

L^AT_EX 中文简明教程

A Brief Chinese Tutorial on L^AT_EX

编程爱好者协会

2020/05/04

本文以 CC BY-NC-SA 4.0 协议随源码发布





目录

I 前言：关于 L ^A T _E X	2
1.1 认识 L ^A T _E X 和 T _E X	2
1.2 L ^A T _E X 能做到的事情，以及选择 L ^A T _E X 的理由	2
1.3 L ^A T _E X 的开发工具链及资料指南	3
II 开始使用 L ^A T _E X	4
2.1 你好，L ^A T _E X	4
2.2 支持中文的 L ^A T _E X 文档	4
2.3 初学者会用到的包	5
2.3.1 语言支持包	5
2.3.2 理工科类宏包	5
2.3.3 图表和图片宏包	7
III L ^A T _E X 常用功能	10
3.1 行和章节层次	10
3.1.1 基本操作	10
3.1.2 改变标题格式	10
IV 附录	12
A 常见问题（FAQ）	12



第 I 部分 前言：关于 L^AT_EX

本文使用 L^AT_EX 写就，以本文自身使用的代码为例，展示和讲解 L^AT_EX 基本语法和在计算机工程领域撰写文章的常见用途用法。本文应当连同源码和一份附带的 img 图片资源文件夹发布，若没有，请向发布者索要，但必须在尊重协议的基础上。

1.1. 认识 L^AT_EX 和 T_EX

L^AT_EX 和 T_EX 都是用于排版的语言。T_EX 及其发展出的语言都是指令驱动的，用于生成 pdf 文件。这些语言的基础都在于宏控制序列，开发者可以通过编写代码来进行精确、高度自动化同时高度可定制的排版任务。无论是字体字号还是段落格式，随笔文集还是论文，都可以用 L^AT_EX 中高度集成化的指令设计解决。此外，L^AT_EX 可以自动为您的各种题注进行编号，并方便地在后文中引用，这样您就不需要在编号变动后一个个手动改动引用了，这也意味着出错的几率大大降低。

T_EX 的出现很早，可追溯至 HTML 标记语言出现之前。L^AT_EX 继承和发展了 T_EX，可以简单地认为 T_EX 相对“朴实无华”而 L^AT_EX 内置了众多常用宏。对本文读者而言只需要知道一件事：L^AT_EX 相比一般的 T_EX 更加易用，适合快速上手。本文接下来只讨论 L^AT_EX 而不再讨论 T_EX。

1.2. L^AT_EX 能做到的事情，以及选择 L^AT_EX 的理由

L^AT_EX 在排版工作上几乎无所不能。使用社区提供的宏包，它能完成几乎任何常见排版需求。

对选择 L^AT_EX 顾虑最大的人群恐怕是 Microsoft Office Word 的用户。诚然 Markdown 和 HTML&CSS 都可用于排版，但是 Markdown 和 HTML&CSS 的业务还是主要集中于网络界面。Markdown 也可以做文档排版，但是就笔者的经验而言，Markdown 使用上虽然简易，但是很多自定义功能实现反而比 L^AT_EX 复杂一些。而在离线程序文档中，Markdown 写就的 README 和 L^AT_EX 就更没有什么比较意义了。所以，笔者并不认为 Markdown 和 L^AT_EX 有太多的冲突。也有读者可能会使用方正和 Adobe 的产品，但那些产品通常是收费的（当然，Microsoft Office 也是收费的），而且主要面向专业的排版厂商，个人开发者使用那种体量的企业级产品实在是没有必要。我们这里主要阐述 L^AT_EX 相比 Microsoft Office Word 的优劣。

相比于 Microsoft Office Word 的 GUI 为核心的设计，在 L^AT_EX 下您一般不需要操心把图片放在哪，L^AT_EX 会尝试自己解决这件事。如果对它的工作不满意，也可以通过参数自行指定。相比 Word 下需要点来点去拖来拖去，L^AT_EX 只需要您已经写好的代码就能得到最终结果，要修改效果也只需要简单地改写代码，不需要打开额外的属性页面。

L^AT_EX 另一大亮点是经常集成包管理系统，只需要引入几个包敲几行代码就能生成漂亮的数学公式或是化学方程式，物理论文或是生物论文自然也可以。如果需要插图，L^AT_EX 也有一些常见的绘图软件，只要需要的示意图不是太复杂，您就能用 L^AT_EX 把它画出来。相比于传统讲义设计中繁琐的示意图绘制、公式插入和特殊符号插入，您在 L^AT_EX 下只需要几行代码。当然，这些代码需要用户自行编写。

