

一份不太简短的 L^AT_EX 2_ε 介绍

或 67 分钟了解 L^AT_EX 2_ε

原版作者: Tobias Oetiker

Hubert Partl, Irene Hyna and Elisabeth Schlegl

原版版本: Version 5.05, July 18, 2015

中文翻译: 中文 T_EX 学会

中文版本: 版本 5.05, 二零一六年二月

版权所有 © 1995 – 2005 Tobias Oetiker 及贡献者。保留所有权利。

本文档是自由的；你可以在自由软件协会颁布的 GNU 通用公共许可证的条款下再次发布或修改本文档。许可证可以是第二版，或者任何（自行选择的）后继版本。

本文档基于使用的目的发布，但并不负责任何担保；亦没有用于商业目的或某一特定目的的任何暗示性的担保。更多的细节请查看 GNU 通用公共许可证。

你应该随本文档同时收到一份 GNU 通用公共许可证的拷贝；如果没有，请发信给自由软件协会，地址：675 Mass Ave, Cambridge, MA 02139, USA。

中文版致谢

5.05 中文版致谢

4.20 中文版致谢

中文 T_EX 学会启动的 lshort-zh-cn 修正计划！本项计划历史八个月，参加的朋友有：

CT _E X 论坛 ID	翻译章节
zpxing	前言、第二章、第五章 1-2.4 & 3、第六章
Frogge	第一章
liwenjun	第三章
lijian605	第四章
gprsnl	第五章 2.5-2.11

haginile 和 Frogge 通读了全篇，并给出了详细的勘误表。blackold 对于第二章亦有所贡献。最后由 zpxing 统筹全书。

.....

3.20 中文版致谢

本文档的翻译工作由 CT_EX 版主“经典问题”倡议，历经近十个月才得以完成。期间参与翻译工作的朋友有：

CT _E X 论坛 ID	翻译章节	源文件名
经典问题	前言	overview.tex
高原之狼	第一章	things.tex
controlong	第二章	typeset.tex
cxtterm	第三章	math.tex, lssym.tex
aloft	第四章	spec.tex
ganzhi	第五章	custom.tex

在此特向这些奉献者表示感谢！

英文版致谢

Much of the material used in this introduction comes from an Austrian introduction to L^AT_EX 2.09 written in German by:

Hubert Partl <partl@mail.boku.ac.at>

Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna <Irene.Hyna@bmwf.ac.at>

Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl <noemail>

in Graz

If you are interested in the German document, you can find a version updated for L^AT_EX 2_ε by Jörg Knappen at CTAN://info/lshort/german

The following individuals helped with corrections, suggestions and material to improve this paper. They put in a big effort to help me get this document into its present shape. I would like to sincerely thank all of them. Naturally, all the mistakes you'll find in this book are mine. If you ever find a word that is spelled correctly, it must have been one of the people below dropping me a line.

Eric Abrahamsen, Lenimar Nunes de Andrade, Eilinger August, Rosemary Bailey, Barbara Beeton, Marc Bevand, Connor Blakey, Salvatore Bonaccorso, Pietro Braione, Friedemann Brauer, Markus Brühwiler, Jan Busa, David Carlisle, Neil Carter, Carl Cerecke, Mike Chapman, Pierre Chardaire, Xingyou Chen, Christopher Chin, Diego Clavadetscher, Wim van Dam, Benjamin Deschwanden Jan Dittberner, Michael John Downes, Matthias Dreier, David Dureisseix, Hans Ehrbar, Elliot, Rockrush Engch, William Faulk, Robin Fairbairns, Johan Falk, Jörg Fischer, Frank Fischli, Daniel Flipo, Frank, Mic Milic Frederickx, David Frey, Erik Frisk, Hans Fugal, Robert Funnell, Greg Gamble, Andy Goth, Cyril Goutte, Kasper B. Graversen, Arlo Griffiths, Alexandre Guimond, Neil Hammond, Christoph Hamburger, Rasmus Borup Hansen, Joseph Hilferty, Daniel Hirsbrunner, Martien Hulsen, Björn Hvittfeldt, Morten Høgholm, Werner Icking, Eric Jacoboni, Jakob, Alan Jeffrey, Martin Jenkins, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Nils Kanning, Andrzej Kawalec, Christian Kern, Alain Kessi, Axel Kielhorn, Sander de Kievit, Kjetil Kjernsmo, Tobias Klauser, Jörg Knappen, Michael Koundouros, Matt Kraai, Tobias Krewer, Flori Lambrechts, Mike Lee, Maik Lehardt, Rémi Letot, Axel Liljencrantz, Jasper Loy, Johan Lundberg, Martin Maechler, Alexander Mai, Claus Malten, Kevin Van Maren, Pablo Markin, I. J. Vera Marún, Hendrik Maryns, Chris McCormack, Aleksandar S. Milosevic, Henrik Mitsch, Stefan M. Moser, Philipp Nagele, Richard Nagy, Manuel Oetiker, Urs Oswald, Hubert Partl, Marcelo Pasin, Martin Pfister, Lan Thuy Pham, Breno Pietracci, Demerson Andre Polli, Maksym Polyakov, Nikos Pothitos, John Reffing, Mike Ressler, Brian Ripley, Kurt Rosenfeld, Bernd Rosenlecher, Chris Rowley, Young U. Ryu, Risto Saarelma, András Salamon, José Carlos Santos, Christopher Sawtell, Gilles Schintgen, Craig Schlenter, Hanspeter Schmid, Baron Schwartz, Jordi Serra i Solanich, Miles Spielberg, Susan Stewart, Matthieu Stigler, Geoffrey Swindale, Laszlo Szathmary, Boris Tobotras, Josef Tkadlec, Scott Veirs, Didier Verna, Carl-Gustav Werner, Fabian Wernli, Matthew Widmann, David Woodhouse, Chris York, Rick Zacccone, Fritz Zaucker, and Mikhail Zotov.

前言

L^AT_EX [1] 是一个文档准备系统 (Document Preparing System), 它非常适用于生成高印刷质量的科技类和数学类文档。它也能够生成所有其他种类的文档, 小到简单的信件, 大到完整的书籍。L^AT_EX 使用 T_EX [2] 作为它的排版引擎。

这份短小的手册描述了 L^AT_EX 2_ε 的使用, 对 L^AT_EX 的大多数应用来说应该是足够了。参考文献 [1, 3] 对 L^AT_EX 系统提供了完整的描述。

本手册共有八章和两篇附录:

第一章 讲述 L^AT_EX 的来源, 源代码的基本结构, 以及如何编译源代码生成文档。

第二章 讲述在 L^AT_EX 中如何书写文字, 包括中文。

第三章 讲述文档排版的基本元素——标题、目录、列表、图片、表格等等。结合前一张的内容, 你应当能够制作内容较为丰富的文档了。

第四章 L^AT_EX 排版公式的能力是众人皆知的。本章的内容涉及了一些常见的公式形式和符号。章节末尾提供了 L^AT_EX 常见的数学符号。

第五章 介绍了如何修改文档的一些基本样式, 包括字体、段落、页面尺寸、页眉页脚等。

第六章 介绍了 L^AT_EX 的一些扩展功能: 排版参考文献、排版索引、排版带有颜色和超链接的电子文档, 甚至排版幻灯片。

第七章 介绍了 L^AT_EX 的画图功能。作为入门手册, 这一部分点到为止。

第八章 当你相当熟悉前面几章的内容, 需要自己编写命令和宏包扩展 L^AT_EX 的功能时, 本章介绍了一些基本的命令满足你的需求。

附录 A 介绍了如何安装 T_EX 发行版和更新宏包。

附录 B 当新手遇到错误和需要寻求更多帮助时, 本章是一个基本的引导。

这些章节是循序渐进的, 建议刚刚熟悉 L^AT_EX 的读者按顺序阅读。一定要认真阅读例子, 它们贯穿全篇手册, 包含了很多的信息。

如果你已经对 L^AT_EX 较为熟练, 本手册的资源已不足够解决你的问题时, 请访问 “Comprehensive T_EX Archive Network” (CTAN) 站点, 主页是 www.ctan.org。所有的宏包也可以从 mirrors.ctan.org 和遍布全球的各个镜像站点中获得。

在本书中你会找到其他引用 CTAN 的地方, 形式为 CTAN:// 和之后的树状结构。引用本身是一个超链接, 点击后将打开内容在 CTAN 上相应位置的页面。

要在自己的电脑上安装 T_EX 发行版, 请参考附录 A 中的内容。各个操作系统下的 T_EX 发行版位于 CTAN://systems。

如果你有意在这份文档中增加、删除或者改变一些内容，请通知作者。作者对 L^AT_EX 初学者的反馈特别感兴趣，尤其是关于这份介绍哪些内容很容易理解，哪些内容可能需要更好地解释，而哪些内容由于太过难以理解、非常不常用而不适宜放在本手册。

鲁尚文 <louisstuart96@gmail.com>

中国科学院国家空间科学中心

lshort 的的最新中文版本位于：

[CTAN://info/lshort/chinese](http://info/lshort/chinese)

如果用户对其他语言的版本感兴趣，请浏览 [CTAN://info/lshort](http://info/lshort)。

目录

中文版致谢	i
英文版致谢	iii
前言	v
第一章 L ^A T _E X 须知	1
1.1 概述	1
1.1.1 T _E X	1
1.1.2 L ^A T _E X	1
1.1.3 L ^A T _E X 的优缺点	1
1.2 L ^A T _E X 命令和代码架构	2
1.2.1 L ^A T _E X 命令和环境	2
1.2.2 L ^A T _E X 源代码架构	3
1.3 用命令行操作 L ^A T _E X	3
1.3.1 引擎、格式和命令	4
1.3.2 latex 命令	4
1.3.3 pdflatex 和 xelatex 命令	5
1.4 宏包和文档类	5
1.4.1 文档类	5
1.4.2 宏包	6
1.5 L ^A T _E X 用到的文件一览	6
1.6 大文档的组织	7
第二章 用 L ^A T _E X 排版文字	9
2.1 语言文字和编码	9
2.1.1 ASCII 编码	9
2.1.2 扩展编码	9
2.1.3 UTF-8 编码	9
2.2 排版中文	10
2.2.1 xeCJK 宏包	10
2.2.2 ctex 宏包和文档类	10
2.3 L ^A T _E X 中的符号	11
2.3.1 空格和分段	11
2.3.2 注释	11
2.3.3 特殊字符	11
2.4 段落	11

2.5	单词之间的空格	12
2.6	断行和断页	12
2.6.1	手动断行和断页	12
2.6.2	连字符	12
2.7	预定义的字符串	12
第三章	文档元素	13
3.1	章节标题和目录	13
3.2	标题页	13
3.3	交叉引用	13
3.4	脚注	13
3.5	文字强调	13
3.6	特殊环境	13
3.6.1	列表	13
3.6.2	对齐环境	13
3.6.3	引用环境	13
3.6.4	摘要环境	14
3.6.5	代码环境	14
3.7	表格	14
3.8	图片	14
3.9	浮动体	14
3.10	盒子	14
3.11	画线功能	14
第四章	排版数学公式	15
4.1	\LaTeX 宏集	15
4.2	公式排版基础	15
4.2.1	数学模式浅谈	17
4.3	公式元素	17
4.3.1	一般符号	17
4.3.2	指数、上下标和导数	18
4.3.3	分式和根式	18
4.3.4	关系符	19
4.3.5	算符	19
4.3.6	巨算符	20
4.3.7	数学重音和上下括号	20
4.3.8	箭头	21
4.3.9	括号和定界符	21
4.4	多行公式	22
4.4.1	长公式折行	22
4.4.2	多行公式	22
4.4.3	公用编号的多行公式	23
4.5	数组和矩阵	23
4.6	公式中的空格	25
4.7	数学符号的字体控制	26

4.7.1	数学字母字体	26
4.7.2	数学符号的尺寸	26
4.7.3	加粗的数学符号	27
4.8	定理环境	27
4.8.1	<code>amsthm</code> 宏包	28
4.8.2	证明环境和证毕符号	29
4.9	符号表	31
4.9.1	L ^A T _E X 普通符号	31
4.9.2	\mathcal{AMS} 符号	35
第五章	排版样式设定	39
5.1	字体和字号	39
5.1.1	字体样式	39
5.1.2	字号	39
5.1.3	选用字体宏包	41
5.1.4	X _g L ^A T _E X 下使用 <code>fontspec</code> 宏包更改字体	41
5.1.5	使用 <code>xeCJK</code> 宏包更改中文字体	41
5.2	段落格式和间距	43
5.2.1	设置和修改长度	43
5.2.2	行距	43
5.2.3	段落格式	44
5.2.4	水平间距	44
5.2.5	垂直间距	45
5.3	页面和分栏	45
5.3.1	利用 <code>geometry</code> 宏包设置页面参数	45
5.3.2	页面内容的垂直对齐	47
5.3.3	分栏	47
5.4	页眉页脚	47
5.4.1	基本的页眉页脚样式	47
5.4.2	手动更改页眉页脚的内容	48
5.4.3	<code>fancyhdr</code> 宏包	48
第六章	特色工具和功能	51
6.1	参考文献和 Bib _T E _X 工具	51
6.1.1	基本的参考文献和引用	51
6.1.2	Bib _T E _X 数据库	51
6.1.3	使用 Bib _T E _X 排版参考文献	52
6.1.4	<code>natbib</code> 宏包	54
6.2	索引和 <code>makeindex</code> 工具	54
6.2.1	使用 <code>makeindex</code> 工具的方法	55
6.2.2	索引项的写法	55
6.3	使用颜色	56
6.3.1	颜色的表达方式	56
6.3.2	带颜色的文本和盒子	57
6.4	使用超链接	57

6.4.1	hyperref 宏包	57
6.4.2	超链接	58
6.4.3	PDF 书签	58
6.4.4	PDF 文档属性	58
6.5	beamer 幻灯片宏包	58
第七章	绘图功能	59
7.1	概述	59
7.2	picture 环境	59
7.3	交换图	59
7.4	TikZ 宏包	59
第八章	自定义 L^AT_EX 命令和功能	61
8.1	自定义命令和环境	61
8.1.1	定义新命令	61
8.1.2	定义环境	62
8.2	编写自己的宏包和文档类	63
8.2.1	编写简单的宏包	63
8.2.2	在宏包中调用其它宏包	63
8.2.3	编写自己的文档类	63
8.3	计数器	64
附录 A	安装 T_EX 发行版	65
A.1	T _E X 发行版简介	65
A.2	安装和更新宏包	65
A.3	L ^A T _E X 编辑器简介	65
附录 B	排除错误、寻求帮助	67
B.1	常见 L ^A T _E X 错误及原因	67
B.2	查看帮助文档	67
B.3	常见宏包简介	67
参考文献		69
索引		71

源代码示例列表

1.1	L ^A T _E X 的一个最简单的源代码示例。	3
5.1	fancyhdr 宏包的使用方法示例。	49
6.1	B _I B _T E _X 数据库示例 <code>books.bib</code> 。	53
6.2	利用 <code>books.bib</code> 生成参考文献的源代码 <code>demo.tex</code> 。	53
8.1	宏包的一个最简示例。	63

第一章 L^AT_EX 须知

欢迎加入 L^AT_EX 使用者的队伍！本章开头用简短的篇幅介绍了 L^AT_EX 的来源，然后介绍了 L^AT_EX 源代码的写法，如何编译 L^AT_EX 源代码生成文档，以及理解接下来的章节所必须的一些知识。

1.1 概述

1.1.1 T_EX

T_EX 是高德纳 (Donald E. Knuth) 开发的、以排版文字和数学公式为目的的一个计算机软件 [2]。高德纳从 1977 年开始开发 T_EX，以发掘当时开始进入出版工业的数字印刷设备的潜力。正在编写著作《计算机程序设计艺术》的高德纳，意图扭转排版质量每况愈下的状况，以免影响他的出书。我们现在使用的 T_EX 排版引擎发布于 1982 年，在 1989 年又稍加改进以更好地支持 8-bit 字符和多语言排版。T_EX 以其卓越的稳定性、跨平台、几乎没有 Bug 而著称。T_EX 的版本号不断趋近于 π ，当前为 3.141592653。

T_EX 读作“Tech”，其中“ch”的发音类似于“h”，与汉字“泰赫”的发音类似。T_EX 的拼写来自希腊词语 τεχνική (technique, 技术) 的开头几个字母。在 ASCII 字符环境，T_EX 写作 TeX。

1.1.2 L^AT_EX

L^AT_EX 为 T_EX 基础上的一套格式，令作者能够使用预定义的专业格式以较高质量排版和印刷他们的作品。L^AT_EX 的最初开发者为 Leslie Lamport 博士 [1]。L^AT_EX 使用 T_EX 程序作为自己的排版引擎。当下 L^AT_EX 主要的维护者为 Frank Mittelbach。

L^AT_EX 读作“Lah-tech”或者“Lay-tech”，近似于汉字“拉泰赫”或“雷泰赫”。L^AT_EX 在 ASCII 字符环境写作 LaTeX。当前的 L^AT_EX 版本为 L^AT_EX 2 _{ϵ} ，意思是超出了第二版，接近但没达到第三版，在 ASCII 字符环境写作 LaTeX2 _{ϵ} 。

1.1.3 L^AT_EX 的优缺点

L^AT_EX 总会拿来和一些“所见即所得” (What You See Is What You Get) 的文字处理和排版工具比较优缺点。笔者认为这种比较并不值得提倡，毕竟所有工具都有自己值得使用的原因。

