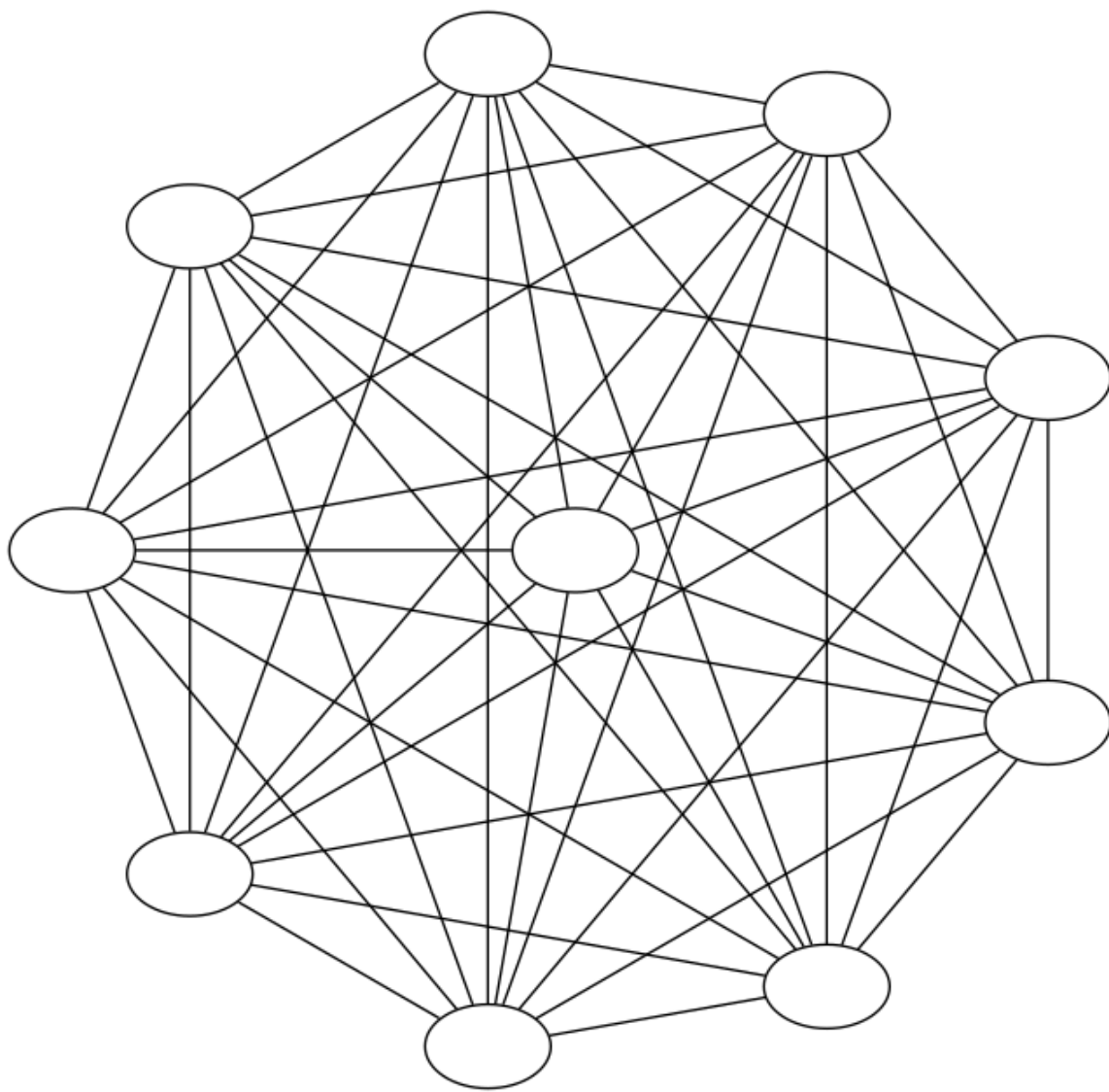
还有，这里有一个常识问题，就是我实际上是没有最终的电子版的。出版社最终的版本校对也是打印出来使用纸质版校对的。这可能主要是为了留下校对的“痕迹和证据”。另外，直接在纸上勾勾画画确实比在电子文档上做标记来得快。