传统计算机工程领域讲义编写中，代码高亮常常是通过截图或第三方工具实现的。这一切在 L^AT_EX 社区提供的宏包面前都黯然失色。在 L^AT_EX 下，您所需要的只是引入包，设置高亮格式，把代码复制进来。如果愿意，您甚至可以自己实现一门语言及其在 L^AT_EX 下的高亮。哪怕拥有 yacc 工



具的用户也需要自己编写代码才能实现这样的自定义语言高亮，但在 L^AT_EX 下，用户只需要使用提供好的工具，甚至没有必要真的把这门语言实现出来。

然而，L^AT_EX 是基于 CLI 界面的工具（当然也有 IDE，但底层仍然是 CLI），对于使用惯了 Windows 提供的图形化操作的用户而言，L^AT_EX 可能不是太美好。而且，L^AT_EX 并不是所见即所得的工具，所以您需要不断改写和编译您的源码才能得到期望的文档。但是，如果您真的是本文所面向的计算机领域工程师，我有理由相信您是乐于折腾手头的工具和设备的 geek——否则恐怕不会选择这个行业。如果果真如此的话，有理由相信 L^AT_EX 能成为您今后的最佳伴侣。

1.3. L^AT_EX 的开发工具链及资料指南

中文社区最常见的 L^AT_EX 开发工具是 C_T_EX，但笔者在使用 C_T_EX 的过程中遇到了不小的麻烦，主要是附带的 MiK_T_EX 版本太旧，无法用于更新宏包。笔者的建议是使用官网的 MiK_T_EX 进行 L^AT_EX 的编辑。如果您愿意，也可以安装不带 MiK_T_EX 的 C_T_EX 包，再单独安装 MiK_T_EX，这样可以有效避免宏包无法更新的问题。

MiK_T_EX 和 C_T_EX 本质上是提供了开发环境，实际编译工作是附带的工具链来做的。比较常见的工具链有 pdfL^AT_EX、X_YL^AT_EX、X_YL_T_EX 等，笔者建议中文用户使用 X_YL^AT_EX。本文就是使用 X_YL^AT_EX 编写的。

您可以从清华大学开源软件站¹获取 C_T_EX 套装，而德语 T_EX 用户组维护的网站 T_EXdoc Online²处可以获取 L^AT_EX 常见宏包的文档。此外，stackexchange 也有一个专门的 T_EX 论坛³，可以在那里查询到常见问题的解决方案，还有一个称为 StackOvernet 的论坛也有大量技术资料。Comprehensive T_EX Archive Network⁴同样提供大量可供查阅的文档，该网站也能下载到各种宏包。L^AT_EX 论坛⁵也有一些常见问题的解答。MiK_T_EX 则可以从 MiK_T_EX 官网⁶获得。

L^AT_EX 有相当一部分是社区维护的，除非您能联系到包的作者本人，否则不太可能获取客服技术支持什么的。L^AT_EX 宏包因此会出现各种兼容性问题，请注意自行查阅材料确定哪些包可以混用，哪些不可以。此外，MiK_T_EX 用户应当了解，安装到计算机上的 MiK_T_EX 已经附带了包管理器，可以用于从网络上下载包并自动安装。

¹<https://mirrors.tuna.tsinghua.edu.cn/>

²<http://texdoc.net/>

³<https://tex.stackexchange.com/>

⁴<http://ctan.math.illinois.edu>

⁵<https://latex.org/forum/>

⁶<https://miktex.org/>




第 II 部分 开始使用 L^AT_EX

2.1. 你好，L^AT_EX

L^AT_EX 的主要构成十分简单，第一行声明基本文档类型（`\documentclass`），包含宏包（非必要，但没有宏包基本上做不了任何事情），前导区（preamble）声明必要的东西，最后开始写文档体。

```
1 \documentclass{article}
2
3 % \begin 表示开始环境，与\end 配对
4 % document 表示文档体环境
5 % 百分号开始的内容一般是注释，但也有例外
6 \begin{document}
7 Hello \LaTeX!
8 \end{document}
```

代码 2.1: Hello L^AT_EX

这就是一个 L^AT_EX 的 Hello world 文档（不过输出的是 Hello L^AT_EX 而不是 Hello world），在 TeXworks 界面上按下排版键 （默认快捷键 `Ctrl+T`）即可看到编译结果。该例子中的源码只输出一页 pdf，包含一行 Hello world 字样。它没有引用任何包。从这个例子中可见，L^AT_EX 的显示内容是直接写在文档里面就好的，由控制序列（也即各种指令，控制序列是它的正式名称）指定格式。

