



## 第三章 文档的布局与组织

### §3.1 文档类

在一个 L<sup>A</sup>T<sub>E</sub>X 文件导言中的第一条命令通常用来确定整篇文档的全局处理格式。该条命令的语法是：

2<sub>ε</sub> `\documentclass[选项]{类}`

或

2.09 `\documentstyle[选项]{类}`

其中后者只适用于 L<sup>A</sup>T<sub>E</sub>X 2.09。对于 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>，可以用命令 `\documentstyle`，但这只是为了与老的源文件兼容而提供的。

而类的可能取值是：`book`, `report`, `article` 或 `letter`，只能从中选取一种。（在附录 A 中讲解 `letter` 类的性质。）这些类之间的差异包含页面布局和组织方面的不同。一篇 `article`（论文）可以有 `parts`（部分），`sections`（节），`subsections`（小节），等等，而一篇 `report`（报告）还可以有 `chapters`（章）。一本 `book`（书）也有 `chapters`（章），同时对奇偶页区别对待；而且它根据章节标题在每页上打印出当时的页眉。

#### §3.1.1 标准的类选项

用选项可以对格式进行各种不同的修改。它们可如下分组：

##### 选择字体尺寸

可以用如下选项来选择基本的字体尺寸：

2<sub>ε</sub> `10pt` `11pt` `12pt`

这个选项就是在文档中普通文本所取的字体尺寸。默认值是 `10pt`，这意味着如果没有指定尺寸选项时就用这个值。其它所有字体尺寸都是相对于这个标准尺寸而言的，因此如果选定了不同的基本字体尺寸，节的标题，页脚等等就会相应地自动改变尺寸。（注意在 L<sup>A</sup>T<sub>E</sub>X 2.09 中没有 `10pt` 选项；如果想用 `10pt` 做为基本尺寸，只要不指定尺寸选项就可以了。）

##### 指定纸张大小

L<sup>A</sup>T<sub>E</sub>X 是根据选定的字体尺寸和纸张大小来计算行宽和每页行数。同时它也会选择适当的页边，以使文本水平和竖直居中。为了做到这一点，就需要知道所用纸张的格式。下面的选项可以用来指定纸张大小，其全部只存在于 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 中：

$\boxed{2\varepsilon}$ <code>letterpaper(11 × 8.5in)</code>	$\boxed{2\varepsilon}$ <code>a4paper(29.7 × 21cm)</code>
$\boxed{2\varepsilon}$ <code>legalpaper(14 × 8.5in)</code>	$\boxed{2\varepsilon}$ <code>a5paper(21 × 14.8cm)</code>
$\boxed{2\varepsilon}$ <code>executivepaper(10.5 × 7.25in)</code>	$\boxed{2\varepsilon}$ <code>b5paper(25 × 17.6cm)</code>

默认值是 `letterpaper`，即美国信纸尺寸的纸张，大小为  $11 \times 8.5\text{in}$ 。

通常情况下，纸张格式中长的方向是竖直方向，即所谓的纵向模式。利用选项

$\boxed{2\varepsilon}$  `landscape`

可以使短的方向成为竖直方向，即横向模式。

## 页面格式

在一页上的文本可以用下面的选项来使得以一系列或两列形式分布：

$\boxed{2\varepsilon}$  `onecolumn`    `twocolumn`

默认值是 `onecolumn`。当用的是 `twocolumn` 选项时，两列间距以及列间可能存在的标尺的宽度用 `\columnsep` 和 `\columnseprule` 来指定，后面将会讲到它们。

要想奇偶页的页码打印方式不一样，可以用选项

$\boxed{2\varepsilon}$  `oneside`    `twoside`

当用的是 `oneside` 时，所有页码的打印方式是一样的；然而，当用的是 `twoside` 时，在即时标题中若当前页码为奇数，页码出现在右边，若为偶数，页码出现在左边。注意它并不是强迫打印机双面打印。这里的想法是当以后真的双面印刷时，页码总是在每页的外侧，阅读时容易看到。这是 `book` 类的默认值。

对于 `article` 和 `report` 类，默认值是 `oneside`。

对于 `book` 类，每一章通常都是开始于右边，即从奇数页开始。选项

$\boxed{2\varepsilon}$  `openright`     $\boxed{2\varepsilon}$  `openany`

可以控制这个功能：用 `openany` 时，总是在下一页上开始新的一章，而用的是默认值 `openright` 时，必要时可以插上一空页。

通常 `book` 或 `report` 的标题是单独一页的，而在 `article` 中，标题则是与开头的文本在同一页上。利用选项

$\boxed{2\varepsilon}$  `notitlepage`    `titlepage`

可以重定义这种标准行为。请参见 3.3.1 节和 3.3.2 节。

## 其它选项

还剩下的标准选项有：

`leqno` 显示公式中的公式编号出现在左边，而不是像通常那样放在右边（5.1 节）。

`fleqn` 显示公式左对齐，而不是居中（5.1 节）。可以参数 `\mathindent`（下面会讲到它）来设置缩进大小。

`openbib` 参考文献的格式可以改变成每一片断位于一个新行上。默认方式下，每个条目的文本都是聚在一起的。

`draft` 如果 L<sup>A</sup>T<sub>E</sub>X 的断行机制不能很好的发挥作用，总有些文本在右边界突出，那么这个选项就会用一个粗黑条来标识它，以使得突出易见。

<sup>2ε</sup>`final` 与 `draft` 相反，这是默认值。无论文本行有多宽，也不会加上标识。

如果同时给出了多个选项，那就要用逗号把它们分开，例如，

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

选项顺序是无关紧要的。如果同时指定了两个矛盾的选项，如用了 `oneside` 和 `twoside`，无法确定哪个将发挥作用。这主要看在类文件中的具体定义了，因此最后避免出现这种情形。

### 与某些选项相关的参数

有些选项要使用一些已有确定默认值的参数：

`\mathindent`

当选择了 `fleqn` (5.1 节) 时，指定公式相对于左边界的缩进量；

`\columnsep`

为 `twocolumn` 选项指定两列间距 (见 ?? 页上的图 ??)；

`\columnseprule`

为 `twocolumn` 选项确定两列间竖线的宽度。默认值是宽度为零，即没以竖线 (见图 ??)。

这些参数的标准值可以用 L<sup>A</sup>T<sub>E</sub>X 命令 `\setlength` 来修改。例如，要把 `\mathindent` 改为 2.5cm，可以用

```
\setlength{\mathindent}{2.5cm}
```

对这些参数的修改，既可以在导言中进行，也可以在文档的其它任意地方进行。在导言中进行的修改，适用于整个文档，而在文本内的修改，其作用终止于下一次修改，或者它所在的环境结束 (2.3 节)。在后一种情形中，先前的值就开始继续发挥作用。

**练习 3.1:** 打开在练习 2.2 中所用的文本文件，把初始化命令

```
\documentclass{article}
```

先后改为

```
\documentclass[11pt]{article}
```

和

```
\documentclass[12pt]{article},
```

并分别打印出结果。把这两次输出以及练习 2.2 的输出放在一起，比较断行点的不同。

注意：如果有些单词的断开是不适当的，你可以用命令 `\-` 告诉  $\text{\LaTeX}$  哪些地方是恰当的断点，例如，`man\-\u\-\script`。（这是  $\text{\TeX}$  英文单词断开算法不能很好处理的很少的几个单词之一。）在 3.6 节中对修改断开算法的其它方法进行了介绍。

如果在  $\text{\LaTeX}$  处理过程中，有类似于 `Overfull \hbox ...` 这样的警告，这说明  $\text{\TeX}$  不能很好的断开这一行。在输出的结果中，这些行将会越出右边界。一般的原因是  $\text{\TeX}$  不能断开某单词造成的，之所以不能断开某个单词，要么是因为它不可断开，或者  $\text{\TeX}$  单词断开程序不适用。此时在单词中插入建议断点，就可以解决问题。我们很快会给出其它的解决办法。