当然，这里说到常识的主要原因是，很多人忽略了我没有电子版这个常识。我自己买自己的书的电子版也是需要花钱的，主要是电子书只能在一些数字出版平台如京东、以前的亚马逊、当当、微信读书之类的平台上，而不是直接发给我一个PDF。国内的盗版还是比较严重，电子书就更容易被盗版。所以，如果大家看到这里，就不要跟我要电子书了，首先是我没有，其次是即使有也不能给到你。

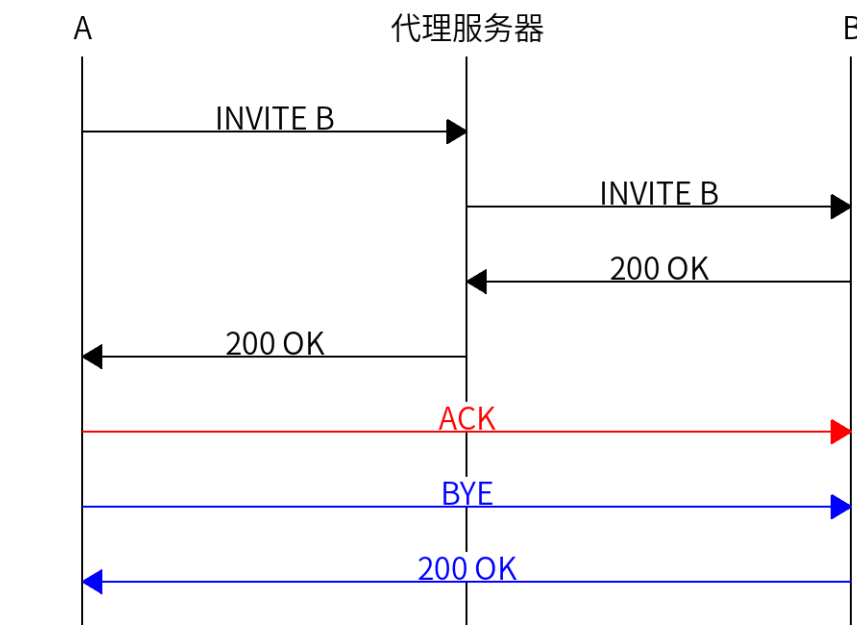
回归主题，再说Markdown，关于Markdown，我还写了几篇文章，然后发到我们公司的网站上，大家感兴趣可以看一下[《小樱桃的Markdown规范》](#)。Markdown有很多版本，有时候为了排版和表达又做了一些扩充，我们也总结了一些比较好的使用经验，即保持简单，又能适当表达我们想要的格式。

下面再说下插图。不管什么样的文档一般或多或少都会有一些插图。画插图，我还是喜欢你[Graphviz](#)和[Mscgen](#)，我现在又学了一个新的技能[Gnuplot](#)。