上述例子还无法输出中文。如果简单将 Hello world 文本更换为中文，你会发现，它什么也不会输出；编译器没有正确辨认中文字符。

2.2. 支持中文的 L^AT_EX 文档

给文档加一个包即可正确向 pdf 文件写入中文。以下是 X_YL^AT_EX 下的做法：

```
1 \documentclass{article}
2 % 有了这个包，就能正确输出中文了
3 \usepackage{ctex}
4
5 \begin{document}
6
7 你好，\LaTeX!
8 \end{document}
```

代码 2.2: 你好，中文 L^AT_EX- X_YL^AT_EX

使用 pdfL^AT_EX，则需要使用另一个方案：

```
1 \documentclass{article}
2 \usepackage{CJK}
3
```



```
4 \begin{document}
5 \begin{CJK}{UTF8}{song}
6
7 你好，\LaTeX!
8 \end{CJK}
9 \end{document}
```

代码 2.3: 你好，中文 L^AT_EX- pdfL^AT_EX

以上代码即是令 L^AT_EX 支持中文的使用方法¹。

2.3. 初学者会用到的包

L^AT_EX 实际上是以宏²为基础的语言，所以严格而言，“包”在这里应当叫做宏包。宏以\def 命令定义，此外，\newenvironment、\newcommand 和\renewcommand 分别可以定义新环境、新宏和修改特定宏的作用。作为一篇简明教程，这里我们只进行“脚本小子”式的教学，讲述常见几个宏包的使用方法，而略去关于宏本身的内容。

2.3.1. 语言支持包

对于中文用户，最常见的应该是 CJK 包和 ctex 包，这两个是中文支持包。此外，还有 greektext 希腊语包等其他语言支持。值得一提的是，有些院校要求子目录使用汉字编号，这可以用 zhnumber 包来完成，后面的章节会详细说明。还有一些包提供更多语言学特性，例如生成音标用的 tipa 包 (IPA 就是国际音标的缩写)。

用 tipa 包输出 Hello L^AT_EX 的音标为 [hə'ləʊ][leɪtɛk]³。

2.3.2. 理工科类宏包

主要是 amsthm 包，它提供数学环境和符号。常见的还有 extarrows 包，它提供特殊的箭头，允许在长箭头上附加文字，可用于书写无机化学方程式或核物理反应方程式。chemfig 和 mhchem 则常用于书写有机化学方程式，SIunits 包提供标准物理学单位。对于计算机行业工作者，最常见的功能大概是引用代码，这经常由 listings 包来完成。还有一个常见的叫做 algorithm2e 的宏包，用于生成伪代码。

下面以一些常见用途来举例子。

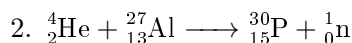
1.
$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t} dt$$

这是傅里叶变换的基本公式。你今天能用上电子产品不仅要感谢麦克斯韦等基础电磁理论的奠基人，还要感谢这个你也许看不懂的东西。不过如果你是学电子信息的，那就一定要弄懂这个东西，还要学好高等数学和复变函数。

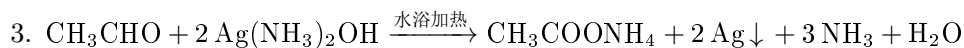
¹如果您正在比对本源码阅读该份 pdf 文件，您可能已经注意到了：上述代码片中的 lstlisting 环境中都加入了一行奇怪的代码，即 (@*\textcolor{bgcolor_gray}{ }*) 而且呈现奇怪的缩进。这里加入一个没有在 pdf 文件中显示出来的控制序列是为了避免错误的换行。总之，读者目前只需要知道一件事：这个控制序列是为了修正 listings 包在引用代码片和中文的不兼容引发的样式问题而加入的，而不是编写中文 L^AT_EX 文档所必要的控制序列。后文会简单讲解该控制序列的含义。至于缩进……L^AT_EX 是一个不在乎缩进的语言，有些时候空行也无关紧要（但更多的时候，空行有特别意义），这里只好先牺牲一点样式效果，避免更大的样式问题。

²宏的概念可以理解为直接原地展开宏内容并执行操作。有程序设计基础的读者可以联想其他语言的宏概念来理解它。提到宏，我们希望各位读者了解一件事：网络上有些人鼓吹宏是洪水猛兽应当尽量避免，这种观念是相当错误的。宏有利于大量减少代码量，不过相比于其他代码生成技术，宏有个显著缺点：可读性很差。我们建议工程师编写宏时辅以充足的文档说明。