当然笔者还是会总结一些 L^AT_EX 的优点：

- 专业的排版输出，产生的文档看上去就像“印刷品”一样。
- 方便而强大的数学公式排版能力，无出其右。
- 绝大多数时候，用户只需掌握一些简单易懂的命令来组织文档结构，无需（或很少）操心文档的版面设计。

- 复杂的专业排版元素，如脚注、交叉引用、参考文献、目录等很容易生成。
- 强大的扩展性，世界各地的人开发了数以千计的 L^AT_EX 宏包用于补充和扩展 L^AT_EX 的功能。本手册附录中的 B.3 小节可见一瞥。更多的宏包参考 *The L^AT_EX Companion* [3]。
- L^AT_EX 促使用户写出结构良好的文档——而这也是 L^AT_EX 存在的初衷。
- L^AT_EX 依赖的 T_EX 排版引擎和其它软件是跨平台、免费、开源的。无论用户使用的是 Windows, OS X, GNU/Linux 还是 FreeBSD 系统，都能轻松获得并使用这一强大的排版工具。

L^AT_EX 的缺点也是显而易见的：

- 入门门槛高。本手册的副标题叫做“67 分钟了解 L^AT_EX 2_ε”，实际上 67 是本手册正文部分的页数。如果你以平均一页一分钟的速度看完了本手册，你只是粗窥门径而已，离学会它还很远。
- 排查错误困难。作为一个靠写代码工作的排版工具，L^AT_EX 使用的宏语言比 C++ 或 Python 等编程语言的调试难度多得多。它虽然能够提示错误，但不提供 Debug 的机制，有时错误提示还很难理解。
- 样式定制困难。L^AT_EX 提供了一个基本上良好的样式，为了让用户不去关注样式而专注于文档结构。但如果想要改进 L^AT_EX 文档的样式则是十分困难。
- 相比“所见即所得”的模式有一些不便，为了查看生成的文档，用户总要不停地编译。

1.2 L^AT_EX 命令和代码架构

L^AT_EX 的源代码本质上就是个文本文件。哪怕用 Windows 的记事本或者 Linux 的 gedit 等简单的编辑器，你也可以编写一份 L^AT_EX 源代码并编译出文档来。专用于编辑 L^AT_EX 源代码的编辑器如 TeXworks / TeXstudio / WinEdt 等提供了一些语法高亮、命令补全等功能，以及调用排版引擎的一些按钮。

1.2.1 L^AT_EX 命令和环境

L^AT_EX 的命令贯穿于代码之中。命令以反斜线 \ 开头，为以下两种形式之一：

- 反斜线和后面的一串字母，如 \LaTeX。它们以任意非字母符号（空格、数字、标点等）作为分隔符。
- 反斜线和后面的一个非字母符号，如 \\$。它们无需分隔符。

要注意 L^AT_EX 命令是**对大小写敏感的**，比如输入 \LaTeX 命令可以生成错落有致的 L^AT_EX 字母组合，但输入 \Latex 或者 \LaTex 什么都得不到，还会报错。

L^AT_EX 命令（字母形式的）忽略其后的所有空格。如果要人为引入空格，需要在命令后面加一对括号阻止其忽略空格¹：

¹另外也可以在命令后面紧跟一个 _ 命令（反斜线加空格），代表插入一个间距。比如 \TeX_user 的输出效果就是 T_EX user。


```
Shall we call ourselves
\TeX users

or \TeX{} users?
```

```
Shall we call ourselves TEXusers
or TEX users?
```

大多数的 L^AT_EX 命令是带一个或多个参数，每个参数用花括号 { 和 } 包裹。有些命令带一个或多个可选参数，以方括号 [和] 包裹。还有些命令在命令名称后可以带一个星号 *，带星号和不带星号的命令效果有一定差异。

L^AT_EX 还引入了**环境**的用法，用以令一些效果在局部生效，或是生成特定的文档元素。L^AT_EX 环境的用法为一对命令 `\begin \end`：

```
\begin{<environment name>}
...
\end{<environment name>}
```

其中 *<environment name>* 为环境名，`\begin` 和 `\end` 中填写的环境名应当一致。`\begin` 在 *<environment name>* 后可以带一个或多个参数，甚至可选参数。环境允许嵌套使用。

1.2.2 L^AT_EX 源代码架构

L^AT_EX 源代码以一个 `\documentclass` 命令作为开头，它规定了文档使用的**文档类**：

```
\documentclass{...}
```

紧接着我们可以用 `\usepackage` 命令调用**宏包**：

```
\usepackage{...}
```

再接着，我们需要用以下一对命令来标记正文内容的开始位置和结束位置，而将正文内容写入其中：

```
\begin{document}
\end{document}
```

在 `\documentclass` 和 `\begin{document}` 之间的位置称为**导言区**，除了使用 `\usepackage` 调用宏包之外，一些对文档的全局设置命令也在这里使用。当然你也可以什么都不写，一个宏包都不调用。一切视自己需求。

1.3 用命令行操作 L^AT_EX

相信你到这里已经急不可耐地想要写一个 L^AT_EX 源代码试一试了。我们这就给一个最小的例子，见源代码 1.1。

```
\documentclass{article}
\begin{document}
‘‘Hello world!’’ from \LaTeX.
\end{document}
```

源代码 1.1: L^AT_EX 的一个最简单的源代码示例。

有相当多的编辑器会提供一个按钮完成编译，不过笔者仍然觉得有必要了解一下其背后的工作原理。L^AT_EX 调用的程序都是基于命令行的，所以建议打开 Windows 命令提示符或者 Linux / OS X 的终端，按照本手册的范例尝试一下调用命令行程序编译。

1.3.1 引擎、格式和命令

在开始讲解怎么编译之前，有必要澄清几个概念：

引擎 全称为排版引擎，是读入源代码并编译生成文档的可执行程序，如 pdf_TE_X、X_YL_AT_EX 等。有时也直接称为编译器。

格式 是定义了一组命令的代码集。L^AT_EX 就是一个格式，高德纳本人还编写了一个简单的 plain_TE_X 格式，没有定义诸如 `\documentclass` 和 `\section` 等等命令。

命令 是引擎和格式二者的结合体。如下文要用到的 `pdflatex` 命令实际上是 pdf_TE_X 引擎结合 L^AT_EX 格式的一个可执行命令，用以将像源代码 1.1 这样的代码编译输出 PDF。有时也把命令写作 pdfL^AT_EX 和 X_YL^AT_EX。

`latex` 命令和 L^AT_EX 格式两个名词经常会混用，在同他人讨论关于 L^AT_EX 的时候需要明确。本手册为避免混淆，文中的 L^AT_EX 一律指的是格式，而命令则用等宽字体 `latex` 表示。

用一个简单的表格总结一下：

	plain T _E X 格式	L ^A T _E X 格式
T _E X 引擎	<code>tex</code>	N/A
pdf _T E _X 引擎	<code>etex</code>	<code>latex</code> ²
	<code>pdftex</code>	<code>pdflatex</code>
X _Y L _A T _E X 引擎	<code>xetex</code>	<code>xelatex</code>

1.3.2 latex 命令

假使你的计算机上已经安装了 L^AT_EX 依赖的程序和工具（安装方法见附录 A）。我们将源代码 1.1 拷贝到一个文本文件中，保存为 `helloworld.tex`。然后在命令行输入：

```
latex helloworld.tex
```

（也可以输入不带扩展名的 `latex helloworld`）。

此时命令行界面会闪过许多信息。如果一切正常，再行检查目录，会发现生成了 `helloworld-.dvi` 以及其它文件。这个扩展名为 DVI 的文件就是编译输出的文档。

Linux 下可以使用命令行程序 `xdvi` 打开这个文件：

```
xdvi helloworld.dvi
```

Windows 下大多预装了 `yap` 软件（Mik_TE_X）或 `dviout` 软件（T_EX Live），我们可以直接双击 `helloworld.dvi` 打开它。

要进一步生成现今流行的 PDF 文档格式，我们还需要用额外的命令将 DVI 转为 PDF：

```
dvipdfmx helloworld.dvi
```

然后就可以用查看 PDF 的软件（Adobe Reader / Foxit Reader 等）打开生成的 `helloworld.pdf` 查看了。

²现今的发行版中 `latex` 命令也用 pdf_TE_X 引擎编译 L^AT_EX 格式的代码，生成的文档是 DVI。

1.3.3 pdf_latex 和 xelatex 命令

这两个命令相比于 latex 命令更为方便，我们可以直接生成 PDF：

```
pdflatex helloworld.tex
```

或者

```
xelatex helloworld.tex
```

xelatex 命令，也就是我们俗称的 X_gLaTeX，有着各种新的特性，如能够直接支持使用系统预装的字体、原生支持 UTF-8 编码等。尤其是排版中文，X_gLaTeX 配合适当的宏包是现在最新、最方便的方式。

1.4 宏包和文档类

我们在 1.2.2 小节描述的 LaTeX 源代码架构中已经见识了宏包和文档类。本节将仔细解释它们。

1.4.1 文档类

文档类规定了 LaTeX 源代码所要生成的文档的性质——普通文章，书籍，等等。LaTeX 源代码的开头须用 \documentclass 指定文档类：

```
\documentclass[<options>]{<class-name>}
```

其中 *<class-name>* 为文档类的名称，如 LaTeX 提供的 article, book, report，在其基础上派生的一些文档类如支持中文排版的 ctexart / ctexbook / ctexrep，或者有其它功能的一些文档类。LaTeX 提供的基础文档类见表 1.1。

表 1.1: LaTeX 提供的基础文档类

article 文章格式的文档类，广泛用于科技论文、报告、说明文档等。

proc 基于 article 文档类的一个简单的学术文档模板。

minimal 一个极其精简的文档类，只设定了纸张大小和字号，用作代码测试的最小工作实例 (Minimal Working Example)。

report 长篇报告格式的文档类，具有章节结构，用于综述、长篇论文、简单的书籍等。

book 书籍文档类，包含前言、正文、后记等结构。

slides 幻灯格式的文档类，使用无衬线字体。

可选参数 *<options>* 为文档类设置选项，以全局地影响文档布局的参数，如字号、纸张大小、单双面等等。LaTeX 的三个基础文档类使用的参数见表 1.2。

比如如下开头的文档将排版为文章，纸张为 A4 大小，基本字号为 11pt：

```
\documentclass[11pt,twoside,a4paper]{article}
```

1.4.2 宏包

在编写 L^AT_EX 源代码时，你时常会发现 L^AT_EX 的基础功能不能满足你的需求，比如排版复杂的表格、插入图片、增加颜色甚至超链接等等。这时你需要依赖一些扩展来增强或补充 L^AT_EX 的功能。这些扩展称为**宏包**。调用宏包的方法非常类似使用文档类的方法：

```
\usepackage[⟨options⟩]{⟨package-name⟩}
```

附录中的 [B.3](#) 汇总了常用的一些宏包。我们在手册接下来的章节中，也会穿插介绍一些最常用的宏包的使用方法。

在使用宏包和文档类之前，一定要首先确认它们是否安装在你的计算机中，否则 `\usepackage` 等命令会报错误。详见附录 [A.2](#)。

每个宏包（包括前面所说的文档类）都定义了许多命令和环境，或者修改了 L^AT_EX 已有的命令和环境。为了明白它们的用法，用户需要查阅宏包和文档类的帮助手册。使用方法是在 Windows 命令提示符或者 Linux 终端下输入命令：

```
texdoc ⟨pkg-name⟩
```

其中 `⟨pkg-name⟩` 是宏包的名称，你也可以输入文档类的名称 `⟨cls-name⟩`。更多获得帮助的方法见附录 [B.2](#)。

1.5 L^AT_EX 用到的文件一览

除了我们需要编写的源代码 `.tex` 文件，我们还可能接触到形形色色的文件。本节简单介绍一下这些文件。

每个宏包和文档类都是带特定扩展名的文件：

表 1.2: L^AT_EX 标准的三个文档类可设定的选项

<code>10pt, 11pt, 12pt</code>	设定文档的基本字号。缺省为 10pt。
<code>a4paper, letterpaper, ...</code>	设定纸张大小，默认为美式纸张 <code>letterpaper</code> 。可设置选项还包括 <code>a5paper, b5paper, executivepaper</code> 和 <code>legalpaper</code> 。
<code>fleqn</code>	令行间公式左对齐（缺省为居中）。
<code>leqno</code>	将公式编号放在左边（缺省为右边）。
<code>titlepage, notitlepage</code>	设定标题命令 <code>\maketitle</code> 是否单独成页。 <code>article</code> 缺省为 <code>notitlepage</code> ， <code>report</code> 和 <code>book</code> 缺省为 <code>titlepage</code> 。
<code>onecolumn, twocolumn</code>	设定单栏/双栏排版。
<code>twoside, oneside</code>	设定是否为单面/双面排版。双面排版时，奇偶页的页眉页脚、页边距不同。 <code>article</code> 和 <code>report</code> 缺省为单面排版， <code>book</code> 缺省为双面。
<code>landscape</code>	设定横向排版。缺省为纵向。
<code>openright, openany</code>	设定新的一章 <code>\chapter</code> 是在奇数页（右侧）开头，还是直接紧跟着上一张开头。 <code>report</code> 缺省为 <code>openany</code> ， <code>book</code> 缺省为 <code>openany</code> 。【 <code>article</code> 没有章一级文档结构】

`.sty` 宏包文件。宏包的名称就是去掉扩展名的文件名。

`.cls` 文档类文件。同样地，文档类名称就是文件名。

`.bib` B_BT_EX 参考文献数据库文件。

`.bst` B_BT_EX 用到的参考文献格式模板。详见 6.1.3。

`.ist` makeindex 用到的索引格式模板。

L^AT_EX 在编译过程中生成相当多的辅助文件和日志。一些功能如交叉引用、参考文献、目录、索引等，需要先编译生成辅助文件，然后再次编译时读入辅助文件得到正确的结果，所以复杂的 L^AT_EX 源代码可能要编译多次：

`.log` 排版引擎生成的日志文件，供排查错误使用。

`.aux` L^AT_EX 生成的主辅助文件，记录交叉引用、目录、参考文献的引用等。

`.toc` L^AT_EX 生成的目录记录文件。

`.lof` L^AT_EX 生成的图片目录记录文件。

`.lot` L^AT_EX 生成的表格目录记录文件。

`.bbl` B_BT_EX 生成的参考文献记录文件。

`.blg` B_BT_EX 生成的日志文件。

`.idx` L^AT_EX 生成的供 makeindex 处理的索引记录文件。

`.ind` makeindex 处理 `.idx` 生成的格式化索引记录文件。

`.ilg` makeindex 生成的日志文件。

`.out` hyperref 宏包生成的 PDF 书签记录文件。

1.6 大文档的组织

当编写较大规模的 L^AT_EX 源代码，如书籍、毕业论文等，你有理由将源代码分成若干个文件而不是写到一堆，比如很自然地每章写一个文件。

L^AT_EX 提供了命令 `\include` 用来在源代码里插入文件：

```
\include{<filename>}
```

`<filename>` 为文件名，如果和要编译的主文件不在一个目录中，则要加上相对路径。`<filename>` 可以不带扩展名，此时默认为 `.tex`；扩展名不是 `.tex` 的文件必须带扩展名。

值得注意的是 `\include` 在读入 `<filename>` 之前会另起一页。有的时候我们并不需要这样，而是用 `\input` 命令，它纯粹是把文件里的内容插入：

```
\input{<filename>}
```

另外 L^AT_EX 提供了一个 `\includeonly` 命令来组织文件，用于**导言区**，指定只载入某些文件：

```
\includeonly{<filename1>,<filename2>,...}
```

导言区使用了 `\includeonly` 后，正文中不在其列表范围的 `\include` 命令不会起效。

最后介绍一个实用的工具宏包 `syntonly`。加载这个宏包后，在导言区使用 `\syntonly` 命令，可令 L^AT_EX 编译后不生成 DVI 或者 PDF 文档，只排查错误，编译速度会快不少：

```
\usepackage{syntonly}  
\syntonly
```

如果想生成文档，则将 `\syntonly` 命令那一行用 `%` 注释掉即可。

第二章 用 L^AT_EX 排版文字

2.1 语言文字和编码

L^AT_EX 源代码为文本文件，而文本文件的一个至关重要的性质是它的编码。在此用尽量短的篇幅介绍一下。

2.1.1 ASCII 编码

计算机的基本存储单位是字节 (byte)，每个字节为八位 (8-bit)，能够代表 0-255 这 256 个数。ASCII (美国通用信息交换码) 覆盖前 128 个数 0-127，也就是 7-bit，能够表示 QWERTY 键盘上能够输入的拉丁字母、数字和符号。

由于 T_EX 最初面向英语文档的排版，ASCII 编码完全够用，而 T_EX 最早也只支持 7-bit 和 ASCII。如果要排版西欧语系中带重音的字符，就要用到后文所叙述的重音命令了，如 Möbius 必须输入 M\ "obius。

2.1.2 扩展编码

在 ASCII 之后，各种语言都发展出来了自己的编码，比如西欧语言的 Latin-1，日本的 Shift-JIS，中国大陆的 GB2312-80 和 GBK 等。它们之中的绝大多数都向下兼容 ASCII，因此无论是在哪种编码下，T_EX 以及 L^AT_EX 的命令和符号都能用。