下图是Graphviz画的：

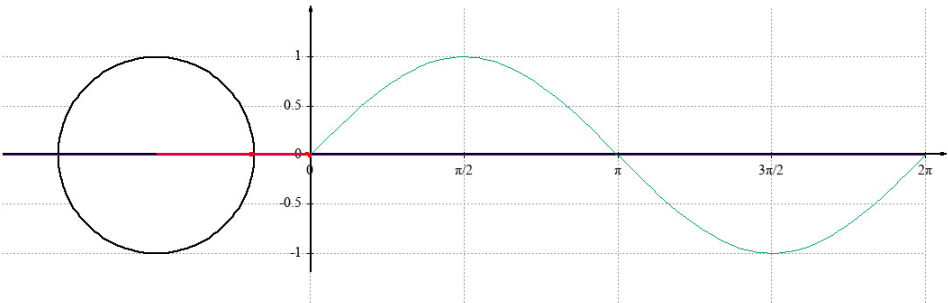


下面是Mscgen画的图：

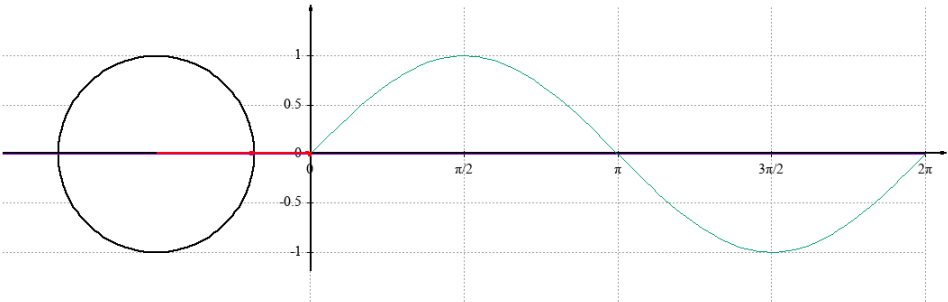


做数字信号处理就不可避免地要讲付里叶变换，讲付里叶变换就得讲正弦曲线，所以我就用Gnuplot画了下面的图，包括动图：

JPG:



GIF:



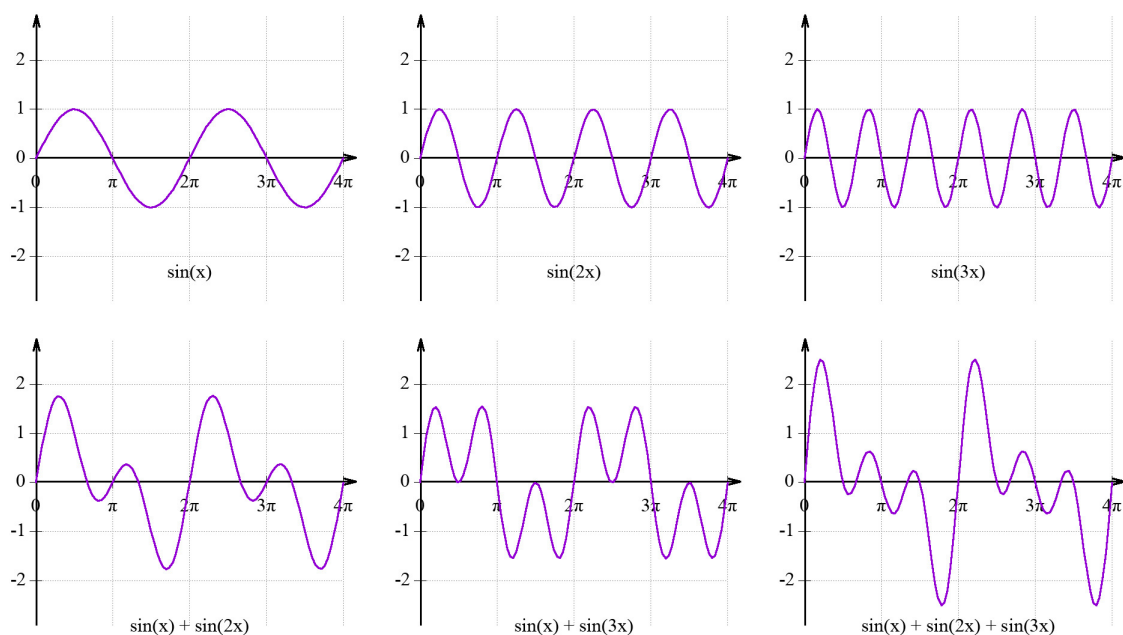
WebP:

!PDF版不支持该类型的图片[webp image]!