**练习 3.2:** 现在把那文本文件的初始化命令改为

```
\documentclass[twocolumn]{article}.
```

如果这次你得到了许多 `\Underfull \hbox ...` 警告，那么这些行将会进行左右调整，从而在单词间会有很大的空档。亲自检查一下输出，看一下单词间隔是否还能令人接受。如果不行的话，就试着在下一行开头那个单词中加上建议断点。

注意：如果在前面的练习中你用的是 `book` 类或者 `report` 类，而不是 `article` 类，你会注意到结果实际上是没有区别的。这是因为这些类只对文档的最终结构有作用。一般地讲，对短文章（比如说 10–20 页），应该用 `article` 类，对长篇报告用 `report`，这样可以分章。而且章总是从新页开始。要写书就要用 `book` 类。

### §3.1.2 利用宏包增加功能 2 $\epsilon$

虽然文档类确定了文档的全部一般性质，如页面步局和章节划分，但是也可以用一些更具体的宏包，来改变某些命令的行为，甚至定义一些全新的命令，以增加非  $\text{\LaTeX}$  标准的其它功能。在  $\text{\LaTeX}$  的发行中，有很多这样的宏包，甚至你也可以从  $\text{\LaTeX}$  使用者团体那里获取成千个可用的宏包（见附录 D），当然你也可以自己编写（附录 C）。

一个宏包就是一组  $\text{\LaTeX}$ （或  $\text{\TeX}$ ）命令集，并且被保存在一个后缀为 `.sty` 的文件中，虽然有些特殊命令只能用在宏包中。要调用一个宏包，只需在导言中用

```
2 $\epsilon$  \usepackage{宏包}
```

这里的宏包就是文件的基本名。用一条 `\usepackage` 可以调用不只一个宏包。例如，在标准  $\text{\LaTeX}$  中有两个宏包，保存在文件 `makeidx.sty`（8.4 节）和 `ifthen`（7.3.5 节）中。那么可以用如下方式调用它们：

```
\usepackage{makeidx,ifthen}
```

一个宏包也可以具有与之相关的选项，它以与文档类选项相同的方式来选择，即要把选项名放在中括号内。其一般的语法是：

`\usepackage[选项 1, 选项 2 ...]{宏包 1, 宏包 2, ...}`

这里所列出的选项将适用于所有选择的宏包。如果有宏包不理解所给选项，将在监视器上给出一条警告信息。

至于有哪些选项可用，那就要看具体的宏包了。你必须去阅读随宏包一起的描述和 / 或文档，不但看看到底有哪些选项，还要看看这个宏包是做什么的，它定义了那些新命令。

### §3.1.3 样式选项 2.09

在 L<sup>A</sup>T<sub>E</sub>X 2.09 中并没有 `\usepackage` 命令；事实上，若 `\documentclass` 地方用的是 `\documentstyle`，该命令就根本没有作用。在 L<sup>A</sup>T<sub>E</sub>X 2.09 中，宏包与样式选项以同样的方法处理，即其文件名要加到在 `\documentstyle` 命令中所列出的选项表中。正是由于这个原因，它们也称为样式选项文件，或简称为样式文件，这也就是这些文件后缀 `.sty` 的来源。在 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 中继续使用这个后缀，是为了保证原来的样式文件还可以作为宏包使用。

在真正的选项（在类或宏包文件中对其进行程序设计）和宏包（从与之同名的文件中读取）之间是有本质的区别的。而在 L<sup>A</sup>T<sub>E</sub>X 2.09 中，这种差别是不明确的，经常导致混淆。

在原来版本的 L<sup>A</sup>T<sub>E</sub>X 中，要想向类（或“主样式”）`article` 调入宏包 `makeidx`，同时带有选项 `12pt` 和 `titlepage`，就必须用

2.09 `\documentstyle[makeidx,12pt,titlepage]{article}`

由于这里是把宏包做为选项来处理，显然它们自己不可能再有选项了。

### §3.1.4 全局和局部选项 2<sub>ε</sub>

用 `\documentclass` 命令指定的选项有一个有趣的特点，那就是它适用于所有后面的宏包。这就是说如果有几个宏包具有相同的选项，那就只需在 `\documentclass` 中声明一次。例如，你可以设计一个宏包，修改 `article`，以生成一个局部的特殊样式，使得当文本为单列或双列时做不同的事情；这个宏包可以用类选项 `onecolumn` 和 `twocolumn` 来实现这一点。或者对 `draft` 选项进行精细加工，以生成类似于手稿那样的双倍行距。之与相应的，几个宏包可能具有与语言有关的特征，用选项 `french` 或 `german` 来激活；那么只要在 `\documentclass` 命令中声明就可以适用于所有的选项。这样的选项称为全局的，因为它对后续的所有宏包都有作用。

全局选项并不一定只限于列在 3.1.1 节中的那些标准类选项。如果类和任一宏包都不理解所列出的一个或多个选项，只会显示出一条警告信息。相反地，由 `\usepackage` 命令指定的选项只适用于列于该条命令中的宏包；而且它应适用于其中每一宏包。如果那些宏包中的一个或多个不认识任一局部选项，都会显示出一条警告信息。

## §3.2 页面样式

基本的页面格式是由页面样式决定的。除了一种特殊情形外，该命令通常放在导言中。其形式为：

`\pagestyle{ 样式 }`

对不可省参数，有如下可取的值：

**plain**

页面的页眉是空的，页脚由居中的页码组成。当在导言中没有 `\pagestyle` 时，这是默认值。

**empty**

页眉和页脚都是空的；也不显示页码。

**headings**

页眉由页码及文档类所决定的标题信息（章节标题）组成；页脚为空。  
对每一章的第一页没有作用。

**myheadings**

同 **headings** 差不多，除了页眉的标题不是自动选取，而且由 `\markright` 或 `\markboth` 命令显式决定（见下面的解释）。

命令

`\thispagestyle{ 样式 }`

的作用同 `\pagestyle` 一样，只是它只对当前页起作用。例如，利用命令

`\thispagestyle{empty}`

可以取消当前页的页码。这实际上只是不打印页码，下一页的页码就与没有用该命令一样。

### §3.2.1 生成页眉的声明

对于页面样式 **headings** 和 **myheadings**，出现在页眉中的信息可以用下面的声明来确定：

`\markright{ 右边纸页眉 }`

`\markboth{ 左边纸页眉 }{ 右边纸页眉 }`

`\markboth` 声明用于文档类选项为 `\twoside`，这时假定偶数页位于左边，奇数页右边。而且，对左边页，页码打印在页眉的左边，对右边页，页码打印在页眉的右边。

对于单面输出，认为所有页都是右手方向的。在这种情况下，应该用声明 `\markright`。但对双面输出，也可以用它来重新定义 `\markboth` 中的右边纸页眉。

对于页面样式 **headings**，位于页眉上的标准信息是章、节和小节的标题，具体要看文档和页面样式，对此有如下的图解：

样式		左边纸	右边纸
book, report	单面页	—	章
	双面页	章	节
article	单面页	—	节
	双面页	节	小节

如果在一页上有不只一个 `\section` 或 `\subsection`，就在页眉上显示最后那一个的内容。

### §3.2.2 页的编号

定义页编号的声明形式如下：

`\pagenumbering{ 数字形式 }`

数字形式 可取如下值：

arabic 通常（阿拉伯）数字，

roman 小写罗马数字，

Roman 大写罗马数字，

alpha 小写字母

Alpha 大写字母。

标准值是 arabic。这个声明把当前页码重置为 1。为了用罗马数字显示前言的页码，而其余部分用罗马数字显示，而且第一章开始的页码为 1，那就必须在开始前言时用声明 `\pagenumbering{roman}`，然后紧接第一个 `\chapter` 命令，把其重设为 `\pagenumbering{arabic}`。（在 3.3.5 节中有另一种方法。）

如果想使页码不是从 1 开始编号，那么可以使用命令

`\setcounter{page}{ 页码 }`

这里的 页码 是前一页的编号。

**练习 3.3:** 向你练习用的文本文件中再填一些内容，使其长度不只一页，而且其导言为：

```
\documentclass{article}
\pagestyle{myheadings} \markright{Exercises}
\pagenumbering{Roman}
\begin{document}
```