³实际上 L^AT_EX 不仅这一种读法，详见附录 FAQ。

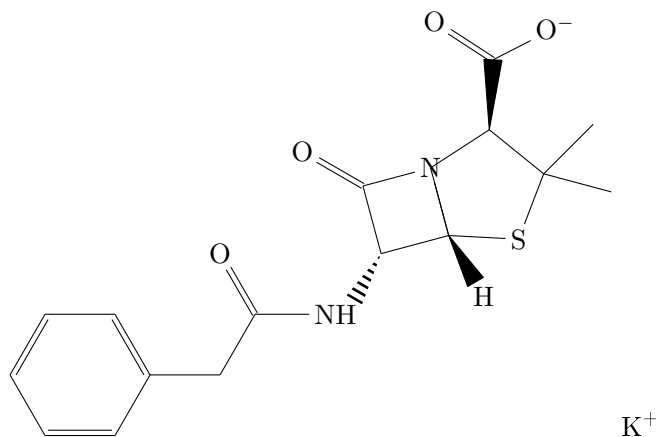


这是一个高中常见的 α 粒子轰击铝的核物理方程式。使用 α 粒子轰击铝箔后产生不稳定的 ${}^{30}_{15}\text{P}$ ，这种放射性核素很快就会继续衰变。 ${}^{30}_{15}\text{P}$ 是最早从实验室制造出来的人工放射性核素之一。



这个是银镜反应的化学方程式，我们在这里用它展示有特殊反应条件的方程式如何用 L^AT_EX 书写。

4. chemfig 绘制的结构式：



这种物质叫做青霉素钾，常用于制药。选取这个结构式来绘制是因为它具有许多有机化学结构式中会出现的特性，例如环、环嵌套以及各种分支。说真的，如果你只是用 L^AT_EX 来做化学作业，那我建议你直接导入插图，使用 chemfig 绘制如此简单的一个结构式就花费了我不少时间来查阅资料。

5. algorithm2e 生成的伪代码：

输入：待排序数组 $A[n]$ ，要求其类型强有序

输出：排好序的数组，仍存放在 $A[n]$ 中

```

1 Function BUBBLE-SORT( $A[n]$ )
2 begin
3   for  $i \leftarrow 1$  to  $n$  do
4     for  $j \leftarrow 1$  to  $n - i$  do
5       if  $A[j] > A[j + 1]$  then
6         SWAP( $A[j]$ ,  $A[j + 1]$ )
7       end
8     end
9   end
10 end

```

这是一段冒泡排序的伪代码（pseudo-code），仅仅是对算法的描述，不能被实际编译运行。冒泡排序是一个相当经典和简单的排序算法，它每一趟循环把前/后若干个值中最大/小者放在最后/前（取决于你要怎样的排序结果，以及如何实现它），所以很显然，每一趟排序后必然已经有至少一个值被放在了它最终该在的位置上。《算法导论》中具体讨论了这个算法。

6. listings 提供的代码片功能：



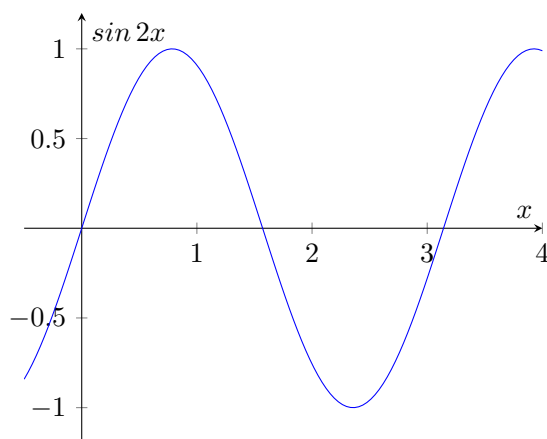
```

1 // A simple C++ example
2 #include<iostream>
3 using std::cout;
4 using std::endl;
5
6 int main() {
7     cout << "Hello world" << endl;
8     return 0;
9 }

```

这是一个简单的 C++ 程序，作用就是输出一行 Hello world。一个常见的误解是“std::cin 一定比 scanf 慢”，其实并不一定如此；SysTutorials 上一个题为“C++ 的 cin 真的比 scanf 慢很多吗”的主题讨论了这一点。