下面的视频是webm格式的，内容跟上面的动图一样，有的浏览器可能不能正常播放（在PDF版



中也不可见)。



画合适图要使用合适的工具，每个工具都有它的长处。关于这些软件的使用方法和绘图的源文件示例，我后面有时间会整理出来。

我是个程序员，因此比较喜欢用代码生成图，这样更“精确”，且风格统一。而且使用这些代码还有一个好处就是我可以直接把代码嵌入到Markdown里，在生成PDF、Word文档或网页时可以自动生成图。这样所有的代码都Commit到Git仓库时里，以后再找原始文件修改的时候就很方便。而与之相对的，如果用了其它专业的作图工具生成图，由于图以及图的源文件一般都是二进制的，不适合存到Git仓库里，容易散落到各个地方导致以后找不到。

当然，为了能较好地排版，我还是花了不少心力的。一方面是我希望排成比较好看的电子书，另一方面就是我也希望我们产品文档能适合同时在网站上展示，同时能生成PDF。作为一款2B的产品，我们不可避免地还是要分发PDF的手册。

Markdown的好处其实主要是让你整理思路，把章节，重点、代码等用简单的标记标记出来，而无须过多关注排版。但无论如何还是要排版的，因为要给你看的话，别人看到的还是最终效果。我做了一些排版的模板，可以大致兼容比较多的场景，也能生成比较好的PDF、docx和HTM文档，感兴趣的同学可以看看[技术图书排版](#)。

另外，我还对mscgen打了补丁：<https://github.com/seven1240/mscgen>。主要是解决了图片分辨率的问题，在现代的视网膜显示器上，必须至少使用2倍的分辨率才能显示得比较清晰。而且印刷也需要大约300dpi的分辨率。原始的Mscgen是一个比较古老的工具，也不知道去哪里提交补丁，只好自己做了放到Docker镜像里。

除了这些工具外，还有一个比较新的流程图工具是[Mermaid](#)。Mermaid貌似功能更强大，而且好像Github的Markdown对它还有内建的支持。然而其工具链依赖于Javascript以及Headless Chrome等，装在Docker里非常耗资源，因而我也一直没有用。并且我也总觉得它画得图不如Graphviz和

Mscgen画得好看。

最近在用墨问便签，体验了文字输入，感觉很方便。然后我就又学了一个新技能，在macOS上使用语音输入。开启的方法也很简单，在「设置」→「键盘」里就可以开启，开启后选择中文就可以了。然后每次启用语音输入就按两下Fn那个键，倒也方便。

之前我一直都是使用键盘输入的，我用五笔打字，一般速度也比较快，也没有动力使用语音输入，总感觉准确度不高又比较麻烦。另外就是因为我经常打字会有很多英文和代码，使用语音输入的话，中英文混合识别效果就比较差。但经过这次尝试，我觉得还是需要尝试一下新东西，而且，现在感觉或许是有点儿年纪大了，五笔输入法有个问题，就是早年记住字很容易打出来，不会打的字总打不出来，即使查了怎么打也不容易记住，很多时候还是要切换到拼音输入拼出来（这当然不能掩盖五笔输入法的好处，那就是不认识的字也能根据笔画打出来）。这就更有理由尝试使用语音输入了。反正语音输入也是先大致打个草稿，回头还需要再修正，遇到英语有时也只能先跳过去，回头再补。这篇文章就是断断续续用语音输入的，也许是刚开始不大熟练，感觉其实思路并没有那么快，语音讲出来的东西还是比较口语化，即使识别率正确，回头需要修改的内容也比较多。

最后还是回到我新出的这本书啊——《Kamailio实战》。书刚印出来的时候出版社的仓库赶上封控，无法发货，因此我们把部分书从印刷厂直接运到了我们公司，一部分放到京东的仓库给大家发书，使用的是我们微信上的《小樱桃商城》。我们的商城比较简单，物流信息不能闭环。现在[出版社官方的天猫店](#)也上架了这本书，多了另一个购买选择，后面也陆续会有更多的商城上架，希望大家都能方便地买到书。