西欧 (拉丁字母)、俄语系 (西里尔字母) 等的编码较容易处理，因为它们刚好能够利用 128-255 这个编码范围。latex 命令及 pdfL^AT_EX 命令下，这些编码的支持由 inputenc 宏包提供。比如将文档保存为 Latin-1 编码，并在导言区使用：

```
\usepackage[latin1]{inputenc}
```

接下来，Möbius 就直接可以通过 (用适当输入法) 输入 Möbius 得到了。其它一些语言也可以使用 inputenc 宏包配合 babel 宏包排版。

GBK 等编码是多字节编码，ASCII 字符为一个字节，而非 ASCII 字符为两个字节，这就需要借助一些宏包进行较为复杂的判断和处理。CJK 宏包就是处理汉语等多字节编码的语言使用的宏包。但 CJK 宏包的使用非常不方便，笔者不再推荐直接使用。后一节将会简单介绍现在推荐的中文支持方案。

2.1.3 UTF-8 编码

Unicode 是一个多国语言文字的集合，覆盖了几乎全球范围内的语言文字。UTF-8 是 Unicode 的一套编码方案，一个字符可以由一个到四个 (现在用到三个) 字节编码，其中单字节范围仍然是 ASCII 字符。

latex 命令及 pdfL^AT_EX 命令下可以使用 inputenc 宏包支持 UTF-8：

```
\usepackage[utf8]{inputenc}
```

X_YL^AT_EX 原生支持 UTF-8 编码，而且也不适用 `inputenc` 宏包。将源代码以 UTF-8 格式保存，并用 `fontspec` 宏包（见 5.1.4 小节）载入所需的字体，就可以在 `.tex` 源代码中输入任意语言的文字。但各个语言的特殊排版要求需要更多的宏包支持（印地语、阿拉伯语等等），如 `babel`、`polyglossia` 等。

2.2 排版中文

2.2.1 xeCJK 宏包

用 L^AT_EX 排版中文的一大门槛是中文字体。T_EX 使用的字体格式仅支持不超过 256 个字符，要想排版中文，比如旧式的 CJK 宏包，往往需要复杂的预处理，将中文字体拆分成数百个小字体，非常麻烦¹。

X_YL^AT_EX 支持直接使用系统安装的 TTF/OTF 等格式的字体，加上对 UTF-8 编码的原生支持，免去了预处理字体的麻烦。在此基础上的 `xeCJK` 宏包更进一步完善了中英文混排、标点符号等排版细节，比如中英文之间插入空隙、中文行尾的回车不引入空格、标点符号不出现在行首，等等。

`xeCJK` 宏包支持用简单的命令配置中文字体。举一个在 Windows 下使用 `xeCJK` 的例子，源代码须保存为 UTF-8 编码（设置字体的命令详见 5.1.5 小节）：

```
\documentclass{article}
\usepackage{xeCJK}
\setCJKmainfont{SimSun}
\begin{document}
中文LaTeX排版。
\end{document}
```

2.2.2 ctex 宏包和文档类

`ctex` 宏包和文档类是在 CJK 和 `xeCJK` 基础上的进一步封装。`ctex` 文档类包括 `ctexart` / `ctexrep` / `ctexbook`，是对 L^AT_EX 的三个标准文档类的封装，对 L^AT_EX 的排版样式做了许多调整，以切合中文排版风格。最新版本的 `ctex` 宏包/文档类甚至支持自动配置字体。比如上述例子可进一步简化为：

```
\documentclass{ctexart}
\begin{document}
中文LaTeX排版。
\end{document}
```

`ctex` 宏包/文档类支持源代码使用 UTF-8 和 GBK 编码（要在文档类指定 UTF8 或 GBK 选项，详见 `ctex` 宏包参考手册），用 `latex+dvipdfmx` 命令、`pdfLATEX` 或 `XYLATEX` 命令（不支持 GBK）都能够编译。推荐使用 UTF-8 编码编写源代码，用 `XYLATEX` 命令编译。

¹现在的 `pdfLATEX` 以及 `latex + dvipdfmx` 通过字体映射的方式支持直接使用中文字体，不需要经过预处理，但仍然是不方便新手操作的。

2.3 L^AT_EX 中的符号

2.3.1 空格和分段

L^AT_EX 源代码中，空格键和 Tab 键输入的空白字符视为“空格”。连续的若干个空白字符视为一个空格。另外一行开头的空格是忽略不计的。

行末的回车视为一个空格；但连续两个回车，也就是空行，会将文字分段。多个空行被视为一个空行。以下为一个示例，左边为源代码的写法，右边为输出的效果。

```
Several spaces      equal one.
  Front spaces are ignored.

An empty line starts a new
paragraph.
```

Several spaces equal one. Front spaces
are ignored.

An empty line starts a new paragraph.

2.3.2 注释

L^AT_EX 用 % 字符作为注释。在这个字符之后直到行末，所有的字符都被忽略，连同回车引入的空格。我们需要注意以下示例中“回车引入的空格被忽略”的效果：

```
This is an % short comment
% ---
% Long and organized
% comments
% ---
example: Comments do not bre%
ak a word.
```

This is an example: Comments do not
break a word.

2.3.3 特殊字符

以下字符在 L^AT_EX 里有特殊用途，如 % 表示注释，\$, ^、_ 等用于排版数学公式，& 用于排版表格，等等。直接输入这些字符得不到对应的符号，还往往会出错：

\$ % & { } _ ^ ~ \

如果想要输入以上符号，需要使用以下带反斜线的形式输入：

```
\# \$ \% \& \{ \} \_ \^ \~ \backslash
\^{} \~{}
\textbackslash
```

\$ % & { } _ ^ ~ \

事实上这些带反斜线的形式就是 L^AT_EX 命令。\\ 和 \~ 两个命令的形式比较特殊，如果不加一对括号，就和后面的字符形成了重音效果（详见某节）。\\ 被直接定义成了手动换行的命令，输入反斜杠就只好用 \textbackslash。

2.4 段落

2.5 单词之间的空格

2.6 断行和断页

2.6.1 手动断行和断页

2.6.2 连字符

2.7 预定义的字符串

第三章 文档元素

3.1 章节标题和目录

3.2 标题页

3.3 交叉引用

3.4 脚注

3.5 文字强调

3.6 特殊环境

3.6.1 列表

3.6.2 对齐环境

3.6.3 引用环境

3.6.4 摘要环境

3.6.5 代码环境

3.7 表格

3.8 图片

3.9 浮动体

3.10 盒子

3.11 画线功能

第四章 排版数学公式

准备好了！本章将见识到 \LaTeX 闻名的强项——排版数学公式。当然你得注意了，本章的内容只是一点皮毛，虽然对大多数人来说已经够用了，但是如果不能解决你的问题的话也不要大惊小怪，求助于搜索引擎或者有经验的人不失为一个好办法。

4.1 \mathcal{AMS} 宏集

在介绍数学公式排版之前，简单介绍一下 \mathcal{AMS} 宏集。 \mathcal{AMS} 宏集合是美国数学学会 (American Mathematical Society) 提供的对 \LaTeX 原生的数学公式排版的扩展，其核心是 `amsmath` 宏包，对多行公式的排版提供了有力的支持。此外，`amsfonts` 宏包以及基于它的 `amssymb` 宏包提供了丰富的数学符号；`amsthm` 宏包扩展了 \LaTeX 定理证明格式。

本章介绍的许多命令和环境依赖于 `amsmath` 宏包。这些命令和环境将以蓝色示意。

4.2 公式排版基础

数学公式有两种排版方式：其一是与文字混排，称为**行内公式**；其二是单独列为一行排版，称为**行间公式**。

行内公式由一对 `$` 符号包裹：

```
Add $a$ squared and $b$ squared  
to get $c$ squared. Or, using  
a more mathematical approach:  
$a^2 + b^2 = c^2$
```

```
Add  $a$  squared and  $b$  squared to get  $c$   
squared. Or, using a more mathematical  
approach:  $a^2 + b^2 = c^2$ 
```

单独成行的**行间公式**在 \LaTeX 里由 `equation` 环境包裹。`equation` 环境为公式自动生成一个编号，这个编号可以用 `\label` 和 `\ref` 生成交叉引用，`amsmath` 的 `\eqref` 命令甚至为引用自动加上括号；你还可以用 `\tag` 命令手动修改公式的编号，或者用 `\notag` 命令取消为公式编号。

```
Add $a$ squared and $b$ squared
to get $c$ squared
\begin{equation}
a^2 + b^2 = c^2
\end{equation}
Einstein says
\begin{equation}
E = mc^2 \label{clever}
\end{equation}
This is a reference to
\eqref{clever}.
```

Add a squared and b squared to get c squared

$$a^2 + b^2 = c^2 \quad (4.1)$$

Einstein says

$$E = mc^2 \quad (4.2)$$

This is a reference to (4.2).

```
It's wrong to say
\begin{equation}
1 + 1 = 3 \tag{dumb}
\end{equation}
or
\begin{equation}
1 + 1 = 4 \notag
\end{equation}
```

It's wrong to say

$$1 + 1 = 3 \quad (\text{dumb})$$

or

$$1 + 1 = 4$$

当然你不会愿意为每个公式都手动取消编号。L^AT_EX 提供了一对命令 `\[` 和 `\]` 用于生成不带编号的行间公式，与之等效的是 `displaymath` 环境。有的人更喜欢 `equation*` 环境，体现了带星号和不带星号的环境之间的区别。

```
Again\ldots
\begin{equation*}
a^2 + b^2 = c^2
\end{equation*}
or you can type less for the
same effect:
\[ a^2 + b^2 = c^2 \]
or totally the same:
\begin{displaymath}
a^2 + b^2 = c^2
\end{displaymath}
```

Again...

$$a^2 + b^2 = c^2$$

or you can type less for the same effect:

$$a^2 + b^2 = c^2$$

or totally the same:

$$a^2 + b^2 = c^2$$

我们通过一个例子展示行内公式和行间公式的对比。为了与文字相适应，行内公式在排版大的公式元素（分式、巨算符等）时显得很“局促”：

```

This is text style:
 $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2}$ 
 $= \frac{\pi^2}{6}$ .
And this is display style:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$


```

This is text style: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$. And this is display style:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

4.2.1 数学模式浅谈

当你使用 `$` 开启行内公式输入，或是使用 `\[` 命令、`equation` 环境时，你就进入了所谓的**数学模式**。数学模式相比于文本模式有以下特点：

1. 数学模式中输入的空格全部被忽略。数学符号的间隙默认完全由符号的性质（关系符号、运算符等）决定。需要人为引入空隙时，使用 `\quad` 和 `\qquad` 等命令。详见4.6节。
2. 不允许有空行（分段）。每个公式（每组多行公式）自成一个段落。
3. 所有的字母被当作数学公式中的变量处理，字母间距与文本模式不一致，也无法生成单词之间的空格。如果想在数学公式中输入正常字体的文本，`amsmath` 提供了一个方便的 `\text` 命令¹。

```

 $x^2 \geq 0 \quad \text{for all } x \in \mathbb{R}$ 

```

$x^2 \geq 0 \quad \text{for all } x \in \mathbb{R}$

4.3 公式元素

本节我们将接触到形形色色的数学符号，它们是 L^AT_EX 卓越的数学公式排版能力的基础。L^AT_EX 默认提供了常用的数学符号，`amssymb` 宏包提供了一些次常用的符号。这些符号全部能在本章末尾的 4.9 节列出的表格里查到。

4.3.1 一般符号

希腊字母符号的名称就是其英文名称，如 α (`\alpha`)、 β (`\beta`) 等等。大写的希腊字母为首字母大写的命令，如 Γ (`\Gamma`)、 Δ (`\Delta`) 等等。无穷大符号为 ∞ (`\infty`)。

省略号有 \dots (`\dots`) 和 \cdots (`\cdots`) 两种形式。它们有各自合适的用途：

```

 $a_1, a_2, \dots, a_n$ 
 $a_1 + a_2 + \cdots + a_n$ 

```

a_1, a_2, \dots, a_n
 $a_1 + a_2 + \cdots + a_n$

除此之外在矩阵中会用到竖排的 \vdots (`\vdots`) 和斜排的 \ddots (`\ddots`)。

¹`\text` 命令仅适合在公式中穿插少量文字。如果你的情况正好相反，需要在许多文字中穿插使用公式，则应该像正常的行内公式那样，而不是滥用 `\text` 命令。

4.3.4 关系符

L^AT_EX 常见的关系符号除了可以直接输入的 =, >, <, 其它符号用命令输入, 如不等号 ≠ (\ne)、大于等于号 ≥ (\ge)、小于等于号 ≤ (\le)、约等号 ≈ (\approx)、等价 ≡ (\equiv)、相似 ∼ (\sim)²等等。更多符号命令可参考表 4.6 以及表 4.18。

L^AT_EX 还提供了自定义二元关系符的命令 \stackrel{\dots}{\dots}, 用于将一个符号叠加在原有的二元关系符之上:

```
\[
f_n(x) \stackrel{*}{\approx} 1
\]
```

$$f_n(x) \overset{*}{\approx} 1$$

4.3.5 算符

L^AT_EX 中的算符大多数是二元算符, 除了直接用键盘可以输入的 +, −, *, /, 其它符号用命令输入, 如乘号 × (\times)、除号 ÷ (\div)、点乘 · (\cdot)、加减号 ± (\pm) 等等。更多符号命令可参考表 4.7 以及表 4.17。

∇ (\nabla) 和 ∂ (\partial) 也是常用的算符, 虽然它们不属于二元算符。

L^AT_EX 将函数名称整体作为一个算符排版, 函数名的字体是直立字体。其中有一部分符号在上下位置可以书写一些内容作为条件, 类似于后文所叙述的巨算符。

表 4.1: L^AT_EX 作为算符的函数名称一览。

不带上下限的算符				
\sin	\arcsin	\sinh	\exp	\dim
\cos	\arccos	\cosh	\log	\ker
\tan	\arctan	\tanh	\lg	\hom
\cot	\arg	\coth	\ln	\deg
\sec	\csc			
带上下限的算符				
\lim	\limsup	\liminf	\sup	\inf
\min	\max	\det	\Pr	\gcd

```
\[
\lim_{x \rightarrow 0}
\frac{\sin x}{x}=1
\]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

对于求模表达式, L^AT_EX 提供了 \pmod 和 \bmod 命令, 前者相当于一个二元运算符, 后者作为同余表达式的后缀:

```
$a\bmod b \\\
x\equiv a \pmod{b}$
```

$$a \bmod b$$

$$x \equiv a \pmod{b}$$

上表中的算符如果不满足使用的话, amsmath 允许用户用 \DeclareMathOperator 定义自己的算符, 其中带星号的命令定义带上下限的算符:

²经常用来替代 \sim 作为波浪号。但注意只能在数学模式使用。

```
\DeclareMathOperator{\argh}{argh}
\DeclareMathOperator*{\Nut}{Nut}
```

效果为:

$$\backslash\argh 3 = \backslashnut_{\{x=1\}} 4x\backslash$$

$$\argh 3 = \operatorname{Nut}_{x=1} 4x$$

4.3.6 巨算符

积分号 \int (`\int`)、求和号 \sum (`\sum`) 等符号称为**巨算符**。巨算符在行内公式和行间公式的大小和形状有区别。

In text:
`$\sum_{i=1}^n \quad`
`\int_0^{\frac{\pi}{2}} \quad`
`\prod_{\epsilon} \$ \quad`
 In display:
`\[\sum_{i=1}^n \quad`
`\int_0^{\frac{\pi}{2}} \quad`
`\prod_{\epsilon} \]`

In text: $\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$
 In display:

$$\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$$

巨算符的上下标用作其上下限。行间公式中，积分号默认将上下限放在右上角和右下角，求和号默认在上下方；行内公式一律默认在右上角和右下角。可以在巨算符后使用 `\limits` 手动令上下限显示在上下方，`\nolimits` 则相反。

In text:
`$\sum\limits_{i=1}^n \quad`
`\int\limits_0^{\frac{\pi}{2}} \quad`
`\prod\limits_{\epsilon} \$ \quad`
 In display:
`\[\sum\nolimits_{i=1}^n \quad`
`\int\limits_0^{\frac{\pi}{2}} \quad`
`\prod\nolimits_{\epsilon} \]`

In text: $\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$
 In display:

$$\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$$

`amsmath` 宏包还提供了 `\substack` 命令，能够在上下限位置书写多行表达式：

`\[`
`\sum^n_{\substack{0<i<n \\ j\subseteq i}}`
`P(i,j) = Q(i,j)`
`\]`

$$\sum_{\substack{0<i<n \\ j\subseteq i}}^n P(i,j) = Q(i,j)$$

4.3.7 数学重音和上下括号

数学符号可以像文字一样加重音，比如对时间求导的符号 \dot{r} (`\dot{r}`)、 \ddot{r} (`\ddot{r}`)、表示向量的箭头 \vec{r} (`\vec{r}`) 等，详见表 4.9。使用时要注意重音符号的作用区域，一般应当对某个符号而不是“符号加下标”使用重音：

```
\bar{x}_0 \quad \bar{x}_0 \\
\vec{x}_0 \quad \vec{x}_0
```

$$\bar{x}_0 \quad \bar{x}_0$$

$$\vec{x}_0 \quad \vec{x}_0$$

L^AT_EX 也能为多个字符加重音，包括直接画线的 `\overline` 和 `\underline` 命令（可叠加使用）、宽重音符号 `\widehat`、表示向量的箭头 `\overrightarrow` 等。后两者详见表 4.9 和 4.11 等。