7. 函数图象：



这个例子展示了如何在 L^AT_EX 下用 pgfplots 绘制一幅笛卡尔坐标系下的函数图象。除此之外，pgf 包还有许多其他绘图功能。

2.3.3. 图表和图片宏包

caption 包和 subcaption 包用于题注。包含图片时，最常用的包是 graphicx。multirow 包用于进行单元格合并，TikZ 和 pgf 用于绘制图样。制表时经常配合 array 包制定样式。特别地，circuitikz 包主要用于绘制电路图。

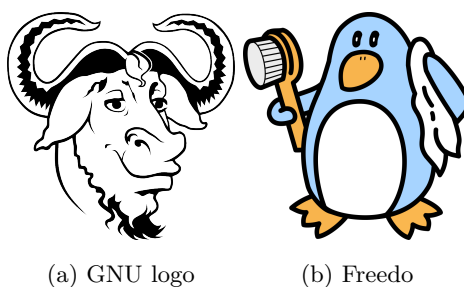


图 2.1: 两个 logos

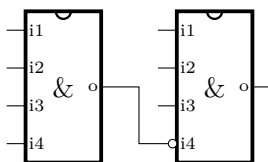


这个例子中,L^AT_EX 为这两张图片分别分配了题注,GNU 的 logo 得到编号 2.1a,小企鹅 Freedo⁴得到编号 2.1b。两张图还有共同题注,编号为 2.1。Aurélio A. Heckert 提供了图 2.1a, Rubén Rodríguez Pérez 等人提供了图 2.1b⁵, 这两张图都来自GNU 画廊⁶。

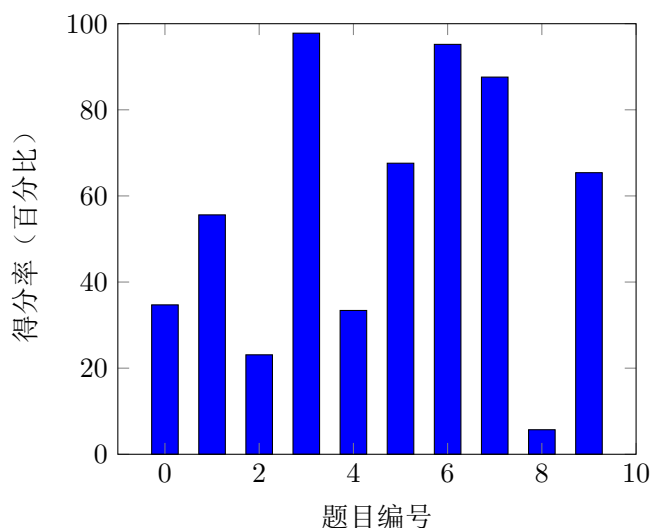
(0,0)	row 0		
column 0	center		

表 2.1: 一张随意画的 L^AT_EX 表格

这个表格例子展示了 L^AT_EX 中复杂表格的绘制方法。L^AT_EX 下的原生表格实际上做不到这点,但是我们有支持自定义表格的宏包。一般而言, Microsoft Office Word 能做到的表格功能, L^AT_EX 都有对应实现。



这是一个使用 circuitTikZ 包绘制的电路图,包含两个连在一起的四路与门,其中上一级与门的输出反相后连接到下一级与门的第四个输入(注意后一级与门的 i4 引脚上的圆圈标记)。这个芯片形式的与门部件是使用 pgf 底层命令定义的,并在 circuitTikZ 包层面调用封装好的部件。可以在本文随附源码的 contents/assets/circuitikz/and4b.tex 看到这个部件的详细定义,包括绘图细节和引脚排布。



⁴Freedo 是GNU Linux-libre (一个操作系统内核)的吉祥物。值得一提的是, Linux 最广为人知的吉祥物也是一只企鹅,名为 Tux。然而,许多 Linux 发行版如今选择其他动物作为自身的吉祥物。

⁵这张图是在<https://svgtopng.com/>进行了 SVG 格式到 PNG 格式的转码后得到的。

⁶GNU 是一个致力于发扬自由软件精神的组织,知名的 GPL 协议就来自于这个组织。作为最知名的开源支持者之一, GNU 经常以相当直白的方式表达对专有软件的不满。包括 CIA 和亚马逊在内的组织和企业都被GNU 操作系统官网挂起来批评过。其他一些开源支持者,例如 FFmpeg 的开发者,还列出了一个“耻辱柱”,上面列出了未遵守开源协议的软件清单;许多大厂也没能逃过被钉上 FFmpeg 耻辱柱的命运。不仅如此,许多基本开源的软件(例如某些官方包管理服务器使用了专有软件的操作系统)因为被认为“不够开源”而被各类自由软件组织开除自由软件籍。