### §3.2.3 与段落有关的距离

下述参数对段落的形式有影响，可以用在第 7.2 节介绍的 `\setlength` 来给它们设定新值：

`\parskip`

两个段落之间的距离。以 ex 为单位，可以使它随着字体尺寸而自动改



变。其应是一个橡皮长度。

`\parindent`

段落中第一行的缩进量。

`\baselinestretch`

这是一个度量两条基线之间正常间距的数值，所谓基线就是字母所坐的位置，即忽略类似于 g 和 y 这样的字母下挂的部分。这个值初始化为 1，表示标准的线距。它可以用下面的命令改值：

`\renewcommand{\baselinestretch}{因子}`

这里的 因子 是十进制小数，如 1.5 表示增加了 50%。这样其就适用于所用的字体尺寸。如果这条命令放在导言外，那么要直到选择了另一条字体尺寸命令，其才会发挥作用（4.1.2 节）。

这些参数既可以放在导言中，也可以放在文档的其它任何地方。在后一种情况里，这种修改的作用持续到下一次修改为止，或者它所在的环境结束（2.3 节）。

真正的行间距是包含在 `\baselineskip` 这个长度参数中，只要声明了一个字体尺寸，就会自动设置其值。对每一种字体尺寸，都存在一个正常值，把它乘上 `\baselinestretch`，就是 `\baselineskip`。详情请见 4.1.2 节。

如果在一个段落中间修改 `\baselineskip`，新值确定了整段的行间距。更精确地说，在一段结束时的值才有效。

**练习 3.4:** 在你的练习文件的导言中加入下列命令：

```
\setlength{\parindent}{0em}
\setlength{\parskip}{1.5ex plus 0.5ex minus 0.5ex}
\renewcommand{\baselinestretch}{1.2}
```

在处理完这个练习后，把 `\baselinestretch` 的值改一下，如改为 1.5，重复这个练习，这样就会得到知道它的作用。做完这些练习后把上述命令从文件中去掉。

### §3.2.4 页面格式

每一页都是由页眉、包含实际文本的正文，以及页脚组成的。所谓页样式的选择就是确定在页眉和页脚中应包含哪些信息。

对于页眉，正文与页脚之间的距离，上页边界和左页边界，文本行宽，以及页眉、正文和页脚的高度， $\text{\LaTeX}$  都有默认值。在图 3.1 中演示了这些格式化长度的意义。可以给这些长度声明一个新值，这些声明最好放在导言中，使用命令 `\setlength`（7.2 节）。例如，可以用

`\setlength{\textwidth}{12.5cm}`

来使文本行宽变为 12.5cm。

`\oddsidemargin`  
 奇数页的左边界,  
`\evensidemargin`  
 偶数页的左边界,  
`\topmargin`  
 从上页边到页眉顶的距离,  
`\headheight`  
 页眉的高度,  
`\headsep`  
 页眉基线到正文顶部的距离,  
`\topskip`  
 从正文顶部到正文第一行的基线之间的距离,  
`\textheight, \textwidth`  
 主要正文的高度和宽度,  
`\footskip`  
 从正文底部到页脚底部的距离,  
 $2\epsilon$  `\paperwidth, \paperheight`  
 由纸张大小选项所确定的总宽度和高度, 包含所有的页边,  
 $2.09$  `\footheight`  
 已过时, 在  $\text{\LaTeX 2}_\epsilon$  中已被去掉。

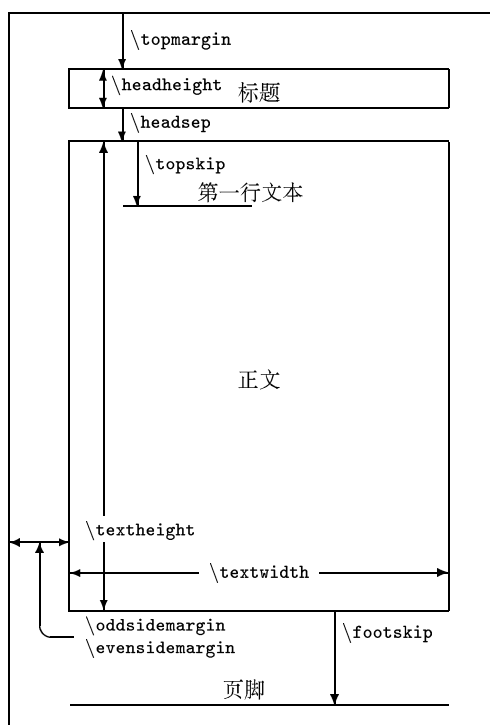


图 3.1: 页面布局参数

在  $\text{\LaTeX 2.09}$  中还定义了一个额外的参数 `\footheight`, 但从来没有用到它。因此在  $\text{\LaTeX 2}_\epsilon$  中就去掉它了这个参数。

对于一列和两列输出时的页面格式, 在附录 ?? 结尾处的图 ?? 和 ?? 中有详细的图示。

为了能准确计算页面布局, 我们必须知道  $\text{\LaTeX}$  中的度量是从纸张上边一英寸的点和纸张左边一英寸的点开始的。因此整个左边界是 `\oddsidemargin` 加上一英寸。已经包含了这个额外一英寸的  $\text{\LaTeX 2}_\epsilon$  参数 `\paperwidth` 和 `\paperheight` 是通过在 `\documentclass` 中的纸张大小选项赋值的; 在内部用它来计算页边界, 从而使文本居中。用户也可以使用这两个参数。

对于文档类 `book` 或者选项 `twoside`, 每一页上正文的底边恰好处在相同的位置上。而对于其它的类或选项, 这就会稍有点儿变化。在前面那两种情形中, 常量底边是用内部命令 `\flushbottom` 来得到的, 而变化的底边是用命令 `\raggedbottom` 生成的。用户也可以用这两个声明来随时改变底边的形式, 从而使之与文档类和选项无关。

**练习 3.5:** 你也可以用上面的参数来改变文本的页面格式。在你练习用文本文件的导言中加入下述指令:

```
\setlength{\textwidth}{13cm} \setlength{\textheight}{20.5cm}
```

这样输出中的上边界和左边界就会很小。给 `\oddsidemargin` 和 `\topmargin` 一新值，以改变这点。

注意：不要忘记了在左边和上边有一英寸的额外长度。当你选择参数值 `\oddsidemargin` 和 `\topmargin` 时必须考虑到这一点。

**练习 3.6:** 继续向所用的文件中添加内容，使其当用的是精简页面格式时也不只两个满页。在导言中加入 `\flushbottom`，那你就会看到所有页上最后一行是在相同的位置上。

**练习 3.7:** 从文件中去掉命令 `\flushbottom`，而选择文档类为

```
\documentclass[twoside]{article}
```

那么现在即使不用 `\flushbottom`，最后一行也在相同的位置上。另一方面，奇数页的左边界与偶数页的右边界可能不一致。可以用 `\evensidemargin` 声明来校正。

### §3.2.5 单双列页面

文档类选项 `twocolumn` 可以使整篇文档的每页都是双列形式显示。而默认值是每页为单列形式。如果要使某一单独页以双列或单列形式排版，可以用下面的声明来达此目的：

```
\twocolumn[ 文本 ]
```

它中止当前页，用两列形式排版后面的每一页。这里的不省略文本是用一列形式显示在页面顶部的，所用的是整页的宽度。

```
\onecolumn
```

中止当前的双列页面形式，用单列形式排版后续的每一页。

选项 `twocolumn` 会自动地改变某些相对于单列格式时的样式参数，例如缩进量。而对于命令 `\twocolumn`，就不会进行这种操作。如果你想要进行这种改变，那就必须用相应的 `\setlength` 声明来实现这一点。如果整篇文档都是双列格式，那么类选项不失为一种最好的方案。

还有一个页面样式参数 `\columnwidth`，它表示一列文本的宽度。对于单列文本，它与 `\textwidth` 是相同的，当选择的是 `twocolumn` 时， $\text{\LaTeX}$  就会根据 `\textwidth` 和 `\columnsep` 的值来计算出其值。文档作者不应该改变这个值，但可以利用它，如果画一个与列文本同宽的标尺。