```
$0.\overline{3} =
\underline{\underline{1/3}}$ \\
\hat{XY} \quad \widehat{XY} \\
\vec{AB} \quad \overrightarrow{AB}
```

$$0.\overline{3} = \underline{\underline{1/3}}$$

$$\hat{XY} \quad \widehat{XY}$$

$$\vec{AB} \quad \overrightarrow{AB}$$

`\overbrace` 和 `\underbrace` 命令用来生成上/下括号，各自可带一个上/下标公式。

```
\underbrace{\overbrace{a+b+c}^6}_{\text{meaning of life}} = 42
```

$$\underbrace{a+b+c}_6 \cdot \underbrace{d+e+f}_7 = 42$$

meaning of life

4.3.8 箭头

除了作为上下标之外，箭头还用于表示过程。amsmath 的 `\xleftarrow` 和 `\xrightarrow` 命令可以为箭头增加上下标：

```
\[ a\xleftarrow{x+y+z} b \]
\[ c\xrightarrow[x<y]{a*b*c} d \]
```

$$a \xleftarrow{x+y+z} b$$

$$c \xrightarrow[x<y]{a*b*c} d$$

4.3.9 括号和定界符

L^AT_EX 提供了多种括号和定界符表示公式块的边界。除小括号 ()、中括号 [] 之外，其余都是 L^AT_EX 命令，包括大括号 { }。表 4.12 和 4.13 给出了更多的括号/定界符命令。

```
{a,b,c} \neq \{a,b,c\}
```

$$a, b, c \neq \{a, b, c\}$$

使用 `\left` 和 `\right` 命令可令括号（定界符）的大小可变，在行间公式中常用。L^AT_EX 会自动根据括号内的公式大小决定定界符大小。`\left` 和 `\right` 必须成对使用。需要使用单个定界符时，另一个定界符写成 `\left.` 或 `\right.`。

```
\[ 1 + \left(\frac{1}{1-x^2}\right)^3 \quad
\left.\frac{\partial f}{\partial t}\right|_{t=0} \]
```

$$1 + \left(\frac{1}{1-x^2}\right)^3 \quad \left.\frac{\partial f}{\partial t}\right|_{t=0}$$

有时我们不满意于 L^AT_EX 为我们自动调节的定界符大小。这时我们还可以用 `\big`、`\bigg` 等命令生成固定大小的定界符。更常用的形式是类似 `\left` 的 `\bigl`、`\biggl` 等，以及类似 `\right` 的 `\bigr`、`\biggr` 等（`\bigl` 和 `\bigr` 不必成对出现）。

`align` 环境会给每行公式都编号。我们仍然可以用 `\notag` 去掉某行的编号。在以下的例子，我们还需调整加号的对齐位置：

```
\begin{align}
a &= b + c \\
&= d + e + f + g + h + i \\
&+ j + k + l \notag \\
&\quad + m + n + o \\
&= p + q + r + s
\end{align}
```

$$a = b + c \quad (4.6)$$

$$= d + e + f + g + h + i + j + k + l \\ + m + n + o \quad (4.7)$$

$$= p + q + r + s \quad (4.8)$$

`align` 还能够对齐多组公式，除等号前的 `&` 之外，公式之间也用 `&` 分隔：

```
\begin{align}
a &= 1 & b &= 2 & c &= 3 \\
d &= -1 & e &= -2 & f &= -5
\end{align}
```

$$a = 1 \quad b = 2 \quad c = 3 \quad (4.9)$$

$$d = -1 \quad e = -2 \quad f = -5 \quad (4.10)$$

如果我们不需要按等号对齐，只需罗列数个公式，`gather` 将是一个很好用的环境：

```
\begin{gather}
a = b + c \\
d = e + f + g \\
h + i = j + k \notag \\
l + m = n
\end{gather}
```

$$a = b + c \quad (4.11)$$

$$d = e + f + g \quad (4.12)$$

$$h + i = j + k$$

$$l + m = n \quad (4.13)$$

`align` 和 `gather` 有对应的不带编号的版本 `align*` 和 `gather*`。

4.4.3 公用编号的多行公式

另一个常见的需求是将多个公式组在一起公用一个编号，编号位于公式的居中位置。为此，`amsmath` 宏包提供了诸如 `aligned`、`gathered` 等环境，与 `equation` 环境套用。以 `-ed` 结尾的环境用法与前一节不以 `-ed` 结尾的环境用法一一对应。我们仅以 `aligned` 举例：

```
\begin{equation}
\begin{aligned}
a &= b + c \\
d &= e + f + g \\
h + i &= j + k \\
l + m &= n
\end{aligned}
\end{equation}
```

$$a = b + c$$

$$d = e + f + g$$

$$h + i = j + k$$

$$l + m = n$$

(4.14)

4.5 数组和矩阵

为了排版二维数组，`LaTeX` 提供了 `array` 环境，用法与 `tabular` 环境极为类似，也需要定义列格式，并用 `\\` 换行。数组可作为一个子公式，在外套用 `\left`、`\right` 等定界符：

```
\[
\mathbf{X} = \left(
\begin{array}{ccc}
x_1 & x_2 & \ldots \\
x_3 & x_4 & \ldots \\
\vdots & \vdots & \ddots
\end{array}
\right)
\]
```

$$\mathbf{X} = \begin{pmatrix} x_1 & x_2 & \cdots \\ x_3 & x_4 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

我们还可以利用空的定界符排版出这样的效果：

```
\[
|x| = \left\{
\begin{array}{rl}
-x & \text{if } x < 0, \\
0 & \text{if } x = 0, \\
x & \text{if } x > 0.
\end{array}
\right.
\]
```

$$|x| = \begin{cases} -x & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ x & \text{if } x > 0. \end{cases}$$

不过上述例子可以用 `amsmath` 提供的 `cases` 环境更轻松地完成：

```
\[
|x| =
\begin{cases}
-x & \text{if } x < 0, \\
0 & \text{if } x = 0, \\
x & \text{if } x > 0.
\end{cases}
\]
```

$$|x| = \begin{cases} -x & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ x & \text{if } x > 0. \end{cases}$$

我们当然也可以用 `array` 环境排版各种矩阵，不过 `amsmath` 宏包还提供了种种排版矩阵的环境，有不带定界符的 `matrix`，有带各种定界符的矩阵 `pmatrix`、`bmatrix`、`Bmatrix`、`vmatrix`、`Vmatrix`。使用这些环境时，无需给定列格式：

```
\[
\begin{matrix}
1 & 2 \\
3 & 4
\end{matrix} \quad
\begin{bmatrix}
p_{11} & p_{12} & \ldots & p_{1n} \\
& p_{21} & p_{22} & \ldots & p_{2n} \\
& & \ddots & & \\
& & & p_{m1} & p_{m2} & \cdots & p_{mn}
\end{bmatrix}
\]
```

$$\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix} \quad \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mn} \end{bmatrix}$$

4.6 公式中的空格

前文提到过，绝大部分时候，数学公式中各元素的间距是根据符号类型自动生成的，需要我们手动调整的情况极少。我们已经认识了两个生成间距的命令 `\quad` 和 `\qquad`。在公式中我们还可能用到的间距包括 `\,`、`\:`、`\;` 以及负间距 `\!`。文本中的 `_` 也能使用在数学公式中。

无额外间距		aa
<code>\quad</code>	\quad	$a \quad a$
<code>\qquad</code>	\qquad	$a \qquad a$
<code>\,</code>	$\,$	$a \, a$
<code>\:</code>	$\:$	$a : a$
<code>\;</code>	$\;$	$a ; a$
<code>\!</code>	$\!$	$a \! a$
<code>_</code>	$_$	$a _ a$

一个常见的用途是修正积分的被积函数 $f(x)$ 和微元 dx 之间的距离。注意微元里的 d 用的是直立体：

```
\[
\int_a^b f(x) \mathrm{d}x
\qquad
\int_a^b f(x) \_, \mathrm{d}x
\]
```

$$\int_a^b f(x) dx \qquad \int_a^b f(x) \mathrm{d}x$$

另一个用途是生成多重积分号。如果我们直接连写两个 `\int`，之间的间距将会过宽，此时可以使用负间距 `\!` 修正之。不过 `amsmath` 提供了更方便的多重积分号，如二重积分 `\iint`、三重积分 `\iiint` 等。

表 4.3: 数学符号尺寸。

<code>\displaystyle</code>	行间公式尺寸	$\sum a$
<code>\textstyle</code>	行内公式尺寸	$\sum a$
<code>\scriptstyle</code>	上下标尺寸	a
<code>\scriptscriptstyle</code>	次级上下标尺寸	a

```
\[
P = \frac{\displaystyle{
\sum_{i=1}^n (x_i - x)
(y_i - y)}}{
{\displaystyle{\left[
\sum_{i=1}^n (x_i - x)^2
\sum_{i=1}^n (y_i - y)^2
\right]^{1/2}}}}
\]
```

$$P = \frac{\sum_{i=1}^n (x_i - x)(y_i - y)}{\left[\sum_{i=1}^n (x_i - x)^2 \sum_{i=1}^n (y_i - y)^2 \right]^{1/2}}$$

4.7.3 加粗的数学符号

在 \LaTeX 中为符号切换数学字体并不十分自由，只能通过 `\mathbf` 等有限的命令切换字体。比如想得到粗斜体的符号，就没有现成的命令³；再比如 `\mathbf` 只能改变拉丁字母，希腊字母就没有用。

\LaTeX 提供了一个命令 `\boldmath` 令用户可以将整套数学字体切换为粗体版本。但这个命令只能在公式外使用：

```
$\mu, M \quad \quad
\mathbf{\mu}, \mathbf{M}$
\quad \{\boldmath$\mu, M$\}
```

 $\mu, M \quad \mu, \mathbf{M} \quad \boldsymbol{\mu}, \mathbf{M}$

`amsmath` 包含的子宏包 `amsbsy` 提供了一个 `\boldsymbol` 命令，用于打破 `\boldmath` 的限制，在公式内部将一部分符号切换为粗体。

```
$\mu, M \quad \quad
\boldsymbol{\mu}, \boldsymbol{M}$
```

 $\mu, M \quad \boldsymbol{\mu}, \mathbf{M}$

然而定界符、巨算符等一些符号本身没有粗体版本，`\boldsymbol` 也得不到粗体。 \LaTeX 工具宏集之一的 `bm` 宏包可以用 `\bm` 生成“伪粗体”，一定程度上解决了不带粗体版本的符号的问题。这里不做过多介绍，需要使用的用户请参考 `bm` 宏包的帮助文档。

4.8 定理环境

使用 \LaTeX 排版数学和其他科技文档时，会接触到大量的定理、证明等内容。 \LaTeX 提供了一个基本的命令 `\newtheorem` 提供定理环境的定义：

```
\newtheorem{<type>}{<title>}[<section-name>]
\newtheorem{<type>}[<counter>]{<title>}
```

³国内可能还有使用粗斜体表示向量符号的习惯，但这并不是正确的习惯。

其中 $\langle type \rangle$ 为定理类型的名称, 作为一个环境来使用。 $\langle title \rangle$ 是定理类型的标签(“定理”, “公理”等), 排版在序号之前。

定理的序号由两个可选参数之一决定, 它们**不能同时使用**:

- $\langle section name \rangle$ 为章节名称, 这使定理序号成为章节的下一级序号;
- $\langle counter \rangle$ 为用 `\newcounter` 自定义的计数器名称 (详见 8.3 节), 定理序号由这个计数器管理。

如果两个可选参数都不用的话, 则使用一个默认的计数器。

例如, 我们用以下代码定义了一个 `mytheorem` 环境:

```
\newtheorem{mytheorem}{My Theorem}[chapter]
```

于是我们可以使用 `mytheorem` 环境排版定理。定理带一个可选参数, 用于注明定理的名称, 如“法拉第定律”等。在环境内还可以用 `\label` 声明引用:

```
\begin{mytheorem}\label{thm:light}
The light speed in vaccum
is $299,792,458 \mathrm{m/s}$.
\end{mytheorem}
\begin{mytheorem}[Relativity]
The relationship of energy,
momentum and mass is
 $E^2 = m^2 c^4 + p^2 c^2$ ,
where  $c$  is the light speed
described in theorem \ref{thm:light}.
\end{mytheorem}
```

My Theorem 4.1. *The light speed in vaccum is 299,792,458m/s.*

My Theorem 4.2 (Relativity). *The relationship of energy, momentum and mass is $E^2 = m^2 c^4 + p^2 c^2$, where c is the light speed described in theorem 4.1.*

4.8.1 amsthm 宏包

L^AT_EX 只给了原始的证明环境格式 (粗体标签、斜体正文、定理名用小括号包裹)。如果需要修改格式, 则要依赖其它的宏包, 如 `amsthm`、`ntheorem` 等等。本小节简单介绍一下 `amsthm` 的用法。

`amsthm` 提供了 `\theoremstyle` 命令支持定理格式的切换, 在用 `\newtheorem` 命令定义定理环境之前使用。`amsthm` 预定义了三种格式用于 `\theoremstyle: plain` 和 L^AT_EX 原始的格式一致; `description` 使用粗体标签、正体内容; `remark` 使用斜体标签、正体内容。

另外 `amsthm` 还支持用带星号的 `\newtheorem*` 定义不带序号的定理环境:

```
\theoremstyle{definition} \newtheorem{law}{Law}
\theoremstyle{plain} \newtheorem{jury}[law]{Jury}
\theoremstyle{remark} \newtheorem*{mar}{Margaret}
```

以上例子定义的 `jury` 环境与 `law` 环境共用编号, `mar` 环境不编号:

```
\begin{law} \label{law:box}
Don' t hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law~\ref{law:box}.\end{jury}
\begin{jury}
You will disregard the last
statement.\end{jury}
\begin{mar}No, No, No\end{mar}
\begin{mar}Denis!\end{mar}
```

Law 1. Don' t hide in the witness box

Jury 2 (The Twelve). *It could be you!*
So beware and see law 1.

Jury 3. *You will disregard the last state-*
ment.

Margaret. No, No, No

Margaret. Denis!

amsthm 还支持使用 `\newtheoremstyle` 命令自定义定理格式, 更为方便使用的是 `ntheorem` 宏包。感兴趣的读者可参阅它们的帮助手册。

4.8.2 证明环境和证毕符号

amsthm 还提供了一个 `proof` 环境用于排版定理的证明过程。`proof` 环境末尾自动加上一个 □ 证毕符号:

```
\begin{proof}
For simplicity, we use
\[
E=mc^2.
\]
That's it.
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2.$$

That's it. □

如果行末是一个不带编号的公式, □ 符号会另起一行, 这时可使用 `\qedhere` 命令将 □ 符号放在公式末尾:

```
\begin{proof}
For simplicity, we use
\[
E=mc^2.\qedhere
\]
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2.$$

□

`\qedhere` 对于 `align*` 等命令也有效:

```
\begin{proof}
Assuming  $\gamma$ 
 $= 1/\sqrt{1-v^2/c^2}$ , then
\begin{align*}
E &= \gamma m_0 c^2 \\
p &= \gamma m_0 v \qedhere
\end{align*}
\end{proof}
```

Proof. Assuming $\gamma = 1/\sqrt{1-v^2/c^2}$,
then

$$E = \gamma m_0 c^2$$

$$p = \gamma m_0 v$$

□

在使用带编号的公式时，建议最好**不要使用** `\qedhere` 命令，而是让 `proof` 环境自动生成。对带编号的公式使用 `\qedhere` 命令会使 \square 符号放在一个难看的位置，紧贴着公式：

```
\begin{proof}
For simplicity, we use
\begin{equation}
E=mc^2.\qedhere
\end{equation}
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2. \quad (4.15)$$

\square

在 `\align` 中使用则会盖掉公式的编号；使用 `equation` 嵌套 `aligned` 环境时，`\qedhere` 会将 \square 直接放在公式后。这些位置都不太正常。

4.9 符号表

有几个注意事项：

1. 蓝色的命令依赖 `amsmath` 宏包（非 `amssymb` 宏包）；
2. 带有角标 ^{*a*} 的符号命令依赖 `latexsym` 宏包。

4.9.1 L^AT_EX 普通符号

表 4.4: 文本/数学模式通用符号

这些符号可用于文本和数学模式。

{	\{	}	\}	\$	\\$	%	\%
†	\dag	§	\S	©	\copyright	...	\dots
‡	\ddag	¶	\P	£	\pounds		

表 4.5: 希腊字母。

`\Alpha`, `\Beta` 等希腊字母符号不存在，因为它们和拉丁字母 A,B 等一模一样；小写字母里也不存在 `\omicron`，直接用 *o* 代替。

α	<code>\alpha</code>	θ	<code>\thetaeta</code>	<i>o</i>	<code>o</code>	υ	<code>\upsilonlon</code>
β	<code>\betaeta</code>	ϑ	<code>\varthetaeta</code>	π	<code>\pi</code>	ϕ	<code>\phi</code>
γ	<code>\gamma</code>	ι	<code>\iota</code>	ϖ	<code>\varpi</code>	φ	<code>\varphi</code>
δ	<code>\delta</code>	κ	<code>\kappa</code>	ρ	<code>\rho</code>	χ	<code>\chi</code>
ϵ	<code>\epsilon</code>	λ	<code>\lambda</code>	ϱ	<code>\varrho</code>	ψ	<code>\psi</code>
ε	<code>\varepsilon</code>	μ	<code>\mu</code>	σ	<code>\sigma</code>	ω	<code>\omega</code>
ζ	<code>\zeta</code>	ν	<code>\nu</code>	ς	<code>\varsigma</code>		
η	<code>\eta</code>	ξ	<code>\xi</code>	τ	<code>\tau</code>		
Γ	<code>\Gamma</code>	Λ	<code>\Lambda</code>	Σ	<code>\Sigma</code>	Ψ	<code>\Psi</code>
Δ	<code>\Delta</code>	Ξ	<code>\Xi</code>	Υ	<code>\Upsilon</code>	Ω	<code>\Omega</code>
Θ	<code>\Theta</code>	Π	<code>\Pi</code>	Φ	<code>\Phi</code>		