下文中将会逐步讲解 L^AT_EX 内置功能和宏包中的常用功能。

注意 有许多宏包中的工具需要编译两次才能得到正确结果。所以，遇到编译没出错但效果样式有问题的情况先别着急，试试再编译一次。极少数情况下，这也有助于修正编译错误。发布前也要编译两次再进行审阅。此外，有时候宏包包含顺序也会引发问题，出现宏包内部错误时不妨尝试调整宏包的包含顺序。



第 III 部分 LaTeX 常用功能

3.1. 行和章节层次

3.1.1. 基本操作

很容易发现，在源码下换行后，编译出的 pdf 文件对应位置往往并没有还行（这就是为什么本文源码有那么多空行）。LaTeX 下最常见的两种换行方式分别是使用换行符号`\`，以及额外空一行。这两种方式都可以用于换行。本文中多数地方使用源码中空行来换行，但细心的读者也许已经发现，本文源码中 document 环境的 title 就用了`\`来换行。

```
\title{\bf \LaTeX 简明教程 \ A Brief Tutorial on \LaTeX}
```

使用`\`换行还有一个技巧，如果附带一对方括号，方括号内部写入一个长度，那么当前换行将会产生额外的空白。本文附录 A 中的行间额外空白就是通过`\[5mm]`实现的。

此外，也有时使用`\newline`进行换行。

LaTeX 使用的层次结构可以十分多样。不同文档类型（以`\documentclass`声明类型）常见的搭配也不同，比如 article 类型经常搭配 part、section 等结构类型而不是 chapter，但无论是哪一个，其本质是大同小异的。LaTeX 下任何段落层次结构都以类似方式声明，即“`\结构类型名称 {标题}`”的形式。

常见的结构类型有 part、section、subsection、subsubsection。使用`\section{标题}`就可以生成一个以“标题”为标题的小节。一般而言，标题会附带有一个编号。

3.1.2. 改变标题格式

本文介绍的改变标题格式的方法以 titlesec 的`\titleformat`为例，这是最常见的方案。

考虑到引用位置的需求，各级标题中最重要的部分就是编号。这里我们先明确计数器的概念。LaTeX 中计数器一般指的是 part、section 等默认计数器（你也可以生成自己的计数器，但本文不讨论这点），它们表示的就是对应结构的当前计数值。例如，本小节当前的计数器（subsection）值为 3.1.2。你也可以自行修改计数器的值，但本文也不讨论这点。

LaTeX 下，很多计数器具有一个以 the 为前缀的控制序列，用于控制编号输出样式。例如，`\thepart`表示当前 part 序号。本小节的位置可以用`\thesubsection`表示¹。

宏	在该文档本位置的值
<code>\thepart</code>	III
<code>\thesection</code>	3.1
<code>\thesubsection</code>	3.1.2

这里特别强调一点：计数器名称不带反斜杠（`\`），但是对应的宏是有的。LaTeX 的一切宏都以反斜杠开头以标识出它是个宏。除了特定宏指令参数外，part 都不会被当作计数器名称，而是普通的纯文本（plain text）。

计时器对应宏的显示样式可以用`\renewcommand{样式控制序列}`来改变。如果你希望使用罗马数字作为 section 序号，那么可以这样做：

¹正在对照源码的读者可能注意到了上文使用了`\ref`。`\ref`是用于全文引用某位置的语法，但是`\thesubsection`永远只能指示当前位置。这就是二者的区别。



```
\renewcommand{\Roman{section}}
```



第 IV 部分 附录

附录 A 常见问题 (FAQ)

问 L^AT_EX 到底如何发音?

答 这要追溯到它的词源。T_EX 一词来源于 $\tau\epsilon\chi$, 这是一个希腊语中的词根, 与科学和艺术有关, 发音类似 technology 中的 tech 音节 (事实上 tech 这个词根也与这一希腊语词根有关), 可读作 [tek] 或 [tex] (注意不能读作 [teks])。前缀 la-则发 [leɪ] 或 [laɪ]。合起来, 则 L^AT_EX 的发音接近于 lay-tek 或 lay-tech。

问 为什么我无法像本文源码中那样使用 \addbs 等宏?