最最后，希望我的书以及本文能对大家有所帮助，也欢迎大家批评。

本文首先发表在[我的个人博客——杜金房的书](#)，转载请注明出处。

## 第六章 Pandoc安装与使用

我一般都使用Pandoc将Markdown文件生成PDF与HTML。这里简单记录一下我的安装和使用经验，供参考。

### 6.1 安装Pandoc

一般来说，安装Pandoc可以直接按官方的方法安装：<https://pandoc.org/installing.html>。

### 6.2 安装Latex

如果你只是生成 `docx` 和HTML格式的文件，则不需要安装Latex，但如果需要生成PDF，就需要。

安装Latex比较麻烦，而且占用很大的空间（完成安装要3~5G）一般有如下两种安装方式：

- MacTeX：完整安装，参见 <https://www.tug.org/mactex/>
- BasicTex：小型安装，参见：<https://www.tug.org/mactex/morepackages.html>
- TinyTex：另一个小型安装方法，参见：<https://yihui.org/tinytex/>

如果你不想让Latex“污染”本地的系统，又熟悉Docker，最好使用后面的Docker运行Pandoc和Latex，不过，Docker要稍慢一些。

#### 6.2.1 在macOS上安装Pandoc和Latex

在macOS上安装Pandoc非常简单，只需要执行如下命令即可：

---

```
brew install pandoc
```

---

安装Latex：

---

```
brew install --cask basictex
```

---

安装后会环境影响变量，打开一个新Shell检查如下命令是否可以正常执行：

---

```
latex -v
xelatex -v
pdflatex -v
```

---

如果还有问题，可以找到 [basictex](#) 的安装的Package，双击重新安装，安装包的名称因时间不同，笔者的路径如下：

---

```
/opt/homebrew/Caskroom/basictex/2022.0314/mactex-basictex-20220314.pkg
```

---

使用如下命令找到的（注意，旧版的Homebrew会将软件安装在 [/usr/local/](#) 而不是 [/opt/](#)）：

---

```
brew list basictex
find /opt -name mactex-basictex-20220314.pkg
```

---

如果你使用本书的Latex模板，则需要下载一些对应的中文字符，这些字体大部分可以在下面的网址中找到。为了方便使用这些字体，我本地维护了一个Makefile和一个 [.cache](#) 目录，通过 [make cache](#) 就能下载这些字体。

---

```
.cache:
 mkdir .cache
cache: .cache
 cd .cache && \
 cp ../sources.list* . && \
 cp ../install-bin-unix.sh . && \
 curl -L -o NotoSansCJK-Bold.ttc https://github.com/seven1240/font/raw/master/noto/NotoSansCJK-Bold.ttc/
↪ NotoSansCJK-Bold.ttc && \
 curl -L -o NotoSansCJK-DemiLight.ttc https://github.com/seven1240/font/raw/master/noto/
↪ NotoSansCJK-DemiLight.ttc/NotoSansCJK-DemiLight.ttc && \
 curl -L -o NotoSerifCJK-Regular.ttc https://github.com/seven1240/font/raw/master/noto/
↪ NotoSerifCJK-Regular.ttc/NotoSerifCJK-Regular.ttc && \
 curl -L -o NotoSansMonoCJKsc-Regular.otf https://github.com/seven1240/font/raw/master/noto/
↪ NotoSansMonoCJKsc/NotoSansMonoCJKsc-Regular.otf && \
```

```

curl -L -o SourceCodePro-Regular.ttf https://raw.githubusercontent.com/adobe-fonts/source-code-pro/
↪ release/TTF/SourceCodePro-Regular.ttf && \
curl -L -o SourceCodePro-It.ttf https://raw.githubusercontent.com/adobe-fonts/source-code-pro/release/
↪ TTF/SourceCodePro-It.ttf && \
curl -L -o adobekaitistd-regular.otf https://github.com/seven1240/font/raw/master/adobe/
↪ adobekaitistd-regular.otf && \
curl -L -o adobefangsongstd-regular.otf https://github.com/seven1240/font/raw/master/adobe/
↪ adobefangsongstd-regular.otf && \
curl -L https://github.com/jgm/pandoc/releases/download/2.19.2/pandoc-2.19.2-linux-amd64.tar.gz --output
↪ pandoc-2.19.2-linux-amd64.tar.gz && \
curl -L https://github.com/jgm/pandoc/releases/download/2.19.2/pandoc-2.19.2-linux-arm64.tar.gz --output
↪ pandoc-2.19.2-linux-arm64.tar.gz