## §3.3 文档中的部分

每一篇文档都要被分成章、节、小节等等。在结尾处还可能有附录，开头处有标题页，目录和摘要等等。在  $\text{\LaTeX}$  中有许多可用的命令，把用户从这种复杂的格式考虑中解脱出来。除此而外，还可以进行连续的标题编号和小编号。甚至目录也可以自动生成和显示出来。

```

\title{
  How to Write DVI Drivers}

\author{
  Helmut Kopka\thanks{Tel.
    05556--401--451 FRG}\
  Max--Planck--Institut\
  f\"ur Aeronomie
\and
  Phillip G. Hardy
  \thanks{Tel.
    319--824--7134 USA}\
  University\of Iowa}

\maketitle

```

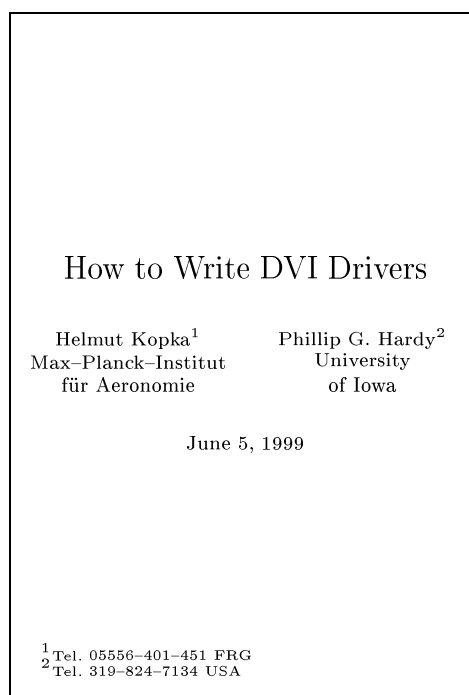


图 3.2: 标题页示例及其结果

某些章节命令的作用与所选用的文档类有关，并不是在所有的类中都可以用所有的命令。

### §3.3.1 标题页

标题页可以用如下环境来生成没有格式的形式：

```
\begin{titlepage} 标题页文本 \end{titlepage}
```

或者用如下命令利用 L<sup>A</sup>T<sub>E</sub>X 已预定义好的格式：

```
\title{ 标题文本 }
```

```
\author{ 作者名与地址 }
```

```
\date{ 日期文本 }
```

在标题页的 L<sup>A</sup>T<sub>E</sub>X 标准格式中，所有内容将来都是以居中的形式出现的。如果标题太长，也会自动断行。作者自己可以用命令 `\` 来选择断行点，也就是说所用形式为 `\title{...\`。

如果有几个作者，其姓名可以用 `\and` 彼此分开，如

```
\author{G. Smith \and J. Jones}
```

这些姓名将在会打印在一行上。而

```
\author{ 作者 1\学术团体 1\ 地址 1
```

```
\and 作者 2\学术团体 2\ 地址 2}
```

就会把 作者 1，学术团体 1，地址 1 和 作者 2，学术团体 2，地址 2 分成两部分，分别一行行的居中排列。而且这相邻的两块在标题页上也是居中的。

如果不想把作者名左右相邻排版，也可以把它们上下排列，这时只要用 `\\` 取代 `\and` 就可以了。此时，可以用紧接 `\\` 后的可以省略的长度定义 [距离] 来调整竖直距离。

如果没有给出 `\date` 命令，就会自动在标题页的作者项后加上当前日期。另一方面，命令 `\date{日期文本}` 就会在出现日期的地方显示 日期文本。这个地方可以插入任何文本，也可以用断行命令 `\\` 以插入多行文本。

命令

```
\thanks{ 脚注文本 }
```

可以出现在 `title`, `author` 和 `date` 文本的任何地方。它会在命令出现的地方加上一个标志，而在标题页上以脚注的形式排版脚注文本。

要想以出现在 `\title`, `\author`, `\date` 和 `\thanks` 中的条目生成标题页，就必须用命令

```
\maketitle
```

标题页本身没有页码，后续文档第一页的页码为 1。（对于 `book`，页码是由 3.3.5 节中的特殊命令控制的。）只有在 `book` 和 `report` 中，标题页才会单独占一页。而在 `article` 中，当使用了命令 `\maketitle` 时，就会以来自于 `\title`, `\author` 以及可能有的 `\date` 和 `\thanks` 中的条目在第一页上创建居中标题头。如果用了 `titlepage` 文档选项，那么即使在 `article` 类中也会让标题位于单独一页上。

图 3.2 是一个 L<sup>A</sup>T<sub>E</sub>X 标准格式的标题页示例。注意由于在标题页的定义中没有出现 `\date` 命令，所以当前日期是自动加上的。可以用这条命令在出现日期的地方显示任何所需要的文本。

对于在 `titlepage` 环境中的没有格式的标题页，命令 `\title` 和 `\author` 是没有作用的，整个标题页的设计来自于作者在这个环境中的定义。此时我们可以利用第 4 章中的所有构造命令。在这种情形里，当结束 `titlepage` 标题页环境时，就会实现标题页的排版，因此也不需用命令 `\maketitle`。

**练习 3.8:** 删掉练习 3.5–3.7 中的改变页面格式的声明。向练习用的文本文件中加入标题 ‘Exercises’，作者就是你自己的姓名和地址，以及日期条目为 ‘地方，日期’ 形式的标题头。为此，在 `\begin{document}` 命令后输入如下命令：

```
\title{Exercises} \author{ 你的姓名 \\ 你的地址 }
```

```
\date{ 你所在的城市, \today } \maketitle
```

要确保你所用的文档类是 `article`。当打印出文档后，把文档类命令改为

```
\documentclass[titlepage]{article}
```

这样就可以把标题信息显示在单独一页上，而不是原来那种标题头形式。

在这些命令的每一行前面加上注释符号 % 以取消它们。要这种方法你可以避免在下面的练习中还会得到标题页，当然可以把 % 去掉，从而很容易地重新激活这些命令。

### §3.3.2 摘要

可以用下面的命令生成摘要：

```
\begin{abstract} 摘要文本 \end{abstract}
```

在文档类 `report` 中，摘要位于单独一页上的，而且有页码；在 `article` 中，摘要是紧接标题头位于第一页上，除非选择了文档类选项 `titlepage`，在这种情形中，摘要也是单独占一页的。在文档类 `book` 中没有摘要。

### §3.3.3 章节

下面的命令可以用来自动生成有序的章节：

```
\part      \chapter    \subsection    \paragraph
          \section     \subsubsection \subparagraph
```

除 `\part` 外，其它命令形成一个章节序列。在文档类 `book` 和 `report` 中，最高的章节层次是 `\chapter`。章用命令 `\section` 分为节，它又可以进一步用 `\subsection` 等等再细分。在文档类 `article` 中，序列是由 `\section` 开始的，因为这时 `\chapter` 是不可用的。

所有这些命令的语法是

`\章节命令 [短标题]{标题}` 或者

`\章节命令 *{标题}`

在第一种情形中，把序列中下一个编号赋给章节，并把这个编号同来自于“标题”的文本一起做为章节标题显示出来。“短标题”文本是用于目录（3.4节）和页眉（假定已选择了 `headings` 页面样式）。如果没有这部分文本，就假定其与“标题”是一样的。除非这里的标题太长，不适用出现在上述地方，否则通常就不出现这部分文本。

在第二种（\* 形式）情形中，不会打印出节号，而且不会在目录表中列出该项（见 3.4.3 节中的例外）。

章节标题的大小和编号的长度与章节命令在序列中的位置有关。对于文档类 `article`，`\section` 命令就生成单个数字（如 7），而 `\subsection` 命令生成两个数字，中间用句号分开这两部分（如 7.3），其它依次类推。

在文档类 `book` 和 `report` 中，用 `\chapter` 命令给出的章标题是单个数字为编号，而 `\section` 得到的则是两个数字，依次类推。而且，`\chapter` 命令总是开始新页，并在章节标题上面打印 **Chapter n**，这里的 **n** 表示当前章编号。在本书当前位置，我们是在 *Chapter 3, Section 3.3, Subsection 3.3.3*。