表 4.6: 二元关系符。

所有的二元关系符都可以加 `\not` 前缀得到相反意义的关系符，例如 `\not=` 就得到不等号（同 `\ne`）。

$<$	<code><</code>	$>$	<code>></code>	$=$	<code>=</code>
\leq	<code>\leq</code> or <code>\le</code>	\geq	<code>\geq</code> or <code>\ge</code>	\equiv	<code>\equiv</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	$\dot{=}$	<code>\doteq</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\sim	<code>\sim</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\simeq	<code>\simeq</code>
\subset	<code>\subset</code>	\supset	<code>\supset</code>	\approx	<code>\approx</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cong	<code>\cong</code>
\sqsubset^a	<code>\sqsubset^a</code>	\sqsupset^a	<code>\sqsupset^a</code>	\Join^a	<code>\Join^a</code>
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\bowtie	<code>\bowtie</code>
\in	<code>\in</code>	\ni , \owns	<code>\ni</code> , <code>\owns</code>	\propto	<code>\propto</code>
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\models	<code>\models</code>
\mid	<code>\mid</code>	\parallel	<code>\parallel</code>	\perp	<code>\perp</code>
\smile	<code>\smile</code>	\frown	<code>\frown</code>	\asymp	<code>\asymp</code>
$:$	<code>:</code>	\notin	<code>\notin</code>	\neq or \ne	<code>\neq</code> or <code>\ne</code>

表 4.7: 二元运算符。

$+$	<code>+</code>	$-$	<code>-</code>	
\pm	<code>\pm</code>	\mp	<code>\mp</code>	\triangleleft <code>\triangleleft</code>
\cdot	<code>\cdot</code>	\div	<code>\div</code>	\triangleright <code>\triangleright</code>
\times	<code>\times</code>	\backslash	<code>\setminus</code>	\star <code>\star</code>
\cup	<code>\cup</code>	\cap	<code>\cap</code>	\ast <code>\ast</code>
\sqcup	<code>\sqcup</code>	\sqcap	<code>\sqcap</code>	\circ <code>\circ</code>
\vee , <code>\lor</code>		\wedge <code>\land</code>		\bullet <code>\bullet</code>
\oplus	<code>\oplus</code>	\ominus	<code>\ominus</code>	\diamond <code>\diamond</code>
\odot	<code>\odot</code>	\oslash	<code>\oslash</code>	\uplus <code>\uplus</code>
\otimes	<code>\otimes</code>	\bigcirc	<code>\bigcirc</code>	\amalg <code>\amalg</code>
\bigtriangleup	<code>\bigtriangleup</code>	\bigtriangledown	<code>\bigtriangledown</code>	\dagger <code>\dagger</code>
\lhd ^{<i>a</i>}		\rhd ^{<i>a</i>}		\ddagger <code>\ddagger</code>
\unlhd ^{<i>a</i>}		\unrhd ^{<i>a</i>}		\wr <code>\wr</code>

表 4.8: 巨算符。

Σ	<code>\sum</code>	\bigcup	<code>\bigcup</code>	\bigvee	<code>\bigvee</code>
\prod	<code>\prod</code>	\bigcap	<code>\bigcap</code>	\bigwedge	<code>\bigwedge</code>
\coprod	<code>\coprod</code>	\bigsqcup	<code>\bigsqcup</code>	\biguplus	<code>\biguplus</code>
\int	<code>\int</code>	\oint	<code>\oint</code>	\bigodot	<code>\bigodot</code>
\bigoplus	<code>\bigoplus</code>	\bigotimes	<code>\bigotimes</code>		

表 4.9: 数学重音符号。

\hat{a}	<code>\hat{a}</code>	\check{a}	<code>\check{a}</code>	\tilde{a}	<code>\tilde{a}</code>
\grave{a}	<code>\grave{a}</code>	\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>
\bar{a}	<code>\bar{a}</code>	\vec{a}	<code>\vec{a}</code>	\widehat{AAA}	<code>\widehat{AAA}</code>
\acute{a}	<code>\acute{a}</code>	\breve{a}	<code>\breve{a}</code>	\widetilde{AAA}	<code>\widetilde{AAA}</code>
\mathring{a}	<code>\mathring{a}</code>				

表 4.10: 箭头。

\leftarrow	<code>\leftarrow</code> or <code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>
\rightarrow	<code>\rightarrow</code> or <code>\to</code>	\longrightarrow	<code>\longrightarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>
\hookrightarrow	<code>\hookrightarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>
\rightleftharpoons	<code>\rightleftharpoons</code>	\iff	<code>\iff</code>
\uparrow	<code>\uparrow</code>	\downarrow	<code>\downarrow</code>
\Uparrow	<code>\Uparrow</code>	\Updownarrow	<code>\Updownarrow</code>
\nearrow	<code>\nearrow</code>	\searrow	<code>\searrow</code>
\swarrow	<code>\swarrow</code>	\nwarrow	<code>\nwarrow</code>
\leadsto	<code>\leadsto</code> ^a		

表 4.11: 作为重音的箭头符号。

\overrightarrow{AB}	<code>\overrightarrow{AB}</code>	\overleftarrow{AB}	<code>\underrightarrow{AB}</code>
\overleftarrow{AB}	<code>\overleftarrow{AB}</code>	\overrightarrow{AB}	<code>\underleftarrow{AB}</code>
\overleftrightarrow{AB}	<code>\overleftrightharrow{AB}</code>	\overleftrightarrow{AB}	<code>\underleftrightharrow{AB}</code>

表 4.12: 定界符。

(())	↑	<code>\uparrow</code>
[[or <code>\lbrack</code>]] or <code>\rbrack</code>	↓	<code>\downarrow</code>
{	<code>\{</code> or <code>\lbrace</code>	}	<code>\}</code> or <code>\rbrace</code>	↕	<code>\updownarrow</code>
⟨	<code>\langle</code>	⟩	<code>\rangle</code>	↗	<code>\Uparrow</code>
	or <code>\vert</code>		<code>\ </code> or <code>\Vert</code>	↘	<code>\Downarrow</code>
/	/	\	<code>\backslash</code>	↕	<code>\Updownarrow</code>
⌊	<code>\lfloor</code>	⌋	<code>\rfloor</code>		
⌈	<code>\rceil</code>	⌋	<code>\lceil</code>		

表 4.13: 用于行间公式的大定界符。

(<code>\lggroup</code>)	<code>\rgroup</code>	(<code>\lmoustache</code>
	<code>\arrowvert</code>		<code>\Arrowvert</code>		<code>\bracevert</code>
}	<code>\rmoustache</code>				

表 4.14: 其他符号。

...	<code>\dots</code>	...	<code>\cdots</code>	⋮	<code>\vdots</code>	⋯	<code>\ddots</code>
\hbar	<code>\hbar</code>	\imath	<code>\imath</code>	\jmath	<code>\jmath</code>	ℓ	<code>\ell</code>
\Re	<code>\Re</code>	\Im	<code>\Im</code>	\aleph	<code>\aleph</code>	\wp	<code>\wp</code>
\forall	<code>\forall</code>	\exists	<code>\exists</code>	\mho	<code>\mho</code>	∂	<code>\partial</code>
'	<code>'</code>	'	<code>\prime</code>	\emptyset	<code>\emptyset</code>	∞	<code>\infty</code>
∇	<code>\nabla</code>	\triangle	<code>\triangle</code>	\Box	<code>\Box</code>	\Diamond	<code>\Diamond</code>
\bot	<code>\bot</code>	\top	<code>\top</code>	\angle	<code>\angle</code>	\surd	<code>\surd</code>
\diamondsuit	<code>\diamondsuit</code>	\heartsuit	<code>\heartsuit</code>	\clubsuit	<code>\clubsuit</code>	\spadesuit	<code>\spadesuit</code>
\neg	<code>\neg</code> or <code>\not</code>	\flat	<code>\flat</code>	\natural	<code>\natural</code>	\sharp	<code>\sharp</code>

4.9.2 \mathcal{AMS} 符号

本小节所有表格的符号依赖 `amssymb` 宏包。

表 4.15: \mathcal{AMS} 定界符

\ulcorner	<code>\ulcorner</code>	\urcorner	<code>\urcorner</code>	\llcorner	<code>\llcorner</code>	\lrcorner	<code>\lrcorner</code>
\lvert	<code>\lvert</code>	\rvert	<code>\rvert</code>	\lVert	<code>\lVert</code>	\rVert	<code>\rVert</code>

表 4.16: \mathcal{AMS} 希腊字母和希伯来字母。

\digamma	<code>\digamma</code>	\varkappa	<code>\varkappa</code>	\beth	<code>\beth</code>	\gimel	<code>\gimel</code>	\daleth	<code>\daleth</code>
------------	-----------------------	-------------	------------------------	---------	--------------------	----------	---------------------	-----------	----------------------

表 4.17: \mathcal{AMS} 二元运算符。

$\dot{+}$	<code>\dotplus</code>	\cdot	<code>\centerdot</code>		
\ltimes	<code>\ltimes</code>	\rtimes	<code>\rtimes</code>	\div	<code>\divideontimes</code>
\mathbb{U}	<code>\doublecup</code>	\mathbb{M}	<code>\doublecap</code>	\smallsetminus	<code>\smallsetminus</code>
\veebar	<code>\veebar</code>	$\bar{\wedge}$	<code>\barwedge</code>	$\overline{\bar{\wedge}}$	<code>\doublebarwedge</code>
\boxplus	<code>\boxplus</code>	\boxminus	<code>\boxminus</code>	\ominus	<code>\circleddash</code>
\boxtimes	<code>\boxtimes</code>	\boxdot	<code>\boxdot</code>	\odot	<code>\circledcirc</code>
\intercal	<code>\intercal</code>	\circledast	<code>\circledast</code>	\times	<code>\rightthreetimes</code>
\curlyvee	<code>\curlyvee</code>	\curlywedge	<code>\curlywedge</code>	\times	<code>\leftthreetimes</code>

表 4.18: \mathcal{AMS} 二元关系符。

\lessdot	\gtrdot	\doteqdot
\leqslant	\geqslant	\risingdotseq
\eqslantless	\eqslantgtr	\fallingdotseq
\leqq	\geqq	\eqcirc
\lll or \llless	\ggg	\circeq
\lesssim	\gtrsim	\triangleq
\lessapprox	\gtrapprox	\bumpeq
\lessgtr	\gtrless	\Bumpeq
\lesseqgtr	\gtreqless	\thicksim
\lesseqqgtr	\gtreqqless	\thickapprox
\preccurlyeq	\succcurlyeq	\approxeq
\curlyeqprec	\curlyeqsucc	\backsim
\precsim	\succsim	\backsimeq
\precapprox	\succapprox	\vDash
\subseteq	\supseteq	\Vdash
\shortparallel	\Supset	\Vvdash
\blacktriangleleft	\sqsupset	\backepsilon
\vartriangleright	\because	\varpropto
\blacktriangleright	\Subset	\between
\trianglerighteq	\smallfrown	\pitchfork
\vartriangleleft	\shortmid	\smallsmile
\trianglelefteq	\therefore	\sqsubset

表 4.19: \mathcal{AMS} 箭头。

\dashleftarrow	<code>\dashleftarrow</code>	\dashrightarrow	<code>\dashrightarrow</code>
\Lleftarrow	<code>\leftleftarrows</code>	\Rrightarrow	<code>\rightrightarrows</code>
\Leftrightarrow	<code>\leftrightharpoons</code>	\Rrightarrow	<code>\rightleftarrows</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>
\twoheadleftarrow	<code>\twoheadleftarrow</code>	\twoheadrightarrow	<code>\twoheadrightarrow</code>
\leftarrowtail	<code>\leftarrowtail</code>	\rightarrowtail	<code>\rightarrowtail</code>
\leftrightharpoons	<code>\leftrightharpoons</code>	\rightleftharpoons	<code>\rightleftharpoons</code>
\looparrowleft	<code>\Lsh</code>	\looparrowright	<code>\Rsh</code>
\curvearrowleft	<code>\looparrowleft</code>	\curvearrowright	<code>\looparrowright</code>
\circlearrowleft	<code>\curvearrowleft</code>	\circlearrowright	<code>\curvearrowright</code>
\multimap	<code>\circlearrowleft</code>	\Uparrow	<code>\circlearrowright</code>
\downdownarrows	<code>\multimap</code>	\Uparrow	<code>\upuparrows</code>
\upharpoonright	<code>\downdownarrows</code>	\Uparrow	<code>\upuparrows</code>
\rightsquigarrow	<code>\upharpoonright</code>	\Downarrow	<code>\downharpoonright</code>
	<code>\rightsquigarrow</code>	\rightsquigarrow	<code>\leftrightsquigarrow</code>

表 4.20: \mathcal{AMS} 反义二元关系符和箭头。

\nless	\ngtr	\varsubsetneqq
\lneq	\gneq	\varsupsetneqq
\nleq	\ngeq	\nsubseteqeq
\nleqslant	\ngeqslant	\nsupseteqeq
\lneqq	\gneqq	\nmid
\lvertneqq	\gvertneqq	\nparallel
\nleqq	\ngeqq	\nshortmid
\lnsim	\gnsim	\nshortparallel
\lnapprox	\gnapprox	\nsim
\nprec	\nsucc	\ncong
\npreceq	\nsucceq	\nvdash
\precneqq	\succneqq	\nvDash
\precnsim	\succnsim	\nVdash
\precnapprox	\succnapprox	\nVDash
\subsetneq	\supsetneq	\ntriangleleft
\varsubsetneq	\varsupsetneq	\ntriangleright
\nsubseteq	\nsupseteq	\ntrianglelefteq
\subseteqeq	\supseteqeq	\ntrianglerighteq
\nleftarrow	\rightarrow	\nleftrightarrow
\nLeftarrow	\nRightarrow	

表 4.21: \mathcal{AMS} 其它符号。

\hbar	\hslash	\Bbbk
\square	\blacksquare	\textcircled{S}
\vartriangle	\blacktriangle	\complement
\triangledown	\blacktriangledown	\Game
\lozenge	\blacklozenge	\bigstar
\angle	\measuredangle	
\diagup	\diagdown	\backprime
\nexists	\Finv	\varnothing
\eth	\sphericalangle	\mho

第五章 排版样式设定

至此你已经基本学会排版内容丰富的文档，标题、目录、章节、公式、列表、图片、表格等等应有尽有。但是你可能已经有点不甘心了，因为似乎你排版出来的文档是千篇一律的模样—— \LaTeX 默认的字体、单调的页眉页脚、不太令你满意的页边距，等等。本章的内容将带你一览如何修改 \LaTeX 的排版样式。

5.1 字体和字号

\LaTeX 根据文档的逻辑结构（章节、脚注等）来选择合适的字体样式以及字号。在某些情况下，你可能会想要在文档中选用不一样的字体样式和字号。你可以使用表 5.1 和表 5.2 中列出的命令以达到修改字体的目的。

```
{\small The small and  
\textbf{bold} Romans ruled}  
\Large all of great big  
\itshape Italy}.
```

The small and **bold** Romans ruled all of
great big *Italy*.

\LaTeX 2_ϵ 的一个重要特征是：字体的各种属性是相互独立的，这意味着你可以改变字体的大小，而仍然保留字体原有的粗体或者斜体的特性。

5.1.1 字体样式

\LaTeX 提供了两组修改字体的命令，见表 5.1。其中诸如 `\bfseries` 形式的命令将会影响之后所有的字符，如果想要让它在局部生效，需要用花括号建立分組，也就是写成 `{\bfseries <some text>}` 这样的形式；对应的 `\textbf` 形式带一个参数，它只会改变参数内部的字体，相比之下更为常用。

在公式中，直接使用 `\textbf` 等命令不会起效，甚至报错。 \LaTeX 已有修改数学字体的命令，详见 4.7.1 小节。

5.1.2 字号

字号命令实际大小依赖于所使用的文档类及其选项。表 5.3 列出了这些命令在标准文档类中的绝对大小，单位为 pt。

使用字号命令的时候，通常也需要用花括号进行分組，如同 `\rmfamily` 那样。

```
He likes {\LARGE large and  
\small small} letters}.
```

He likes large and small letters.