答 源码中前导区内以 \newcommand 声明的宏都不是自带宏。如果您也希望使用类似宏, 需要自己加入自己的文档中。本文中未在前导区以 \newcommand 声明的宏都是文档类型内置宏或来自宏包, 如果是其他宏无法正确编译, 那很可能是宏包安装不正确、文档类型不匹配或宏包引用不正确。

问 如何安装宏包?

答 不同的 L^AT_EX 发行版的具体操作不一样, 请查阅您所使用发行版的文档或咨询技术人员。本文使用的 MiK_TE_X 则附带包管理器, 遇到未安装的包时可能会自动安装它们或者向您索要许可, 取决于您如何安装 MiK_TE_X。

问 我已经确认包和文档类型安装无误, 为何仍然无法编译这份源码?

答 本文似乎无法在 pdfL^AT_EX 下通过编译, 无法编译的读者请尝试切换到 X_YL^AT_EX 下进行编译。不过, 如果字符集不是 UTF-8, 也可能发生错误。如果您的 PC 上包不齐全, 也是一样会发生错误的。

问 为什么生成文档没有乱码, 源码却无法正确显示?

答 本文采用 UTF-8 编码, 笔者建议读者们也尽可能采取 UTF-8 编码。Windows 平台所广泛采用的 GBK2312 编码实际上并不是很常见。如果出现了用自己的软件打开 pdf 文件没有乱码, 打开源码却是乱码, 那么很有可能是编码问题, 使用专门工具将源码从 UTF-8 转到你的平台所使用的编码就可以了。

问 为什么我的 L^AT_EX 报告错误“无法使用 pdfL^AT_EX 编译”?

答 有些宏包不提供对 pdfL^AT_EX 的支持, 所以会人为引入这个错误。强行使用 pdfL^AT_EX 编译只会让情况更糟, 我们建议您尝试改用 X_YL^AT_EX 并尽可能避免使用和 X_YL^AT_EX 不兼容的宏包。

问 源码原本应当是引用的部分, 为何出现问号?

答 \ref 引用的部分没有对应的 \label 就会引发这个现象。检查你的 include 结构关系是否正确, 有无遗漏, 以及引用时有没有出现 typo。



问 本文中的脚注编号样式为什么出现了临时变换？统一成带方括号的形式不是更好吗？

答 国际上的惯例如此，可以认为是约定俗成的规矩。英文资料中，主要是星号或是不带任何附加样式的数字上标比较常见；汉语资料中则是带圈数字等比较常见。一言以蔽之，用加方括号上标作为脚注编号样式是比较少见的。这种样式主要用于参考文献的标号，以及类似的尾注样式。请注意：脚注和尾注完全不是同一种东西。

问 我在尝试编译这份文档的源码，为什么我的编译器报告找不到图片？

答 该文档原文确实包含了一些图片，放在与 L^AT_EX 源码同一目录下的 img 文件夹。如果您编译该文档的位置没有 img 文件夹或 img 文件夹内缺乏应有的文件，就会出现这个问题。

问 我在尝试使用 CJK 搭配 pdfL^AT_EX 输出中文页眉并在前导区设置了标题格式，为什么没有效果？

答 如果您确定要使用 pdfL^AT_EX，请在全文第一次使用中文之前进行 CJK 环境声明。如果前导区也有中文输出，那么只需要在前导区第一次出现中文字符之前声明一个空的 CJK 环境即可。

问 我的编译器报告“无法打开文件”，这是为什么？

答 有两种可能，其一是没有文件写入权限，其二是文件已被占用打开。前者可以通过文件权限标志来判断，确定不是没有写入权限的话就一定是另一个进程已经以占用方式打开了该文件，只要把占用者关掉就可以了。通常而言，占用者主要是各类阅读器，比如 Adobe Reader。

问 我的 L^AT_EX 编译器报告的错误所在行数比我的文档总行数还多，是为什么？

答 很可能是宏包内部问题，尝试调换宏包包含顺序，或者再编译一次，有时可以解决问题。如果还不行，有可能是宏包彼此之间的兼容性问题，也可能是宏包和 L^AT_EX 发行版不兼容。遇到兼容性问题是很难自己解决的，请咨询宏包发布者。