对于每个章节命令，都有一个内部计数器，每当调用一次命令，对应记

数器就会增一，而当调用下一个更高级的章节命令时，其就会被重置为 0。这些计数器不受 \*- 形式的影响，这个事实在如下情况中会造成一种困难：那就是当标准形式和 \*- 形式混用，而 \*- 形式命令在序列中的位置要比标准形式的高。然而，如果 \*- 形式总是低于标准形式，这就不会产生什么问题。下面的序列

```
\section ... \subsection ... \subsubsection* ...
```

的结果就是 \section 和 \subsection 有编号，而 \subsubsection 没有编号。

章节命令 \part 是一个特殊情形，它对其它命令的编号没有影响。

章节的自动编号就意味着在写作的时候，可能不需要知道编号。作者没有必要按最终的顺序进行创作，也可以加进或去掉一些章节。如果作者在正文中想引用某一章节的编号，那么这就需要一种不是直接输入编号的机制。将在 8.3.1 节描述 L<sup>A</sup>T<sub>E</sub>X 的交叉引用系统，它利用如下两条基本命令完成这个任务：

```
\label{名称}      \ref{名称}
```

前一个给章节编号一个名称或关键词，而后一个则是用在正文中，以引用章节的编号。关键词名称可以是字母、数字或符号的任意组合。例如，本书中在 8.3.1 节开头使用了命令 \label{sec:xref}，因此上面这句话中就只需包含进命令 \ref{sec:xref}。

还有一个引用命令，那就是 \pageref，它显示对应的 \label 命令所在那页的页码。

引用命令还可以用在其它许多地方，前提条件只要那些项的编号是自动生成，如图，表和公式。

每一个章节命令都有一个层次号，如 \section 总是第 1 层，\subsection 为第 2 层，...，\subparagraph 为第 5 层。在文档类 article 中，\part 为第 0 层，而在 book 和 report 类中，\part 为第 -1 层，\chapter 成为第 0 层。章节编号向下进行的层次由 secnumdepth 决定。对于 book 和 report，其值为 2，而对于 article，其值为 3。这也就是说对于 book 和 report，章节编号只进行到 \subsection 层次，而对于 article，则是到 \subsubsection。

为了拓展（或减小）章节编号的层次，那就必须改变 secnumdepth 的值。这可以用命令

```
\setcounter{secnumdepth}{数}
```

来实现。（在 7.1 节中对 \setcounter 命令进行了解释。）在 article 中，secnumdepth 可以取 -1 到 5 之间的值，而对于 book 和 report，可取 -2 到 5 之间的值。取最小值意味着禁止所有的章节编号。

在文档中也可以用如下命令来改变章节命令的初始值：

```
\setcounter{章节名称}{数}
```

这里的 章节名称 指的是没有前缀 \ 的章节命令名称。当要用 L<sup>A</sup>T<sub>E</sub>X 处理某个章节时，这个命令是相当有用的。例如，

```
\setcounter{chapter}{2}
```

就把 \chapter 计数器设为 2。当下一次调用 \chapter 时，其值会增 1，即得到 **Chapter 3**。

有时候需要改变章节标题的字体大小和样式。这可以用在 4.1.2–4.1.6 节中描述的用于章节标题文本的声明达此目的。例如，

```
\section*{\Large\slshape Larger Slanted Font}
```

就会以 \Large 尺寸和 \slshape 样式打印出节标题 ‘Larger Slanted Font’。我们不推荐使用这种抛弃标准字体的做法，因为这些修改只对标题文本有作用，而对其前面的编号没有影响。

### §3.3.4 附录

可以用命令 \appendix 来加进附录。这一命令具有重设 article 中的节计数器 and book 与 report 中章计数器的作用，而且它把这些章节命令的编号形式从数字变为大写字母 A,B, ...。另外，还用单词 ‘Appendix’ 代替 ‘Chapter’，因此后续章节的标题前缀变为 ‘Appendix A’, ‘Appendix B’, 等等。低层章节命令的编号中用字母取代了章节编号，如 A.2.1。另外一种方法是，附录也可以包含在如下一个环境中：


```
\begin{appendix}
```

附录文本


```
\end{appendix}
```

### §3.3.5 书的结构

为了简化书的结构，在 book 类中提供了命令

 \frontmatter

前言，目录

 \mainmatter

正文的主体

 \backmatter

参考文献，索引，版本记录

\frontmatter 命令把页码切换为罗马数字格式，并且不对章进行编号；另一条命令 \mainmatter 把页码重设为阿拉伯数字 1，并激活对章的编号；而在 \backmatter 中又再次停止对章的编号。

**练习 3.9:** 在你练习用的文本开头插入命令 \section{Title A}，在靠近中间的某一适当的地方插入 \section{Title B}，这里你要用相应的文本取代



Title A 和 Title B。然后在适当的地方插入合适的 `\subsection` 命令。去掉练习 3.3 中加入的下列的命令：

```
\pagestyle{myheadings} \markright{Exercises}
\pagenumbering{Roman}
```

打印生成结果。

**练习 3.10:** 在第一个 `\section` 命令前加上命令 `\chapter{Chapter title}`，这里的 Chapter title 可以取你自己喜欢的任何内容做为章的标题。把文档类命令改为 `\documentclass[twoside]{report}`，并在导言中调用页面样式命令 `\pagestyle{headings}`。注意在标题和页眉中的章节命令的两面效果。把这里的结果与 3.2.1 节中的表格做比较。

**练习 3.11:** 把章命令改为

```
\chapter[Short form]{Chapter title}
```

并给出章标题的适当缩减形式。那么原来出现完整章标题的页眉，现在取代的是短标题。

## §3.4 目录表

### §3.4.1 自动条目

L<sup>A</sup>T<sub>E</sub>X 可以自动为整篇文档准备和打印一个目录表。这个表包含章节号和相应的在章节命令标准形式中的标题，以及章节起始页码。出现在目录表中的章节深度可以在导言中用命令

```
\setcounter{tocdepth}{数}
```

来设定。这里的 数 与上面描述的 `secnumdepth` 计数器中的意义相同，而自动分章节的最大层次是固定的。到哪一层次的章节可以包含在目录表中，与自动章节编号中的是一样的，也就是说，对 `book` 和 `report` 到 `\subsection`，对 `article` 到 `\subsubsection`。

### §3.4.2 显示目录表

目录表的生成和显示是用下面的命令实现的：

```
\tableofcontents
```

把这条命令放在希望目录所位于的地方，通常就放在标题页和摘要的后面。

这就导致了一个两难的处境，因为目录表的信息是显示在靠文档开头的地方，而这些信息直到文档结束才可能全知道。L<sup>A</sup>T<sub>E</sub>X 是如下解决这个问题的：当文档第一次被处理时，并没有包含进任何目录表信息，而是由 L<sup>A</sup>T<sub>E</sub>X 打开一个与文档文件同名，而后缀为 `.toc` 的新文件；在后面的处理过程中，把目录表所需的条目写到这个文件中。

当下次对这个文档运行 L<sup>A</sup>T<sub>E</sub>X 时, `\tableofcontents` 命令使得文件文档名 `.toc` 被读入, 从而打印出目录表。当继续后面的处理时, 如果在运行后前一次后做了很大改变, 那么 `.toc` 文件会被更新。也就是说打印出来的目录表总是对应于前一次文档的。正是由于这个原因, 对于最后的文档, 应该运行两次 L<sup>A</sup>T<sub>E</sub>X。

### §3.4.3 其它条目

\*- 形式的章节命令不会自动加入到目录表中。为了把它或其它条目插入到目录表中, 可以用命令:

```
\addcontentsline{toc}{章节名称}{条目文本}
```

```
\addtocontents{toc}{条目文本}
```

用第一条命令, 就会形成目录表格式的相应条目, 其中 `section` 标题头要比 `chapter` 的向里缩进一些, 但比 `subsection` 缩进的少。这是由章节名称参数决定的, 取值为章节命令没有了前缀 `\` 字符 (如 `section`)。条目文本与页码一起插入到目录表中。如果想同标准形式那样也有章节编号, 那么条目文本必须具有形式 `\protect\numberline{章节名称}{文本}`。