\LaTeX 还提供了一个基础的命令 `\fontsize` 用于设定任意大小的字号：

表 5.1: 字体命令。

<code>\rmfamily</code>	<code>\textrm{...}</code>	roman
<code>\sffamily</code>	<code>\textsf{...}</code>	sans serif
<code>\ttfamily</code>	<code>\texttt{...}</code>	typewriter
<code>\mdseries</code>	<code>\textmd{...}</code>	medium
<code>\bfseries</code>	<code>\textbf{...}</code>	bold face
<code>\upshape</code>	<code>\textup{...}</code>	upright
<code>\itshape</code>	<code>\textit{...}</code>	<i>italic</i>
<code>\slshape</code>	<code>\textsl{...}</code>	<i>slanted</i>
<code>\scshape</code>	<code>\textsc{...}</code>	SMALL CAPS
<code>\em</code>	<code>\emph{...}</code>	<i>emphasized</i>
<code>\normalfont</code>	<code>\textnormal{...}</code>	normal font

表 5.2: 字号。

<code>\tiny</code>	tiny font	<code>\Large</code>	larger font
<code>\scriptsize</code>	very small font	<code>\LARGE</code>	very large font
<code>\footnotesize</code>	quite small font	<code>\huge</code>	huge
<code>\small</code>	small font	<code>\Huge</code>	largest
<code>\normalsize</code>	normal font		
<code>\large</code>	large font		

表 5.3: 标准文档类中的字号大小。

字号	10pt 选项 (默认)	11pt 选项	12pt 选项
<code>\tiny</code>	5pt	6pt	6pt
<code>\scriptsize</code>	7pt	8pt	8pt
<code>\footnotesize</code>	8pt	9pt	10pt
<code>\small</code>	9pt	10pt	10.95pt
<code>\normalsize</code>	10pt	10.95pt	12pt
<code>\large</code>	12pt	12pt	14.4pt
<code>\Large</code>	14.4pt	14.4pt	17.28pt
<code>\LARGE</code>	17.28pt	17.28pt	20.74pt
<code>\huge</code>	20.74pt	20.74pt	24.88pt
<code>\Huge</code>	24.88pt	24.88pt	24.88pt

```
\fontsize{⟨size⟩}{⟨base line-skip⟩}
```

`\fontsize` 用到两个参数, $\langle size \rangle$ 为字号, $\langle base\ line-skip \rangle$ 为基础行距。表 5.3 中的命令也都各自设定了与字号对应的基础行距, 大小为字号的 1.2 倍。如果不是在导言区, `\fontsize` 的设定需要 `\selectfont` 命令才能立即生效, 而表 5.2 的字号设定都是立即生效的。

任意设置字号有个问题: \LaTeX 排版用到的一些老式字体是固定字号的¹, 它们往往无法自动调整成任意的字号大小。如果可能的话, 应当尽量使用表 5.2 中的命令设置字号。

5.1.3 选用字体宏包

尽管到了这里你知道了如何切换粗体、斜体等等, 以及如何改变字号, 但你依然用着 \LaTeX 默认的那套、由 \TeX 程序的开发者高德纳亲自制作的 Computer Modern 字体。有的人可能很喜欢 Times 或者 Palatino, 或者更好看的字体。这些字体样式的自由设置在 \LaTeX 里还不太容易。

幸好大部分时候, 许多字体宏包为我们完成了整套配置, 我们可以在调用宏包之后, 照常使用 `\bfseries` 或 `\ttfamily` 等我们熟悉的命令。表 5.4 列出了较为常用的字体宏包, 其中相当多的字体包还配置了数学字体, 或者文本、数学字体兼而有之。更多的字体配置参考 [17, 18]。

5.1.4 \XeLaTeX 下使用 `fontspec` 宏包更改字体

\XeLaTeX 能够支持直接调用系统安装的 `.ttf` 或 `.otf` 格式字体²。相比于上一小节, 我们有了更多修改字体的余地。

\XeLaTeX 下支持用户调用字体的宏包是 `fontspec`。宏包提供了几个设置全局字体的命令, 设置 `\rmfamily` 等对应命令的默认字体³:

```
\setmainfont[⟨font features⟩]{⟨font name⟩}
\setsansfont[⟨font features⟩]{⟨font name⟩}
\setmonofont[⟨font features⟩]{⟨font name⟩}
```

其中 $\langle font\ name \rangle$ 使用字体的文件名(带扩展名)或者字体的英文名称。 $\langle font\ features \rangle$ 用来手动配置对应的粗体或斜体(如果有配套的粗体和斜体, 则自动设置), 如 `BoldFont=⟨font name⟩`, `ItalicFont=⟨font name⟩`。 $\langle font\ features \rangle$ 还能配置字体本身的各种特性, 这里不再赘述, 感兴趣的读者请参考 `fontspec` 宏包的帮助文档。

需要注意一点: `fontspec` 宏包会覆盖数学字体设置。需要使用 5.4 中列出的一些数学字体宏包时, 应当在调用 `fontspec` 宏包时加上 `no-math` 选项。`fontspec` 宏包可能由其它宏包(比如 `xeCJK`)加载时, 则在文档类一开头的 `\documentclass` 里加上 `no-math` 选项。

5.1.5 使用 `xeCJK` 宏包更改中文字体

前文已经介绍过的 `xeCJK` 宏包使用了和 `fontspec` 宏包非常类似的语法设置中文字体:

```
\setCJKmainfont[⟨font features⟩]{⟨font name⟩}
\setCJKsansfont[⟨font features⟩]{⟨font name⟩}
\setCJKmonofont[⟨font features⟩]{⟨font name⟩}
```

¹不难看出表 5.3 中的许多字号大致呈等比数列, 比例为 1.2; 而 10.95pt 实际上是 $10 \times \sqrt{1.2}$ 。

²Linux 下的 \TeX Live 为了支持系统安装的字体, 需要额外的配置。详见附录 A。

³新版本 `fontspec` 的命令支持把必选参数 $\langle font\ name \rangle$ 放在可选参数 $\langle font\ features \rangle$ 的前面。

表 5.4: 常见的 L^AT_EX 字体宏包。

lmodern	Latin Modern 字体, 对 Computer Modern 字体的扩展
txfonts	Times 风格的字体宏包
pxfonts	Palation 风格的字体宏包
newtxtext,newtxmath	txfonts 的改进版本, 分别设置文本和数学字体
newpxtext,newpxmath	pxfonts 的改进版本, 分别设置文本和数学字体
mathptmx	psnfss 组件之一, Times 风格, 较为陈旧, 不推荐使用
mathpazo	psnfss 组件之一, Palatino 风格, 较为陈旧, 不推荐使用
ccfonts	Concrete 风格字体
euler	Euler 风格数学字体, 也出自于高德纳之手
fourier	fourier 风格数学字体
arev	Arev 风格的无衬线字体
cmbright	仿 Computer Modern 风格的无衬线字体
libertine	Linux Libertine 衬线字体
droid	Droid Serif/Droid Sans 等
inconsolata	Inconsolata 一款不错的开源等宽字体
sourcesanspro	Source Sans Pro 开源无衬线字体
sourcecodepro	Source Code Pro 开源等宽字体
mathabx	数学符号宏包之一
MnSymbol	数学符号宏包之一
fdsymbol	数学符号宏包之一
mathdesign	配合 Charter/Garamond/Utopia 正文字体的数学字体宏包 (Garamond 字体可能需要单独安装)

由于中文字体少有对应的粗体或斜体, *font features* 里多用其他字体来配置, 比如许多人习惯将宋体的 `BoldFont` 配置为黑体, 而 `ItalicFont` 配置为楷体。

5.2 段落格式和间距

5.2.1 设置和修改长度

本节我们将首次接触到对长度的操作。

5.2.2 行距

前文中我们提到过 `\fontsize` 命令可以修改行距, 但我们很少直接用这个命令, 常用的是 `\linespread` 命令, 它通常放在导言区:

```
\linespread{<factor>}
```

`\linespread` 命令的参数 *factor* 还并不是最终的行距。前文所叙述的 `\fontsize` 命令设置了一个基本行距。`\small`、`\large` 等命令也设置了 1.2 倍的基本行距。而 `\linespread` 相当于在 1.2 倍的基础上乘以 *factor*。所以要想设定 1.5 倍行距的话, 应当设 `\linespread{1.25}`。

若是想要局部地改变某个段落的行距, 需要用到前文所叙述的 `\selectfont` 命令使改动立即生效 (本手册为使汉字和拉丁文字能够搭配, 设定了 1.5 倍行距, 我们用一个 2 倍行距的例子对比):

```
{\linespread{1.67}\selectfont
This paragraph is typeset with
the baseline skip set to 2.0 times
the font size. Note the par
command at the end of the
paragraph.\par}
```

This paragraph has a clear purpose, it shows that after the curly brace has been closed, everything is back to normal.

This paragraph is typeset with the baseline skip set to 2.0 times the font size. Note the par command at the end of the paragraph.

This paragraph has a clear purpose, it shows that after the curly brace has been closed, everything is back to normal.

字号的改变是即时生效的, 而行距的改变直到文字分段时才生效。如果你想改变某一部分文字的行距, 那么不能简单地将文字包含在花括号内。注意下面两个例子中 `\par` 命令的位置⁴。

```
{\Large Don't read this!
It is not true.
You can believe me!\par}
```

Don't read this! It is not true. You can believe me!

```
{\Large This is not true either.
But remember I am a liar.}\par}
```

This is not true either. But remember I am a liar.

⁴`\par` 的作用相当于一个空行。

5.2.3 段落格式

以下命令分别设置段落的左缩进、右缩进和首行缩进：

```
\setlength{\leftskip}{20pt}
\setlength{\rightskip}{20pt}
\setlength{\parindent}{2em}
```

它们和设置字号的命令一样，在分段时生效。

L^AT_EX 默认在段落开始时缩进，长度为你用上述命令设置的 `\parindent`。如果你在某一段不想使用缩进，可使用某一段开头使用

```
\noindent
```

命令。相反地，

```
\indent
```

命令强制开启一段首行缩进的段落。

L^AT_EX 默认在 `\chapter`、`\section` 等章节命令之后的第一段不缩进⁵。如果你想使之缩进，当然可以使用 `\indent` 逐个调整段落，但更简单的方式是在导言区使用 `indentfirst` 宏包：

```
\usepackage{indentfirst}
```

段与段之间的垂直间距为 `\parskip`：

```
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

如上命令设置段落间距为弹性长度，可在 `0.8ex` 到 `1.5ex` 变动。

5.2.4 水平间距

L^AT_EX 默认为单词之间增添了水平间距。我们可以用已经在数学公式中出现的 `\quad` 和 `\qqquad` 命令制造一个额外的间距。但是如果想要得到任意长度的间距，需要用到如下命令：

```
\hspace{<length>}
```

`\hspace` 命令生成的间距如果位于一行的开头或末尾，则有可能被“吞掉”。这时可以使用 `\hspace*` 代替 `\hspace` 命令得到不会因断行而消失的水平间距。

```
This\hspace{1.5cm}is a space
of 1.5 cm.
```

```
This          is a space of 1.5 cm.
```

命令 `\stretch` 生成一个特殊弹性长度，用在 `\hspace` 的参数里。它可以一直延伸，直到一行内可用的空间都被填满：

```
\stretch{<n>}
```

如果同一行内出现多个 `\hspace{\stretch{<n>}}`，这一行的所有可用空间将以每个 `\stretch` 设置的权重 `<n>` 进行分配。

⁵`ctex` 宏包按照中文习惯保持第一段的首行缩进。

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

x x x

在正文中用 `\hspace` 调节水平间距时，往往使用 `em` 作为单位，它会随字号大小而变：

```
{\Large}big\hspace{1em}y\\
{\tiny}tin\hspace{1em}y}
```

big y
tin y

5.2.5 垂直间距

在页面中，段落、章节标题、行间公式、列表、浮动体等元素之间的间距是 L^AT_EX 预设的。比如 `\parskip`，默认设置为 0pt plus 1pt。

如果我们想要人为地增加段落之间的垂直间距，可以在**两个段落之间**的位置使用如下命令：

```
\vspace{<length>}
```

`\vspace` 的间距在一页的顶端或底端可能被“吞掉”，类似 `\hspace` 在一行的开头和末尾那样。对应地，`\vspace*` 命令产生不会因断页而消失的垂直间距。

在段落内部的某两行之间增加垂直间距，一般通过给 `\` 命令加上可选参数。这个办法也可以用于表格：

```
\[<length>]
```

另外 L^AT_EX 还提供了 `\bigskip`、`\medskip`、`\smallskip` 来增加预定义长度的垂直间距。

5.3 页面和分栏

我们不妨回顾一下最开头的文档类属性。L^AT_EX 允许你通过文档类选项控制纸张的大小，包括 `a4paper`、`letterpaper`（美国纸张标准，8.5in×11in）等等，并配合字号设置了适合的页边距。这些页面参数由图 5.1 里给出的各种命令控制，可以用 `\setlength` 命令修改这些参数，以达到调节页面尺寸和边距的作用。

但是，如果你想要直接设置页边距等参数，着实是一件麻烦事。我们根据图 5.1 将各个方向的页边距计算公式给出（以奇数页为例）：

$$\begin{aligned}\langle left-margin \rangle &= 1\text{in} + \backslash\text{hoffset} + \backslash\text{oddsidemargin} \\ \langle right-margin \rangle &= \backslash\text{paperwidth} - \langle left-margin \rangle - \backslash\text{textwidth} \\ \langle top-margin \rangle &= 1\text{in} + \backslash\text{voffset} + \backslash\text{topmargin} + \backslash\text{headheight} + \backslash\text{headsep} \\ \langle bottom-margin \rangle &= \backslash\text{paperheight} - \langle top-margin \rangle - \backslash\text{textheight}\end{aligned}$$

如果我们想设置合适的 $\langle left-margin \rangle$ 和 $\langle right-margin \rangle$ ，就要靠上述方程组把 `\oddsidemargin` 和 `\textwidth` 等参数解出来！

幸好 `geometry` 宏包能够帮我们完成背后繁杂的计算，让我们能够用简便一些的方法设置页面参数。

5.3.1 利用 geometry 宏包设置页面参数

你既可以调用 `geometry` 宏包然后用其提供的 `\geometry` 命令设置页面参数：

The circle is at 1 inch from the top and left of the page. Dashed lines represent $(\backslash\hoffset + 1\text{ inch})$ and $(\backslash\voffset + 1\text{ inch})$ from the top and left of the page.

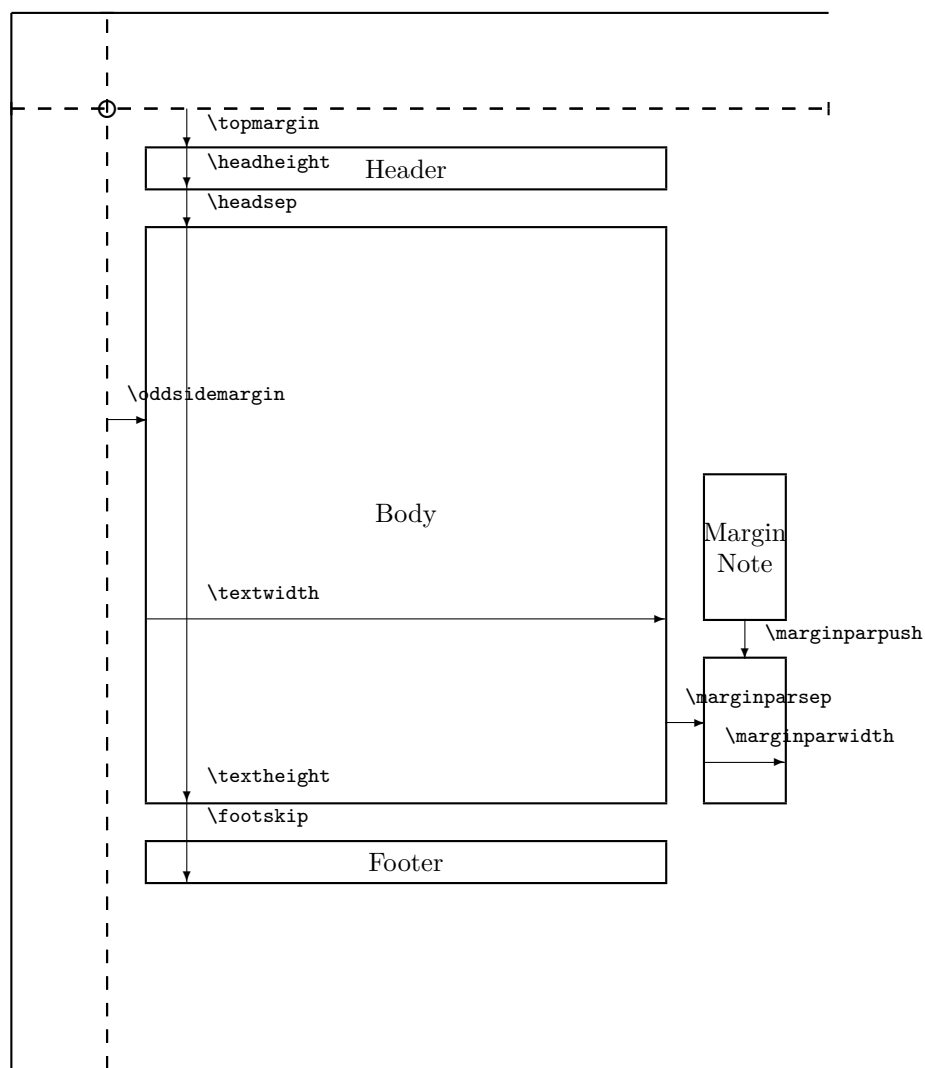


图 5.1: 控制页面的各种参数示意图

```
\usepackage{geometry}
\geometry{<geometry-settings>}
```

也可以将参数作为宏包的选项使用：

```
\usepackage[<geometry-settings>]{geometry}
```

其中 `<geometry-settings>` 多以 `<key>=<value>` 的形式组织。

比如，符合 Microsoft Word 习惯的页面设定是 A4 纸张，上下边距 1 英寸，左右边距 1.25 英寸，于是我们可以通过如下两种等效的方式之一设定页边距：