```

cache-google:

```

cd .cache && \
curl -L -o NotoSansMonoCJKsc-Regular.otf https://github.com/googlefonts/noto-cjk/raw/main/Sans/Mono/
↪ NotoSansMonoCJKsc-Regular.otf && \
curl -L -o NotoSansMonoCJKsc-Bold.otf https://github.com/googlefonts/noto-cjk/raw/main/Sans/Mono/
↪ NotoSansMonoCJKsc-Bold.otf

```

cache-sara:

```

cd .cache && \
curl -L -o sarasa-mono-sc-nerd-bold.ttf https://github.com/laishulu/Sarasa-Mono-SC-Nerd/raw/master/
↪ sarasa-mono-sc-nerd-bold.ttf && \
curl -L -o sarasa-mono-sc-nerd-regular.ttf https://github.com/laishulu/Sarasa-Mono-SC-Nerd/raw/master/
↪ sarasa-mono-sc-nerd-regular.ttf

```

字体下载后，找到相应的字体文件（注意 `.cache` 在 macOS 上默认是个隐藏目录，可以 `ls -l` 查看，或 `open .cache` 打开），双击相应的字体文件即可以安装字体。

**注意：**在2023年初的macOS版本中（我的是13.1 (22C65)），不知道是macOS本身的问题还是字体的问题，有些Noto Sans字体会相互覆盖，这时候就不要双击安装，把字体文件直接拖到 `~/Library/Fonts` 目录下即可（可通过Font Book菜单上的【文件】⇒【在Finder中打开】）。

安装完字体后，可以使用 `fc-list` 查看是否安装成功。如：

```

fc-list
fc-list | grep Noto

```

Latex和字体安装成功后，还需要安装宏包。具体的宏包名称可以在 `make` 的时候看到，如果缺少对应的宏包就会出错，这时可以使用 `tlmgr` 安装。如：

```

tlmgr install longtable
sudo tlmgr install longtable # 有时候需要使用sudo

```

## 6.2.2 在Linux上安装Pandoc和Latex

如果你使用Debian或Ubuntu，可以参考如下命令：

---

```
apt-get update && apt-get install -y \
--no-install-recommends --no-install-suggests \
graphviz mscgen ttf-mscorefonts-installer pandoc \
libgmp10 texlive-xetex lmodern texlive-fonts-recommended wget fontconfig \
make ca-certificates locales xz-utils \
&& localedef -i en_US -c -f UTF-8 -A /usr/share/locale/locale.alias en_US.UTF-8 \
&& mkdir /root/texmf && cd /root/texmf && tlmgr init-usertree \
&& tlmgr install zhspacing \
&& tlmgr install changepage \
&& tlmgr install ulem \
&& tlmgr install soul
```

---

如果你需要最新版本Pandoc，可以直接下载Tar包解压安装，如：

---

```
tar xvzf pandoc-3.0.1-linux-amd64.tar.gz --strip-components 1 -C /usr/local
```

---

## 6.2.3 在Windowsh安装Pandoc和Latex

笔者没有使用Windows系统，请参照上面官方链接中的方法安装。如果你有好的经验和建议，也欢迎告诉我。

如果你没有经验或不想折腾，那建议使用Docker，参见<https://docs.xswitch.cn/xpedia/docker/>。

## 6.3 使用Docker

Pandoc安装和使用很简单，但是如果要生成PDF就需要安装Latex，而Latex非常庞大，而且安装比较麻烦，因此，我制作了一个Docker镜像，集成了大多数我用到的工具，方便大家也方便我自己使用。

你在本书的Makefile中应该可以看到如下内容：

---

```
docker:
docker run --rm -it -v `PWD`:/team ccr.ccs.tencentyun.com/free/pandoc:tiny-3.0.1 bash
```