`\addtocontents` 命令把任何所希望的命令或文本加入到 `.toc` 文件中。它可以是一条格式化命令, 如 `\protect\newpage`, 当显示目录表时这些命令会发挥作用。

### §3.4.4 其它列表

除了目录表, 图与表的清单列表也可以由 L<sup>A</sup>T<sub>E</sub>X 自动生成和显示出来。生成这些列表的命令为:

```
\listoffigures 读和 / 或生成文件 .lof
```

```
\listoftables 读和 / 或生成文件 .lot
```

在这些列表中的条目是根据 `figure` 和 `table` 环境中的 `\caption` 命令自动生成的 (见 6.6.4 节)。其它条目可以用与目录表相同的命令生成, 它们的一般形式为:

```
\addcontentsline{类型}{格式}{条目}
```

```
\addtocontents{类型}{条目}
```

这里的 `类型` 代表 `toc` (目录表)、`lof` (图的列表) 和 `lot` (表格的列表) 三种类型中的一种。`格式` 参数就是上面描述的目录表中的章节名称, 或者图列表中的 `figure`, 以及表格列表中的 `table`。`条目` 参数表示要插入相应文件中的文本。

**练习 3.12:** 在练习文件中, 在抑制了标题页命令后插入命令

```
\pagenumbering{roman}
```

```
\tableofcontents \newpage
```

```
\pagenumbering{arabic}
```

用 L<sup>A</sup>T<sub>E</sub>X 处理两次练习文件，并打印出第二次的结果。在做下面的练习之前，用 % 符号去掉这些命令。

注意：如果章节标题中有命令，那么应该前缀 \protect 命令，以保证它们被安全地传送到相应文件中。

## §3.5 精调正文

### §3.5.1 单词与字符的距离安排

在单词和字符之间的距离，通常都是由 T<sub>E</sub>X 自动安排的，它不但利用字符的自然宽度，而且还考虑特定字符组合的变化。例如，如果 A 后接 V，结果并不是 AV，而是 AV；也就是说为了好看，它们彼此稍移近了一点。在一行中单词之间的距离是一致的，而且距离的选择保证左右边界与页边是对齐的。这也称为左右调整。T<sub>E</sub>X 也尽力使不同行中单词间的距离尽可能一样。

由标点符号结尾的单词，其后要给出一个额外的空档，具体大小要看到底是哪种符号了：接在句号 ‘.’ 或惊叹号 ‘!’ 后面的距离就要比接在逗号后面的大。这相应于英文的排版规则：在句子之间应有多一点的空档。在某些情形中，自动安排的结果可能不会令人满意，这样就可以按下面几节中的方法进行重新定义。

#### 句子结束与句号

T<sub>E</sub>X 把一个接在小写字母后的句号认为是句子的结束，这时就会插入额外的单词间距。这样就会与缩写产生混淆，如 i. e., Prof. Jones 或 Phys. Rev., 此时需要的是正常单词间距。这时可以用字符 ~ 或 \\_ 取代通常的空格。（字符 \_ 只是为了表示一个空格，否则根本看不到它。）这两种方法都是插入通常的单词间距；而且 ~ 禁止在这一点断行。所以上面的例子应该输入 i.~e., Prof.~Jones 和 Phys.\ Rev.，以得到输出 i. e., Prof. Jones 和 Phys. Rev.，这时的间距是很正常的，而且必要时也强迫其中的某些相邻字符必须在一行上。在第三种情形中，把 Phys. 和 Rev. 放在两行上并没有什么不妥。

紧接在大写字母后的句号并不认为是句子的结束，而只认为是一个缩写。可如果它真的是结束一个句子，那就需要在句子前面加上 \@，以得到额外的间距。例如，句子 ‘this sentence ends with NASA.’ 就应该输入为 this is sentence ends with NASA\@.

#### 法语间距

可以用命令 \frenchspacing 来告诉 T<sub>E</sub>X 不必增加某些情形中的额外单词间距，当再用 \nonfrenchspacing 命令时，其又会恢复到原来的方式。对

前一种情形, 命令 `\@` 是被忽略的, 可以不用。

### 字符组合 “‘和’”

用命令 `\`, 可以生成一个很小距离。这一点是有用的, 例如, 当双引号 “和” 与单引号 ‘和’ 在一起时, 可以用 `\`, 把它们分开。例如, 文本 “‘\,’Beginning’ and ‘End’\,” 的输出为 “‘Beginning’ and ‘End’”。

### 倾斜校正

当字样从斜体 (意大利斜体或 *slanted*) 变到竖直字样时, 由于最后一个字符可以斜穿进接下来的空档中, 从而使得这个距离显得比原来要小。因此必须在这儿加上根据不同字符而不同的额外间距 (*d* 的要比 *a* 的大)。TeX 用命令 `\/` 来加进这个额外间距, 称之为倾斜校正。当一个倾斜字母后接一个竖直字母时, 总要用这种校正, 例如 `{\slshape slanted\/} spacing` 可以得到 *slanted spacing*。当倾斜字母后接逗号或句号时, 这种校正可以不加。

L<sup>A</sup>TeX 2<sub>ε</sub> 的字体命令 `\textsl` 和 `\textit` (4.1.4 节) 会自动加上倾斜校正, 这也是它比字体声明 `\slshape` 和 `\itshape` 好用的另一个优点。例如, 不要用 `{\slshape slanted\/}`, 而应该用 `\textsl{slanted}`。然而, 有时候我们可能希望没有这种校正, 那么可以在倾斜字母后面用命令 `\nocorr` 以达此目标。

### 取消连写

`\/` 命令也可以用来禁止连写, 所谓连写, 就是特定的字母组合作为一个字符印刷。文本 `shelf\/ful` 的结果是 *shelfful*, 而不是 *shelfful*。在 2.5.8 节中已提到, TeX 把字母组合 *ff*, *fi*, *fl*, *ffi* 和 *ffl* 处理成连写。输入 `f\/f\/i`, 可以得到 *ffi*, 而不是 *ffi* 连写。

`\/` 命令也可以用来取消特定字母组合之间的距离减少, 如 *AV* 和 *Te*:

`A\/V AV T\/e Te` 而不是 *AV Te*。

### 插入任意距离

可以用下面的命令在文本间插入任意大小的距离:

`\hspace{ 距离 }`

`\hspace*{ 距离 }`

这里的 *距离* 就是所要指定的间隔长度, 例如 1.5cm 或 3em。(注意一个 *em* 就是当前字样中字母 *M* 的宽度。)

这两条命令在调用点和接下来要显示的对象之间插入宽度等于上述距离的空白。用标准形式 (没有 *\**), 可以使得若间隔位于两行之间时, 就去掉这个的空档, 这就如同在一行的开头, 空格要去掉一样。而另一方面, *\**- 形式不管任何情况, 都会插入空白。

这里定义的距离也可以是负数，这样该命令就如同退格，在另一个字符上打印新的字符。

在这条命令前后的空格仍然有作用：

```
This is\hspace{1cm}1cm    This is      1cm
This is \hspace{1cm}1cm    This is      1cm
This is \hspace{1cm} 1cm    This is      1cm
```

命令 `\hfill` 就是 `\hspace{\fill}` 的缩略形式（见 2.4.2 节）。它会在其两边的文本之间插入足够大的空白，从而使它们分别靠近左右页边。Left `\hfill` Right 的输出就是

```
Left                                                                    Right
```

在一行上有多个 `\hfill`，那个它们每个都会插入相同宽度的空白，从而使这一行变得左右协调。例如，输入 Left `\hfill` Center `\hfill` Right 的结果为

```
Left                                                                    Center                                                                    Right
```

如果 `\hfill` 位于一行的开头，根据 `\hspace` 标准形式的行为准则，这个空白是被去掉的。如果确实需要在一行的开头或结尾加上一个橡皮空白，那就必须用 `\hspace*{\hfill}`。然而， $\text{\LaTeX}$  还提供许多命令和环境，以简化许多这种情形的应用（见 4.2.2 节）。