```
\usepackage[left=1.25in,right=1.25in,%
top=1in,bottom=1in]{geometry}
% or like this:
\usepackage[hmargin=1.25in,vmargin=1in]{geometry}
```

又比如，需要设定周围的边距一致为 1.25 英寸，可以用更简单的语法：

```
\usepackage[margin=1.25in]{geometry}
```

对于书籍等双面文档，习惯上奇数页右边、偶数页左边留出较多的页边距，而书脊一侧的奇数页左边、偶数页右边页边距较少。我们可以这样设定：

```
\usepackage[inner=1in,outer=1.25in]{geometry}
```

`geometry` 宏包本身也能够修改纸张大小、页眉页脚高度、边注宽度等等参数。更详细的用法不再赘述，感兴趣的用户可查阅 `geometry` 宏包的帮助文档。

5.3.2 页面内容的垂直对齐

L^AT_EX 默认将页面内容在垂直方向分散对齐。对于有大量图表的文档，许多时候想要做到排版匀称的页面很困难，垂直对齐会造成某些页面的垂直间距过宽。

L^AT_EX 还提供了另一种策略，即将页面内容向顶部对齐，给底部留出高度不一的空白。在导言区或者适合的位置使用以下命令开启顶部对齐的效果：

```
\raggedbottom
```

相反地，`\flushbottom` 命令用于设置页面分散对齐。

5.3.3 分栏

5.4 页眉页脚

5.4.1 基本的页眉页脚样式

L^AT_EX 中提供了命令 `\pagestyle` 来修改页眉页脚的样式：

```
\pagestyle{<page-style>}
```

另外一个命令只影响当页的页眉页脚样式：

```
\thispagestyle{<page-style>}
```

表 5.5: L^AT_EX 预定义的页眉页脚样式

<code>empty</code>	页眉页脚为空
<code>plain</code>	页眉为空, 页脚为页码。(article 和 report 文档类默认; book 文档类的每章第一页也为 plain 格式)
<code>headings</code>	页眉为章节标题和页码, 页脚为空。(book 文档类默认)
<code>myheadings</code>	页眉为页码及 <code>\markboth</code> 和 <code>\markright</code> 命令手动指定的内容, 页脚为空。

`<page-style>` 参数为样式的名称, 在 L^AT_EX 里预定义了四类样式, 见表 5.5。

其中 `headings` 的情况较为复杂:

- `twoside` 选项的 article 文档类, 偶数页页眉为页码和节标题, 奇数页页眉为小节标题和页码;
- `twoside` 选项的 book/report 文档类, 偶数页页眉为页码和章标题, 奇数页页眉为节标题和页码;
- `oneside` 选项的 article 文档类, 页眉为节标题和页码;
- `oneside` 选项的 book/report 文档类, 页眉为章标题页码页码。

5.4.2 手动更改页眉页脚的内容

对于 `headings` 或者 `myheadings` 样式, L^AT_EX 允许用户使用命令手动修改页眉上面的内容:

```
\markright{<right-mark>}
\markboth{<left-mark>}{<right-mark>}
```

其中 `<left-mark>` 和 `<right-mark>` 的内容分别预期出现在左页 (偶数页) 和右页 (奇数页)。

事实上 `\chapter`、`\section` 等命令内部也使用 `\markboth` 或者 `\markright` 写页眉。L^AT_EX 默认在写页眉的时候强制将英文字母变为大写。如果你不喜欢这样, 可以尝试以的代码 (相关命令的用法参照 8.1 节)⁶:

```
\renewcommand\chaptermark[1]{%
  \markboth{Chapter \thechapter\quad #1}{}}
\renewcommand\sectionmark[1]{%
  \markright{\thesection\quad #1}}
```

以上代码适用于 report/book 文档类。对于 article 文档类, 与两个页眉相关的命令分别为 `\sectionmark` 和 `\subsectionmark`。

5.4.3 fancyhdr 宏包

`fancyhdr` 宏包改善了页眉页脚样式的定义方式, 允许我们将内容自由安置在页眉和页脚的左、中、右三个位置, 还为页眉和页脚各加了一条横线。

⁶但是这不能改变页眉页脚的斜体样式, 斜体是定义在 `headings` 样式里的。如果不喜欢斜体, 办法之一是在 `\markboth` 等命令的参数里加上 `\normalfont` 以及想要的字体样式命令, 或直接尝试使用接下来介绍的 `fancyhdr` 宏包。

fancyhdr 自定义了样式名称 fancy。使用 fancyhdr 宏包定义页眉页脚之前,通常先用 `\pagestyle{fancy}` 调用这个样式。在 fancyhdr 中定义页眉页脚的命令为:

```
\fancyhead[<position>]{...}
\fancyfoot[<position>]{...}
```

其中 *<position>* 为 L (左) /C (中) /R (右) 以及与 O (奇数页) /E (偶数页) 字母的组合。

我们用一个示例说明 fancyhdr 的用法,这段代码可以用于导言区,它的效果是将章节标题放在和 headings 一致的位置,但使用加粗格式;页码都放在页脚正中;并修改 fancy 格式引入的横线宽度,“去掉”页脚的横线。更多的用法请读者参考 fancyhdr 宏包的帮助文档。

```
\usepackage{fancyhdr}
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{%
    \markboth{#1}{}}
\renewcommand{\sectionmark}[1]{%
    \markright{\thesection\ #1}}
% delete current header and footer
\fancyhf{}
% set our header and footer
\fancyfoot[C]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0pt} % remove footnote line
```

源代码 5.1: fancyhdr 宏包的使用方法示例。

第六章 特色工具和功能

本章介绍一些特色的 L^AT_EX 辅助功能。前两个功能 BibT_EX 和 makeindex 依靠一些辅助程序自动生成参考文献、索引等；之后的使用颜色、超链接等则令我们生成美观易用的电子文档。电子文档的特点在 beamer 里发挥到极致，它令 L^AT_EX 能够做出不逊于 PowerPoint 的用于幻灯演示的文档。

6.1 参考文献和 BibT_EX 工具

6.1.1 基本的参考文献和引用

L^AT_EX 提供的参考文献和引用方式比较原始，需要用户自行书写参考文献列表。不同学术论文对参考文献列表的格式要求不一样，自行书写是一件极其头疼的事情。相关的命令我们只作最简单的介绍。

L^AT_EX 提供了最基本的 \cite 命令用于在正文中引用参考文献：

`\cite{<citation>}`

\cite 带一个可选参数，为引用的编号之后加上额外的内容，如 \cite [page 22] 可能得到形如 [13, page 22] 这样的引用。

参考文献由 thebibliography 环境包裹，每条参考文献由 \bibitem 开头，其后是参考文献本身的内容：

`\bibitem[<item number>]{<citation>}`

其中 <citation> 是 \cite 用以引用的一个标签，类似交叉引用里的 \label。<item number> 自定义参考文献的序号，如果省略，则按自然排序给定序号。

thebibliography 环境带一个参数，用以设定 \cite 命令生成的引用编号的最大宽度，如 99 意味着不超过两位数字。通常设定为与参考文献的数目一致。

Part1-\cite{pa} has proposed that \ldots

```
\begin{thebibliography}{99}
\bibitem{pa} H.~Part1: \emph{German \TeX},
  TUGboat Volume~9, Issue~1 (1988)
\end{thebibliography}
```

6.1.2 BibT_EX 数据库

BibT_EX 是最为流行的参考文献数据组织格式之一。它的出现让我们摆脱手写参考文献条目的麻烦。我们还可以通过参考文献格式的支持，让同一份 BibT_EX 数据库生成不同格式的参考文献列表。我们首先简单介绍 BibT_EX 数据库，之后介绍如何将数据库利用在 L^AT_EX 中。

BibTeX 数据库以 `.bib` 作为扩展名，其内容是若干个文献条目，每个条目的格式为：

```
@<type>{<citation>,
  <key1> = {<value1>},
  <key2> = {<value2>},
  ...
}
```

其中 `<type>` 为参考文献的类型，如 `article` 为学术论文，`book` 为书籍，`incollection` 为论文集集中的一篇，等等。`<citation>` 为 `\cite` 命令所用的参考文献的标签。在 `<citation>` 之后为条目里的各个数据项，以 `<key> = {<value>}` 的形式组织。

我们在此简单列举学术论文里使用较多的 BibTeX 文献条目格式。所有条目格式参考 [CTAN://biblio/bibtex/base/btxdoc.pdf](http://biblio/bibtex/base/btxdoc.pdf)。

article 学术论文，必需数据项有 `author`, `title`, `journal`, `year`；可选数据项包括 `volume`, `issue`, `number`, `pages`, `doi` 等；

book 书籍，必需数据项有 `author/editor`, `title`, `publisher`, `year`；可选数据项包括 `volume/number`, `series`, `address` 等；

incollection 论文集集中的一篇，必需数据项有 `author`, `title`, `booktitle`, `publisher`, `year`；可选数据项包括 `editor`, `volume/number`, `chapter`, `pages`, `address` 等；

inbook 书中的一章，必需数据项有 `author/editor`, `title`, `chapter/pages`, `publisher`, `year`；可选数据项包括 `volume/number`, `series`, `address` 等。

多数时候，我们无需自己手写 BibTeX 文献条目。从 Google Scholar 或者期刊/数据库的网站上都能够导出 BibTeX 文献条目，老牌的文献管理软件 EndNote 也支持生成 BibTeX 格式的数据库。开源软件 JabRef 甚至支持 BibTeX 文献条目的导入、导出和管理。

6.1.3 使用 BibTeX 排版参考文献

现在我们来查看如何利用 BibTeX 数据库生成参考文献和引用。

第一步：我们当然需要一份 BibTeX 数据库，假设起名为 `books.bib`，和我们的 LaTeX 源代码一般位于同一个目录下。

第二步：在源代码中添加必要的命令。假设源代码的名称为 `demo.tex`。首先需要的是在导言区设定参考文献的格式：

```
\bibliographystyle{<bst-name>}
```

其中 `<bst-name>` 为格式文件的名称。BibTeX 提供了几个预定义的格式，如 `plain`, `unsrt`, `alpha` 等。一些学术论文的模板会提供自有的格式文件，以 `.bst` 作为扩展名，同样和 LaTeX 源代码放在同一个目录下。

其次就是在正文中引用参考文献了。BibTeX 程序在生成参考文献列表的时候，通常只列出你用 `\cite` 命令引用的那些。如果需要列出没有被引用的文献，则需要 `\nocite{<citation>}` 命令；而 `\nocite{*}` 则让所有未引用的参考文献都列出。

再次，在你需要列出参考文献的位置，使用 `\bibliography` 命令代替 `thebibliography` 环境：

```
\bibliography{<bib-name>}
```

```
@book{Lamport1994,
  title = {\LaTeX} A Document Preparation System,
  author = {Leslie Lamport},
  publisher = {Addison-Wesley},
  address = {Reading, Massachusetts},
  year = {1994},
  edition = {2nd}
}

@book{Mittelbach2004,
  title = {The \LaTeX Companion},
  author = {Frank Mittelbach and Michel Goossens and
    Johannes Braams and David Carlisle and Chris Rowley},
  publisher = {Addison-Wesley},
  address = {Reading, Massachusetts},
  year = {2004},
  edition = {2nd}
}
```

源代码 6.1: BibTeX 数据库示例 books.bib。

```
\documentclass{article}
\bibliographystyle{plain}
\begin{document}
\section{Some words}
Some excellent books, for example, \cite{Lamport1994}
and \cite{Mittelbach2004} \ldots

\bibliography{books}
\end{document}
```

源代码 6.2: 利用 books.bib 生成参考文献的源代码 demo.tex。

其中 $\langle bib-name \rangle$ 是 BibTeX 数据库的文件名，不要带 .bib 扩展名。

注意：\bibliographystyle 和 \bibliography 命令缺一不可，没有这两个命令，使用 BibTeX 生成参考文献列表的时候会报错。

第三步：写好了以上两个文件之后，我们就可以开始编译了。

1. 首先使用 latex 或 pdflatex 等命令编译 L^AT_EX 源代码 demo.tex；
2. 接下来用 bibtex 命令处理 demo.aux 文件记录的参考文献格式、引用条目等信息。bibtex 命令处理完后会生成 demo.bbl 文件，内容就是一个 thebibliography 环境；
3. 然后再使用**两遍** latex 或 pdflatex 等命令编译源代码，读入参考文献并正确生成引用。

整个过程使用的命令如下（可以略去扩展名）：

```
pdflatex demo
bibtex demo
pdflatex demo
pdflatex demo
```

我们当然也可以换成 xelatex 命令；如果使用 latex + dvipdfmx 命令，则 dvipdfmx 命令放在最后，相当于先后使用 latex, bibtex, latex, latex, dvipdfmx。

6.1.4 natbib 宏包

时下许多学术期刊比较喜欢使用人名——年份的引用方式，形如 (*Alice et al.*, 2005)。natbib 宏包提供了对这种“自然”引用方式的处理。

natbib 宏包在正文中支持两种引用方式：

$\backslash\text{citep}\{\langle citation \rangle\}$ $\backslash\text{citet}\{\langle citation \rangle\}$
--

它们分别生成形如 (*Alice et al.*, 2005) 和 *Alice et al.*(2005) 的人名——年份引用。

natbib 宏包同样也支持数字引用，并且支持将引用的序号压缩，例如：

```
\usepackage[numbers,sort&compress]{natbib}
```

用以上选项调用 natbib 宏包后，连续引用多篇文献时，会生成形如 [3-7] 的引用而不是 [3, 4, 5, 6, 7]。

natbib 宏包还有更多选项和用法，比如默认的引用是用小括号包裹的，可以为宏包添加 square 选项改为中括号；再比如 \citep 命令也支持可选参数，为引用前后都添加额外内容。这里不再赘述，请参考 natbib 宏包的帮助文档。

6.2 索引和 makeindex 工具

书籍和大文档通常用索引来归纳关键词，方便用户查阅。L^AT_EX 借助配套的 makeindex 程序完成对索引的排版。

表 6.1: 索引项的写法列表。

举例	索引项	备注
普通索引		
hello	hello, 1	普通索引
分级索引，以 ! 分隔，最多支持三级		
hello	hello, 1	一级索引
hello!Peter	Peter, 3	二级索引
hello!Peter!Jack	Jack, 3	三级索引
格式化索引，形式为 $\langle\alpha\rangle@{\langle\textit{format}\rangle}$		
$\langle\alpha\rangle$ 为纯字母，用来排序		
$\langle\textit{format}\rangle$ 为索引的格式，可以包括 L ^A T _E X 代码和公式		
Möbius@M{"obius	Möbius, 2	输出重音
alpha@\${\alpha}	α , 7	输出公式
bold@{\textbf{bold}}	bold , 12	输出粗体
页码范围		
morning (morning, 6-7	范围索引的开头
morning)		范围索引的结尾
格式化索引页码		
Jenny textbf	Jenny, 3	调用 \textbf 加粗页码
Joe see{Jenny}	Joe, <i>see</i> Jenny	调用 \see 生成特殊形式
Joe seealso{Jenny}	Joe, <i>see also</i> Jenny	调用 \seealso 生成特殊形式

6.2.1 使用 makeindex 工具的方法

要使用索引，须经过这么几个步骤（仍设源代码名为 `demo.tex`）：
第一步，在 L^AT_EX 源代码的导言区加载 `makeidx` 宏包，并使用 `\makeindex` 命令开启索引的收集：

```
\usepackage{makeidx}
\makeindex
```

第二步，在正文中需要索引的地方使用 `\index` 命令。`\index` 命令的参数写法详见下一小节；并在需要输出索引的地方（如源代码所有章节之后）使用 `\printindex` 命令。
第三步，用 `pdfLATEX` 等命令编译源代码 `demo.tex`，编译过程中产生索引记录文件 `demo.idx`。
第四步，用 `makeindex` 程序处理 `demo.idx`，生成用于排版的索引列表文件 `demo.ind`。
第五步，再次编译源代码 `demo.tex`，正确生成索引列表。

6.2.2 索引项的写法

添加索引项的命令为：

\index{ $\langle index\ entry\rangle$ }

其中 $\langle index\ entry\rangle$ 为索引项，写法由表 6.1 汇总。其中 `!`、`@` 和 `|` 为特殊符号，如果要向索引项直接输出这些符号，需要加前缀 `"`；而 `"` 需要输入两个引号 `""` 才能输出。
读者可以钻研一下一个较为复杂的，结合多级索引、索引格式、页码格式等的示例。但在自己使用时，仍然要遵循“简单的就是最好的”原则：

```

Test index.
\index{Test@\textsf{"Test}|(textbf}
\index{Test@\textsf{"Test"!sub@"|sub"||see{Test}}
\newpage
Test index.
\index{Test@\textsf{"Test}|)textbf}

```

6.3 使用颜色

L^AT_EX 原生不支持颜色，它依赖 dvipdfmx 或者 pdfL^AT_EX / X_gL^AT_EX 等命令输出带颜色的内容。为了使用颜色，须调用 color 宏包或者 xcolor 宏包。

6.3.1 颜色的表达方式

调用 color 宏包后，我们就可以用如下命令切换颜色：