问 L^AT_EX 经常报出晦涩难懂的错误报告，是我的问题吗？

答 一般而言，出错是用户的问题，但晦涩难懂的错误报告不是。L^AT_EX 是基于宏的语言，相信计算机工程师们都知道一件事情，就是但凡有什么东西跟宏沾上了关系，它的底层原理就会变得极度复杂。L^AT_EX 形式上的简便易用正是建立在底层纷繁复杂的宏上面的，这意味着一旦您写的 L^AT_EX 文档出现语法问题，L^AT_EX 会在宏展开到底层时才能真正发现错误，正是这点导致了 L^AT_EX 的编译报错信息往往极其晦涩难懂。如果您无法接受这种违反人类习惯的报错信息，您可以去尝试阅读和书写 C++ 的宏和模板，尤其是 STL 代码和 Boost 库代码。那样您就会明白，L^AT_EX 报错的晦涩程度在计算机工程领域真的不值一提（笑）。当然这不是在批评 C++，为了让最终用户得到方便，上层开发者不得不写很多连同代码和错误报告（如果最终用户的使用方式有误的话）都极其晦涩难懂的代码，这就是一门语言为了得到通用性和易用性不得不付出的代价，也就是极其难以读懂的底层错误报告。其实耐着性子读完 L^AT_EX 发行版给出的错误报告的话，揪出实际出错位置还是很容易的。说句题外话，C++ 不同编译器提供的错误报告可读性也不同，受不了 GCC 的 yacc 技术提供的报告的话不妨尝试 Clang/LLVM。

问 L^AT_EX 报告了大量警告，有没有必要一个一个消除掉？

答 我们建议您尽可能避免任何警告。然而作为一个排版软件，仅仅是字体问题就足够 L^AT_EX 报告



上百个警告。只要您对最终输出的效果还算满意，那么关于字体的警告也并非十分重要。尽管如此，我们建议您使用专业工具以确定您的 pdf 是否真的使用了您所要求的字体，而不是某些替代品。Underfull \hbox 是另一个常见警告，通常是编译器无法判断合适的换行点造成的，如果您认为当前排版结果合适就可以忽视它。

问 如何给我的 pdf 嵌入字体？

答 通常而言嵌入字体不是 L^AT_EX 的工作，它只负责输出一份文档。如果您希望嵌入所使用的字体，一般的解决方案是使用第三方专业工具。如果您使用的字体不是太偏门，那么实际上 MiK_TE_X 就会帮您嵌入多数字体。

问 我对计算机工程并不感兴趣，L^AT_EX 适用于我吗？

答 只要您不怕麻烦并且追求比 Word 更好的用户体验，那么无论您的工作方向是什么，L^AT_EX 都很适合。文学作品的排版，算法著作，化学实验报告，物理学论文，甚至是漫画书，都可以用 L^AT_EX 来排版。本文主要面向计算机工程领域进行介绍，但这不代表计算机工程领域是 L^AT_EX 的唯一方向。

问 我可以用 L^AT_EX 来写作业或论文吗？

答 一般而言当然可以，不过具体看贵单位的要求如何。有些机构会强制要求你使用 L^AT_EX 的模板撰写论文，毕竟 L^AT_EX 比 Word 更加容易确定格式。有些学校也允许用 L^AT_EX 写论文，相信用户很容易体会到，用 L^AT_EX 写论文远比用 Word 方便，除非你从来没用过 L^AT_EX。笔者个人强烈建议每所学校都引入 L^AT_EX 作为标准文档工具。

问 作为个人开发者，我如何确定素材的法律风险？

答 一般而言，我们推荐使用有协议可查的素材，无论是图片、字体、代码、商标还是文字作品。这几类都是常见的高民事诉讼风险素材。对于各类素材，尽量寻找明确标注版权的网站，以及个人或组织的官方网站并确定作品应当如何被使用，同时尽可能标注来源以降低受到诉讼的风险。有些正版软件附带一些素材版权，如果您在使用这些软件，就可以在其中免费使用相应素材。（请注意：您购买了附带素材 A 版权的正版软件 B，不代表您在同一台机器上的软件 C 也能合法使用素材 A。）多数情况下，素材会有附带的使用条款或协议，我们建议您在发布前仔细审阅这些法律文件（特别地，不对第三方发布的文件一般也没有法律风险）；应当特别提醒的是，不同语言的翻译版本含义出现冲突时应以原文为准，因此我们认为有必要完整阅读文件原文，而不是翻译。有些协议尽管法律效力不明确，如尚无相关司法解释和判例的，我们也仍然不建议违反它们，除非您不介意自己和自己的产品被作为反面教材被挂起来批评并被各大主流厂商排斥。最好的选择是只使用开放的材料和软件以规避法律风险，如果一定要使用法律风险高的素材，我们建议您在发布前仔细审阅相关法律文件，并在文件上随附声明。用于商业用途则务必咨询律师，除非您想和大厂商的法务部门来场法庭辩论。