---

基本的使用方法是执行 `make docker`，它会将当前目录挂载到镜像中的 `/team` 目录中，然后进入 Docker 容器的命令行，你可以在容器中执行任何命令，如 `make`，`make docx` 等。

上述镜像在腾讯云上。此外，我还制作使用过其他不同版本的镜像，具体如下：

- `multiarch`：arm64及amd64镜像，基于Ubuntu Jammy
- `m1`：arm64镜像，基于Ubuntu Jammy
- `tiny`：arm64及amd64，基于Debian Bookworm及<https://yihui.org/tinytex/>，镜像最小。
- `latest`：arm64及amd64，基于Debian Bookworm
- `3.0.1`：pandoc 3.0.1
- `tiny-3.0.1`：pandoc 3.0.1

---

```
$ docker image ls ccr.ccs.tencentyun.com/free/pandoc
```

REPOSITORY	TAG	IMAGE ID	SIZE
ccr.ccs.tencentyun.com/free/pandoc	3.0.1	3e18c37be4e3	1.1GB
ccr.ccs.tencentyun.com/free/pandoc	tiny-3.0.1	8cc1f2bb499b	829MB
ccr.ccs.tencentyun.com/free/pandoc	tiny	bebbeee87788	719MB
ccr.ccs.tencentyun.com/free/pandoc	latest	3036623e1d31	913MB
ccr.ccs.tencentyun.com/free/pandoc	m1	81902b42b585	849MB
ccr.ccs.tencentyun.com/free/pandoc	multiarch	ac93363c5ab3	739MB

---

上述内容是在2023年初的情况，本文档不保证实时更新，仅供参考。

## 写在最后

到此，我们该写一个后记了。如果你查看源文件，你可以看到从这里开始的相关的章节名字后面有 `{-}` 标记，它告诉LaTeX按章节排版但不再生成章节号。

本书所有源代码都可以在Github上找到：<https://github.com/seven1240/latex>。

用LaTeX排版，你几乎可以排出任何你想要的效果。但是，我们在本书中只使用了基本的LaTeX的模板，在表格、图文混排方面还有很多不尽人意。不过，我们的目标并不是做一个完美的排版，因为那样需要在正文中插入很多跟内容不相关格式代码，就背离了我们想使用Markdown把文章写的简单清新的初衷了。所以，我们只是尽力而为，做一个『足够好看』的电子书即可。

如果你的书真的要出版，出版社会帮你排版的。

LaTeX的学习曲线也是很陡的，而且在LaTeX中处理中文，就需要更多的技巧。所幸，现在中文都统一的UTF-8编码了，所有一定保证所有源文件都保存成UTF-8的。如前言中所述，本书不是希望教你成为一个排版专家，这些模板已经写好了，直接拿去用就可以了。当然，如果你真是排版专家，也欢迎提到Github上提 [PR](#) 帮我们做得更好。

我们还有以下问题没有解决：

- 在正文中自由切换字体比较困难，如临时变成楷体、仿宋等。
- 表格，我想默认设成100%宽度。

---

这是	一张表格
----	------

---

内容	太短了不好看
----	--------

不知	有没有办法宽度100%
----	-------------

---



## 作者简介

**杜金房：**（网名：Seven Du），《FreeSWITCH权威指南》<sup>1</sup>、《Kamailio实战》作者、FreeSWITCH中文社区<sup>2</sup>创始人，FreeSWITCH开源项目<sup>3</sup>核心Committer，开源爱好者。北京信悦通科技和烟台小樱桃科技<sup>4</sup>创始人。腾讯云TVP。

---

<sup>1</sup><http://book.dujinfang.com>，2014年出版。

<sup>2</sup><http://www.freeswitch.org.cn>

<sup>3</sup><https://freeswitch.com>

<sup>4</sup><http://x-y-t.cn>

## 版权声明

本书版权归作者所有，保留所有权利。本书相关的模板代码采用创作共用CC-BY-SA<sup>5</sup>发布。

---

<sup>5</sup>[https://zh.wikipedia.org/wiki/Wikipedia:CC-BY-SA\\_3.0协议文本](https://zh.wikipedia.org/wiki/Wikipedia:CC-BY-SA_3.0协议文本)>