```

\color[<color-mode>]{<code>}
\color{<color-name>}

```

颜色的表达方式有两种，其一是使用色彩模型和色彩代码，代码用 0 ~ 1 的数字代表成分的比例：

```

\large\sffamily
{\color[gray]{0.6}
  60\% 灰色} \
{\color[rgb]{0,1,1}
  青色}

```

60% 灰色
青色

color 宏包支持 rgb、cmk 和 gray 模型，xcolor 支持更多的模型如 hsb 等。

还可以直接使用名称代表颜色，前提是已经定义了颜色名称（没定义的话会报错）：









```

\large\sffamily
{\color{red} 红色} \
{\color{blue} 蓝色}

```

红色
蓝色

color 宏包仅定义了八种颜色名称：

black		red		green		blue	
white		cyan		magenta		yellow	

xcolor 宏包定义了 19 种颜色名称，并且还支持将颜色通过表达式混合或互补：

```

\large\sffamily
{\color{red!40} 40\% 红色}\
{\color{blue} 蓝色}
\color{blue!50!black} 蓝黑
\color{black} 黑色}\
{\color{-red} 红色的互补色}

```

40% 红色
蓝色 蓝黑 黑色
红色的互补色

我们还可以通过命令自定义颜色名称，注意这里的 <color-mode> 是必选参数：

```
\definecolor{<color-name>}{<color-mode>}{<code>}
```

如果调用 `color` 或 `xcolor` 宏包时使用 `dvipsnames` 选项，就有额外的 68 种颜色名称可用。`xcolor` 宏包还支持通过其它选项载入更多颜色名称。限于篇幅不展开介绍，详情请参考 `xcolor` 宏包的手册。

6.3.2 带颜色的文本和盒子

原始的 `\color` 命令类似于字体命令 `\bfseries`，它使之后排版的内容全部变成指定的颜色，所以直接使用时通常要加花括号分组。`color` / `xcolor` 宏包都定义了一些方便用户使用的带颜色元素。

输入带颜色的文本可以用类似 `\textbf` 的命令：

```
\textcolor[<color-mode>]{<code>}{<text>}
\textcolor{<color-name>}{<text>}
```

以下命令构造一个带背景色的盒子，`<material>` 为盒子中的内容：

```
\colorbox[<color-mode>]{<code>}{<material>}
\colorbox{<color-name>}{<material>}
```

以下命令构造一个带背景色和有色边框的盒子，`<fcode>` 或 `<fcolor-name>` 用于设置边框颜色：

```
\fcolorbox[<color-mode>]{<fcode>}{<code>}{<material>}
\fcolorbox{<fcolor-name>}{<color-name>}{<material>}
```

```
\sffamily
文字用\textcolor{red}{红色}强调\\
\colorbox[gray]{0.95}{浅灰色背景} \\
\fcolorbox{blue}{yellow}{%
  \textcolor{blue}{蓝色边框+文字，%
    白色背景}
}
```

文字用红色强调
浅灰色背景
蓝色边框 + 文字，白色背景

6.4 使用超链接

PDF 文档格式是现今最流行的电子文档格式，而电子文档的最实用的功能之一就是链接功能。 \LaTeX 中实现这一功能的 `hyperref` 宏包堪称 \LaTeX 里最强大也最复杂的宏包。

6.4.1 `hyperref` 宏包

`hyperref` 宏包涉及到的链接涉及到 \LaTeX 的每一个角落——目录、引用、脚注、索引、参考文献等等，凡是有可能需要链接其它地方的文档元素，几乎都被 `hyperref` 包装起来。但这也使得 `hyperref` 宏包与其它宏包的冲突机会大大增加，虽然宏包已经尽力解决各方面的兼容性，但仍不能面面俱到。为减少冲突的可能性，习惯上将 `hyperref` 宏包放在其它宏包之后调用。

6.4.2 超链接

6.4.3 PDF 书签

6.4.4 PDF 文档属性

6.5 beamer 幻灯片宏包

第七章 绘图功能

7.1 概述

7.2 `picture` 环境

7.3 交换图

7.4 `TikZ` 宏包

第八章 自定义 L^AT_EX 命令和功能

读到这一章之前，如果你确保掌握了前几章的知识并熟练运用，你已经能制作出内容和形式都相当丰富的文档了。但你可能还不会满足：我要如何制作一个简单但像样的毕业论文/书籍/简历模板，每次可以直接套用，而不是再在导言区写一堆代码？

本章的内容将有助于你实现这一个目标，让你能编写可重复利用的模块——宏包和文档类，并在其中自己定义命令和环境。不过作为入门手册，这些知识仍是不全面的。如果你不满足于此，需要参考更多资料。

8.1 自定义命令和环境

你也许已经意识到了，在本手册中介绍的所有命令都被包含在一个矩形框里，并且出现在本书最后的索引中。我并没有直接使用基础的 L^AT_EX 命令来实现这个效果，而是创建了一个宏包 (package)，并在其中定义了我所需要的命令和环境。现在我只需写成这样简单的形式：

```
\begin{command}  
\cmd{dum}  
\end{command}
```

\dum

在这个例子中，我使用了一个新的环境 `command`。这个环境负责在命令的周围画出一个矩形框。同时我还使用了一个命令：`\dum`，这个命令负责输出命令的名字，包括前面的反斜杠。

一旦我不再喜欢在一个矩形框中排版命令，我可以很容易地改变 `command` 环境的定义，来创建新的外观，而不是挨个修改每个画着矩形框的命令示例。

8.1.1 定义新命令

使用如下命令可以定义你自己的命令：

```
\newcommand{<name>}[<num>]{<definition>}
```

基本上，这个命令有两个参数，第一个 `<name>` 是你想要建立的命令的名称，第二个 `<definition>` 是命令的定义。方括号里的参数 `<num>` 是可选的，用于指定新命令所需的参数数目（最多 9 个）。如果不给出这个参数，默认就是 0，也就是新建的命令不要任何参数。

接下来的两个例子有助你的理解。第一个例子定义了一个新的命令：`\tnss`。这个命令是句子“The Not So Short Introduction to L^AT_EX 2_ε”的简写。如果你需要在文档中多次使用本书的名称，那么定义这个命令将是非常方便的。

```
\newcommand{\tnss}{The not
  so Short Introduction to
  \LaTeXe}
This is ‘‘\tnss’’ \ldots{}
‘‘\tnss’’
```

This is “The not so Short Introduction to L^AT_EX 2_ε” ... “The not so Short Introduction to L^AT_EX 2_ε”

下一个例子演示了如何定义一个接受一个参数的命令。在命令的定义中，标记 #1 将被你指定的参数所代替。如果你想使用多个参数，那么可以依次使用 #2、...、#9 等标记。

```
\newcommand{\txsit}[1]
{This is the \emph{#1} Short
  Introduction to \LaTeXe}
% in the document body:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

- This is the *not so* Short Introduction to L^AT_EX 2_ε
- This is the *very* Short Introduction to L^AT_EX 2_ε

L^AT_EX 不允许你新建一个与现有命令重名的命令。如果你确实需要这么做，有一个专门的命令用于处理这种情况：\renewcommand。它使用与命令 \newcommand 相同的语法。

在某些情况之下，你可能会希望使用 \providecommand 命令。它完成与 \newcommand 命令相同的工作。但如果命令已经存在，L^AT_EX 2_ε 将会忽略你的定义。

8.1.2 定义环境

与 \newcommand 命令类似，有一个命令用于定义新的环境。这个命令是 \newenvironment，它的语法如下所示：

```
\newenvironment{<name>}[<num>]{<before>}{<after>}
```

同样地，\newenvironment 命令有一个可选的参数。在 <before> 中的内容将在此环境包含的文本之前处理，而在 <after> 中的内容将在遇到 \end{<name>} 命令时处理。

下面的例子演示了 \newenvironment 命令的用法：

```
\newenvironment{king}
{\rule{1ex}{1ex}%
  \hspace{\stretch{1}}}
{\hspace{\stretch{1}}%
  \rule{1ex}{1ex}}

\begin{king}
My humble subjects \ldots
\end{king}
```

■ My humble subjects ... ■

参数 <num> 的使用方式与 \newcommand 命令相同。L^AT_EX 还同样保证你不会不小心新建重名的环境。如果你确实希望改变一个现有的环境，你可以使用命令 \renewenvironment，它使用与命令 \newenvironment 相同的语法。

8.2 编写自己的宏包和文档类

8.2.1 编写简单的宏包

如果你定义了很多新的环境和命令，你的文档的导言区将变得相当长，在这种情况下，好的方式是建立一个新的 L^AT_EX 宏包来存放所有你自己定义的命令和环境，然后在你的文档中使用 `\usepackage` 命令来调用自定义的宏包。

```
% Demo Package by Tobias Oetiker
\ProvidesPackage{demopack}
\newcommand{\tnss}{The not so Short Introduction
               to \LaTeXe}
\newcommand{\txsit}[1]{The \emph{#1} Short
                      Introduction to \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

源代码 8.1: 宏包的一个最简示例。

写一个宏包的基本工作就是将原本在你的文档导言区里很长的内容拷贝到另一个文件中，这个文件需要以 `.sty` 作扩展名。你还需要加入一个宏包专用的命令：

`\ProvidesPackage{<package name>}`

这个命令应该放在你的宏包的最前面，并且一定要注意：<package name> 需要和宏包的文件名一致。`\ProvidesPackage` 让 L^AT_EX 记录宏包的名称，从而在你尝试再次调用同一个宏包的时候忽略后面的引入¹。源代码 8.1 给出了一个小的宏包示例，其中包含了我们之前定义的一些命令。

8.2.2 在宏包中调用其它宏包

如果你想进一步把各种宏包的功能汇总到一个文件里，而不是在文档的导言区罗列一大堆宏包的话，L^AT_EX 允许你在自己编写的宏包中调用其它宏包，命令为 `\RequirePackage`，用法和 `\usepackage` 一致：

`\RequirePackage[<option1>,<option2>]{<package name>}`

8.2.3 编写自己的文档类

当你更进一步，需要编写自己的文档类，如论文模板等，问题就稍稍麻烦了一些。首先，自己的文档类以 `.cls` 作扩展名，开头使用 `\ProvidesClass` 命令：

`\ProvidesClass{<class name>}`

当然了，<class name> 也需要和文档类的文件名一致。

但是有了上述命令和和你之前学到的 `\newcommand` 等，还并不能完成一个文档类的编写，因为诸如 `\chapter`、`\section` 等等许多常用的命令都是在文档类中定义的。事实上，许多时候我们只需要像调用宏包那样调用一个基本的文档类，省去许多不必要的麻烦。在你的文档类中调用其它文档类的命令是 `\LoadClass`，用法和 `\documentclass` 十分相像：

`\LoadClass[<option1>,<option2>]{<package name>}`

¹但如果你以不同的选项多次引入宏包，则有可能会引起错误。

8.3 计数器

附录 A 安装 T_EX 发行版

A.1 T_EX 发行版简介

A.2 安装和更新宏包

A.3 L^AT_EX 编辑器简介

附录 B 排除错误、寻求帮助

B.1 常见 L^AT_EX 错误及原因

B.2 查看帮助文档

B.3 常见宏包简介

参考文献

- [1] Leslie Lamport. *L^AT_EX: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.
- [2] Donald E. Knuth. *The T_EXbook*, Volume A of *Computers and Typesetting*, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9.
- [3] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley. *The L^AT_EX Companion, (2nd Edition)*. Addison-Wesley, Reading, Massachusetts, 2004, ISBN 0-201-36299-6.
- [4] Michel Goossens, Sebastian Rahtz and Frank Mittelbach. *The L^AT_EX Graphics Companion*. Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-201-85469-4.
- [5] Each L^AT_EX installation should provide a so-called *L^AT_EX Local Guide*, which explains the things that are special to the local system. It should be contained in a file called `local.tex`. Unfortunately, some lazy sysops do not provide such a document. In this case, go and ask your local L^AT_EX guru for help.
- [6] L^AT_EX3 Project Team. *L^AT_EX 2_ε for authors*. Comes with the L^AT_EX 2_ε distribution as `usrguide.tex`.
- [7] L^AT_EX3 Project Team. *L^AT_EX 2_ε for Class and Package writers*. Comes with the L^AT_EX 2_ε distribution as `clsguide.tex`.
- [8] L^AT_EX3 Project Team. *L^AT_EX 2_ε Font selection*. Comes with the L^AT_EX 2_ε distribution as `fntguide.tex`.
- [9] D. P. Carlisle. *Packages in the ‘graphics’ bundle*. Comes with the ‘graphics’ bundle as `grfguide.tex`, available from the same source your L^AT_EX distribution came from.
- [10] Rainer Schöpf, Bernd Raichle, Chris Rowley. *A New Implementation of L^AT_EX’s verbatim Environments*. Comes with the ‘tools’ bundle as `verbatim.dtx`, available from the same source your L^AT_EX distribution came from.
- [11] Vladimir Volovich, Werner Lemberg and L^AT_EX3 Project Team. *Cyrillic languages support in L^AT_EX*. Comes with the L^AT_EX 2_ε distribution as `cyrguide.tex`.
- [12] Graham Williams. *The TeX Catalogue* is a very complete listing of many T_EX and L^AT_EX related packages. Available online from [CTAN://help/Catalogue/catalogue.html](http://help/Catalogue/catalogue.html)
- [13] Keith Reckdahl. *Using EPS Graphics in L^AT_EX 2_ε Documents*, which explains everything and much more than you ever wanted to know about EPS files and their use in L^AT_EX documents. Available online from [CTAN://info/epslatex.ps](http://info/epslatex.ps)

- [14] John D. Hobby. *A User's Manual for MetaPost*. Downloadable from <http://cm.bell-labs.com/who/hobby/>
- [15] Alan Hoenig. *T_EX Unbound*. Oxford University Press, 1998, ISBN 0-19-509685-1; 0-19-509686-X (pbk.)
- [16] Urs Oswald. *Graphics in L^AT_EX 2_ε*, containing some Java source files for generating arbitrary circles and ellipses within the `picture` environment, and *MetaPost - A Tutorial*. Both downloadable from <http://www.ursoswald.ch>
- [17] Stephen G. Hartke. *A Survey of Free Math Fonts for T_EX and L^AT_EX*, an introduction of combination of text and math fonts in L^AT_EX. Downloadable from CTAN://info/Free_Math_Font_Survey/survey.pdf
- [18] *The L^AT_EX font catalogue*, a font catalogue of L^AT_EX font packages. Available online from <http://www.tug.dk/FontCatalogue/>

索引

amsmath, 15
amssymb, 15
amsthm, 15
arev, 42

bold face, 40

ccfonts, 42
cmbright, 42
commands
 \arccos, 19
 \arcsin, 19
 \arctan, 19
 \arg, 19
 \cos, 19
 \cosh, 19
 \cot, 19
 \coth, 19
 \csc, 19
 \deg, 19
 \det, 19
 \dim, 19
 \emph, 40
 \exp, 19
 \gcd, 19
 \hom, 19
 \hspace, 44
 \inf, 19
 \ker, 19
 \lg, 19
 \lim, 19
 \liminf, 19
 \limsup, 19
 \linespread, 43
 \ln, 19
 \log, 19
 \max, 19
 \min, 19
 \newcommand, 61
 \newenvironment, 62
 \par, 43
 \Pr, 19
 \providecommand, 62
 \ProvidesClass, 63
 \ProvidesPackage, 63
 \renewcommand, 62
 \renewenvironment, 62
 \sec, 19
 \setCJKmainfont, 43
 \setCJKmonofont, 43
 \setCJKsansfont, 43
 \setmainfont, 41
 \setmonofont, 41
 \setsansfont, 41
 \sin, 19
 \sinh, 19
 \sup, 19
 \tan, 19
 \tanh, 19
 \tnss, 61
 \usepackage, 63
 \vspace, 45

droid, 42

emphasized, 40
euler, 42

fancyhdr, 48
fdsymbol, 42
font, 39
font size, 39
fontspec, 41
fourier, 42

- inconsolata, 42
- indentfirst, 44
- italic, 40
- Knuth, Donald E., 1
- Lamport, Leslie, 1
- libertine, 42
- lmodern, 42
- mathabx, 42
- mathdesign, 42
- mathpazo, 42
- mathptmx, 42
- Mittelbach, Frank, 1
- MnSymbol, 42
- newpxmath, 42
- newpxtext, 42
- newtxmath, 42
- newtxtext, 42
- normal font, 40
- package, 61
- packages
 - amsfonts, 15
 - amsmath, 15
 - amssymb, 15
 - amsthm, 15
 - arev, 42
 - ccfonts, 42
 - cmbright, 42
 - droid, 42
 - euler, 42
 - fancyhdr, 48
 - fdsymbol, 42
 - fontspec, 41
 - fourier, 42
 - inconsolata, 42
 - indentfirst, 44
 - libertine, 42
 - lmodern, 42
 - mathabx, 42
 - mathdesign, 42
 - mathpazo, 42
 - mathptmx, 42
 - MnSymbol, 42
 - newpxmath, 42
 - newpxtext, 42
 - newtxmath, 42
 - newtxtext, 42
 - pxfonts, 42
 - sourcecodepro, 42
 - sourcesanspro, 42
 - txfonts, 42
 - xeCJK, 41
- pxfonts, 42
- roman, 40
- sans serif, 40
- slanted, 40
- Small Caps, 40
- sourcecodepro, 42
- sourcesanspro, 42
- txfonts, 42
- typewriter, 40
- upright, 40
- xeCJK, 41