也可以用许多其它的长度固定的水平距离：

`\quad` 和 `\qquad`

命令 `\quad` 插入一个长度等于当前字样尺寸的空白，即若当前为 10pt 字样，则插入 10pt，而 `\qquad` 是其两倍。

### 插入长度可变的 ..... 和 \_\_\_\_\_ 序列

还有两条命令，其用法同 `\hfill` 完全一样：

`\dotfill` 和 `\hrulefill`

当不想插入空白时，可以用这些命令如下插入点或直线：

```
Start \dotfill\ Finish\\
Left \hrulefill\ Center \hrulefill\ Right\\
```

的结果为

```
Start ..... Finish
Left _____ Center _____ Right
```

可以在一行上出现 `\hfill`，`\dotfill` 和 `\hrulefill` 的任意组合。如果在同一地方多次使用这种命令，那么相应的填充就单个时的要大很多。

```
Departure \dotfill\dotfill\dotfill\ 8:30 \hfill\hfill
Arrival \hrulefill\ 11:45\\
```

结果为

Departure ..... 8:30

Arrival \_\_\_\_\_ 11:45

### §3.5.2 断行

把文本断成行的操作是由  $\text{T}_{\text{E}}\text{X}$  和  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  自动进行的。然而，也有这样的时刻，你想强迫或希望在某地方断行，或不想在某地方断行。

#### 命令 `\`

可以用命令 `\` 来得到一个有或没有额外行距的新行。该命令的语法为

`\`[ 距离 ]

`\`\*[ 距离 ]

这里的可省参数 距离 是一个长度，它用来定义增加多少额外的行间距。如果此时需要开始一个新页，那就不加这个额外间距，新页开头为下面的文本行。\*- 形式禁止在两行之间分页。

用 `\`\*[10cm]，当前行就会结束，在这行与后面文本加上一个 10cm 的竖直距离，而且这两行文本必须在同一页上。如果需要一个分页，那就会在当前行前面进行，这样当前行，10cm 的竖直距离，以及后面的文本就位于新页上。

命令 `\newline` 与没有可省参数的 `\` 命令是一样的。也就是说，新行之前没有额外的空白，而且此处也可以分页。

这两条命令都可以用在段落中间，除此而外，它们再没有其它意义。

#### 其它的断行命令

命令 `\linebreak` 用来鼓励或强迫在文本中某点断行。它的形式为

`\linebreak`[ 数 ]

这里的 数 为可省参数，它是一个介于 0 到 4 之间的数，用来表示断行的重要性。这条命令鼓励断行，越大的数，鼓励的力度就越大。取 0 值表示允许在本来不会断行的点处（如一个单词的中间）可以断行，而 4 意味着必须断行，它等同于没有参数的 `\linebreak`。这条命令与 `\` 或 `\newline` 的区别在于当前行要进行左右调整，增加单词间距，以使文本充满该行。而对于 `\` 和 `\newline`，当前行用空白填充最后一个单词后的空白，单词间距保持正常值不变。

与之相反的命令为

`\nolinebreak`[ 数 ]

它不鼓励在给定点断行，这里的 数 表示不鼓励的力度。而且没有参数的命令 `\nolinebreak` 同 `\nolinebreak[4]` 有一样的效果，即在这里绝对不可能进行断行。

另外一种使文本呆在一行上的方法是用命令 `\mbox{文本}`。对于类似于 ‘Voyager-1’ 这样的表达式，如果想在连字符处禁止断行，用这个命令就是非常方便的。

### §3.5.3 段落间距

段落间的正常间距是由长度 `\parskip` (3.2.3 节) 设置的，可以用命令 `\setlength` 来改变其初始值：

```
\setlength{\parskip}{距离}
```

也可以用如下命令在某一特定段落间加上额外的竖直距离：

```
\vspace{距离}
```

```
\vspace*{距离}
```

即使在此处开始一个新页，或者命令位于新页的开头，`*`- 形式也要加上这个额外空档，而同样对这两种情形，标准形式禁止加上额外竖直距离。

如果在一个段落中间用这两条命令，那就在当前行后面加上额外间距，而且按能常那样对当前行进行左右调整。

距离参数可以为负数，这样可以抬高后面文本的相应于正常打印的位置。

命令 `\vfill` 是 `\vspace{\fill}` (见 2.4.2 节) 的缩略形式。其等价于水平间距的 `\hfill`，它会插入足够的竖直空白，以使得文本的顶部和底部与页面的顶部和底部对齐。对于同时出现多个 `\hfill` 的解释，也同样适用于 `\vfill`。如果这条命令位于一页的开头，它没有任何作用，这同标准形式 `\vspace{\fill}` 一样。如果想在页面底部加上一个橡皮间距，那就必须用 `*`- 形式 `\vspace*{\fill}` 了。

其它的用于增加段落间距的命令是

```
\bigskip \medskip \smallskip
```

这些命令根据在文档类中声明的字体尺寸来确定增加的竖直距离。

如果不用空行，也可以用命令 `\par` 来表明一个段落的结束。

### §3.5.4 段落的缩进

段落缩进的大小是由长度 `\parindent` (见 3.2.3 节) 决定的，可以用命令 `\setlength` 改变它的值：

```
\setlength{\parindent}{距离}
```

这样每段的第一行左边都有一个宽度为上面距离参数指定的空白。如果想取消某一段落的缩进，或者如果没有缩进（如一节的第一段）时想强迫出现缩进时，可以用下面的命令做到：

```
\noindent 和 \indent
```

要把它放在段落的开始，这样它们才会发挥作用。

### §3.5.5 分页

在  $\text{T}_{\text{E}}\text{X}$  和  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$  中，同自动断行一样，把文本分成页的操作也是自动进行的。同样的，也可以影响程序确定何处为分页点。

#### 正常分页

命令

`\pagebreak[数]`

`\nopagebreak[数]`

等价于断行时的 `\linebreak` 和 `\nolinebreak`。如果 `\pagebreak` 位于两段之间，那就会在此处进行分页。如果它位于一段的中间，那么就会在当前行完成后进行分页。同样要对该行进行左右调整。

命令 `\nopagebreak` 具有相反的作用：在两段之间，它禁止在这儿分页；在一段中间，那它禁止可能在当前行后面出现的分页。

介于 0 到 4 之间的可省略参数表达了鼓励或不鼓励分页的力度。而且 `\linebreak` 要做更多的事：除了把断开的行增加单词间距，进行左右调整，还要增加行间距，以使页面顶部和底部分布一致。

要在一页中间结束该页，用空白填充余下空间，再开始一个新页的命令是

`\newpage`

它等价于分页中的 `\newline` 命令。

**练习 3.13:** 在练习文件中适当的一个地方或两个地方插入 `\newpage` 命令。在输出中，将会看到页面下面是空的。现在去掉这些命令，在一个自然分页点后面一行或两行处插入一个 `\pagebreak` 命令。处理完后，把命令移到正常分页点前面几行处。看看这几种修改时的结果。

#### 有图表时的分页

如果文本中有表格，图画或者为插图预留的空间，它们会被插入在与相应于命令出现的地方，前提条件是在当前页上要有足够的空间。如果没有足够的空间，就会继续正文，而把图表保留到下一页上。

命令

`\clearpage`

同 `\newpage` 命令一样结束当前页，而且用一页或多页来输出没有打印的图表。（请见 6.6 节。）

#### 两列模式时的分页

如果选择了文档类选项 `twocolumn`，或者用了命令 `\twocolumn`，那么命令 `\pagebreak` 和 `\newpage` 就会结束当前列，开始一个新列，即把列当做页



处理。另一方面，`\clearpage` 和 `\cleardoublepage`（见下面）就是中止当前页，如果必要的话，插入一个空列。

### 两面模式时的分页

当选择了文档选项 `twoside` 时，还可以用一个分页命令：

`\cleardoublepages`

它的作用同 `\clearpage` 一样（当前页被中止，所有未打印的图表输出在后面的页上），而且接下来的页应是奇数页。如果必要的话，由此会得到一个具有偶数页码的空页。