# 广告

## 关于广告的广告

请允许我在本书中发布广告。承接其它广告。广告合作联系邮箱：info@x-y-t.cn。

## XSwitch

**XSwitch是一个高度可定制的音视频通信平台** <https://xswitch.cn>

XSwitch是一个SSaaS（Soft-Switch as a Service）平台，可以用来：

- 打电话
- 电话会议
- 视频会议
- 呼叫中心
- 录音录像
- 其它音视频互通等

支持私有化部署。

## 技术支持

烟台小樱桃网络科技有限公司提供商业FreeSWITCH、OpenSIPS及Kamailio技术支持。

- 网址：<http://x-y-t.cn>
- 邮箱：info@x-y-t.cn

下面是我们的微信公众号。为了能将两张图片排在一行上，直接使用了Latex代码。另外两张图片尺寸不同，所以两栏的宽度不是  $0.5:0.5$ ，而是  $0.55:0.44$ （注意两者加起来小于1，这主要是为

为了防止移动版图片放不开产生换行），其中`\linewidth`为行宽。图片只能在PDF中显示，在Word文档和HTML中无法生成。



(a) 小樱桃科技



(b) FreeSWITCH-CN

上述图片的Latex代码如下：

---

```
\begin{figure}
\begin{subfigure}{.55\linewidth}
\centering
\oldincludegraphics[width=.98\linewidth]{img/xyt1.jpg}
\caption{小樱桃科技}
\end{subfigure}
\begin{subfigure}{.44\linewidth}
\centering
\oldincludegraphics[width=.98\linewidth]{img/qr-wechat.png}
\caption{FreeSWITCH-CN}
\end{subfigure}
\end{figure}
```

---

以下图片仅在HTML中显示，代码如下：

---

```
<div>

</div>
```

---

## FreeSWITCH相关图书

- 《FreeSWITCH文集》收集了一些FreeSWITCH文章，相比其它FreeSWITCH书来说，技术内容比较少，便于非技术人员快速了解FreeSWITCH。
- 《FreeSWITCH互联互通》主要收集了一些互联互通的例子，书中有些例子来自《FreeSWITCH权威指南》。

- 《FreeSWITCH实例解析》收集了一些如何使用FreeSWITCH的实际例子，方便读者参考。书中有些内容来自《FreeSWITCH权威指南》。
- 《FreeSWITCH：VoIP实战》是《FreeSWITCH权威指南》的前身，不再更新，但该书是很好的入门书且有其历史意义。
- 《FreeSWITCH WIRESHARK》是一本介绍如何使用Wireshark分析SIP/RTP数据包的书。
- 《FreeSWITCH源代码分析》主要讲解源代码。
- 《FreeSWITCH权威指南》是正式出版的纸质书和电子书，出版于2014年。
- 《FreeSWITCH案例大全》是一本多人贡献的电子书，收集了很多FreeSWITCH实用案例，免费在线阅读。
- 《FreeSWITCH参考手册》是一本多人贡献的电子书，收集了很多FreeSWITCH实用参考，免费在线阅读。
- 《Kamailio实战》是关于Kamailio Proxy Server的书，与FreeSWITCH一起学习事半功倍。

以上所有图书均可以在 <http://book.dujinfang.com> 查看最新信息及购买。FreeSWITCH VIP知识星球里面也有部分电子书。

## 知识星球

杜老师维护着两个知识星球，一个免费版，一个收费版。可以使用如下链接或通过微信扫描二维码加入。

- FreeSWITCH: <https://t.zsxq.com/RBi6Ee2>
- FreeSWITCH VIP: <https://t.zsxq.com/2zb6qBE>



图 6-2: 知识星球

THIS PAGE INTENTIONALLY LEFT BLANK.