### 有限制的分页

声明 `\samepage` 可以使得分页只会发生在段落之间，显示公式或文本的前后，一个章节标题的前后，或者列表环境中的 `\item`（第一个除外）之间。如果想在正文其它地方进行分页，那就必须用命令 `\pagebreak` 了。

实际上这一功能的作用并不会同期望的那样好，正是由于这个原因， $\text{\LaTeX 2}_\epsilon$  提供了一些新方法，以控制不必要的分页。只是为了与原来文档兼容，才继续保留了 `\samepage` 命令。

### 控制分页

$\text{\LaTeX 2}_\epsilon$  用如下命令提供了稍微增加当前页高度的可能性：

$\boxed{2\epsilon}$  `\enlargethispage{尺寸}`

$\boxed{2\epsilon}$  `\enlargethispage*{尺寸}`

这两命令把定义中的 `尺寸` 只加到当前页的 `\textheight` 上。有时候只需要稍微一点儿的调整，就可以避免一个糟糕的分页。命令的 `*`-形式在当前页上无论如何也要调整行距，以最大化文本高度。

### 分页的其它方法

在 3.2.4 节中已提到，在文档类 `book` 或选项 `twoside` 中，在每一页上，页面底边的位置是一样的，而在其它情形中则不然。这意味着在这两种情形里，将需要在适当的位置（如落段之间）插入行间距。

这种情形是由于在 `book` 和 `twoside` 中，有声明 `\flushbottom`，而在所有其它情形中，则是 `\raggedbottom` 声明造成的。它们最初是在内部类和选项的定义中设置的，但用户也可以在导言中或正文的其它地方用这些命令。除非遇到相反的命令，或当前环境结束，否则该命令一致有效。

通常是不可能用选择页面长度 `\textheight`（3.2.4 节）来在一页上放更多的文本。然而，如果确实需要多挤进一行或两行文本，那就必须为它们清除竖直间距。这可以用放在适当点的带负长度的 `\vspace` 或 `\` 命令来做到。最佳地方是在显示，枚举，列表，表格或插图的前后（见第 4 章）。

我们在这里不会去详细讲解分页的内部规则，只提及下面一点，对那些不鼓励在此分页的点， $\text{\TeX}$ 要加上所谓的罚点（对应于命令 `\penalty`），而负的罚点则意味着鼓励分页。段落中的每行都对应着一个罚点，比如说是 10 个罚点（`\linepenalty=10`），这样就可以使得在段落中间分页比段落之间分页要难。

段落的第一行和最后一行给定 150 个罚点（也就是指 `\clubpenalty=150` 和 `\widowpenalty=150`），这样以不鼓励（但没有禁止）在一页上只留下一段的一行。

因此，做为一种手段，我们也可以通过修改罚点值，以获取所需的分页。例如，若取 `\clubpenalty=450`，那么在段落中第一行后分页就要比平常时的难上三倍。把罚点值取为 10000 或更多，就会使得绝对不会在此处分页。然而对罚点的修改，只应该用在其它方法都行不通的时候，因为这破坏了与其它部分文档之间的平衡，从而可能引起更多古怪的分页。

## §3.6 断词

当要对一行进行左右调整时，有时候会出现无法在单词之间断行的情形，因为这样做要么会使单词挤得很紧，要么彼此之间离得很远。这样就需要分开一个单词。这一重大的任务是由  $\text{\TeX}$ （即  $\text{\LaTeX}$  的本质基础）来完成的，它用的断词算法对英语（绝大多数作者用的都是这种语言）文本（绝大多数情形下）处理的很好，然而，它也有出错的时候，这就需要人来干预它了。

如果通常的  $\text{\TeX}$ / $\text{\LaTeX}$  用于其它语言，或者在英文中出现了外文单词，也很有可能出现不正确的断词。（在 3.6.5, 3.6.6 节和 D.1 节有关于  $\text{\LaTeX}$  与其它语言共用的更多讨论。）在这种情况下，那也必须用如下的方法重定义  $\text{\TeX}$  的断词算法。

### §3.6.1 人工断词

最简单地纠正一个被错误断开的单词的方法就是在这个单词中间的恰当位置上插入命令 `\-`。例如对于单词 `manuscript`，根本不可能把它断开，因此如果在断行时它造成了一些问题，那可把它写成 `\man\-\u\-\script`。这就告诉  $\text{\TeX}$  在必要时可以把它断成 `man-uscript` 或 `manu-script`，而不用顾及通常的规则。

`\-` 只是使得在指定位置有可能断开；而没有强迫在此处断开。如果作者坚持要在某一点断开某个单词，比如说在 `manuscript` 的 `u` 和 `s` 之间，那可以输入 `manu-\linebreak script` 来做到。但是，我们并不推荐这种粗暴的做法，因为不管以后对正文有没有修改，这个地方总是有个断行点。

对于英文，单词的拼写即使断开也是不变的，而对于其它语言，这可就

不一定了。例如在德语中，就有一条规则，如果 ck 被分开，它就变成 k-k。  
在 TeX 中可以用一般断开命令来做到这一点：

```
\discretionary{ 前 }{ 后 }{ 无 }
```

这里的 前 与 后 都表示字母（有连字符），它们指的是如果要断开的话，在断点两边的字母，而无 则是没有断开时的正常写法。因此 Boris Becker 的姓名应该写成

```
Boris Be\discretionary{k-}{k}{ck}er
```

只有在特殊情形下才需要这种输入。顺带提一下，\- 命令的定义就是

```
\discretionary{-}{-}{-}。
```

### §3.6.2 断法列表

断开不正确，但是又经常在正文中出现的单词可以在导言中放进一个异常列表，以避免每次费事地插入 \-：

```
\hyphenation{ 列表 }
```

这里的 列表 就是一组单词，用空格或换行分开，而且用连字符表示可允许的断点。例如，

```
\hyphenation{man-u-script com-pu-ter gym-na-sium  
coun-try-man re-sus-ci-tate ...}
```

这个列表中可以包含通常的从 a 到 z 的所有字母，但不能出现特殊字殊或重音。

### §3.6.3 禁止断词

另外一种避免糟糕断词的做法就是为至少一两段关闭断词。实际上命令

```
\begin{sloppypar} 段落文本 \end{sloppypar}
```

并没有禁止断词，只是允许更大的单词间隔，而不会给出警告信息。这就意味着实际上所有行都是在单词间断开的。也可以在导言中或当前环境中加入命令 \sloppy 来减少整篇文档或当前环境中的被断开的单词数目。当行相当窄时，推荐使用这种方法。

当命令 \sloppy 起作用时，那么可以用如下环境来暂时重新打开断词：

```
\begin{fussypar} 段落文本 \end{fussypar}
```

在环境中用命令 \fussy 也可以得到相同的效果。

### §3.6.4 行宽与断词

现在有必要在这里补充一些关于行宽与断词的注解。

当 TeX 处理到一段末尾时，它就尝试在考虑 3.5.1 节中提到的间距要求的前提下，通过在单词间断行，把文本组织成长度相等的行；这也说是，不会出现断词。如果这种尝试行不通，那它就开始尝试在单词中间断开。当行宽很大，或者单词的平均长度很小时，在单词之间断行是相对比较容易的。

而且, 对于给定的行宽, 如果字体尺寸是小的, 那么被断开的单词数目也会少些。

若行宽很窄, 即使可以在单词中断开, 可能左右调整也很困难, 这样  $\text{\TeX}$  就需要插入比正常情况下所允许的要多的单词间距。要做到这一点, 可以用上面给出的 `sloppypar` 环境或 `\sloppy` 命令。这样可以进行调整, 但是代价是在单词间加进了不可接受的间距。在这两种情形中,  $\text{\TeX}$  都会打印出一条警告信息, 要么是 `Overfull \hbox` (不可能左右对齐, 一个单词将向右边突出) 或 `Underfull \hbox` (单词间距太大)。如果无法容忍这两种情形, 那就必须对正文进行调整, 利用命令 `\linebreak` 和 `\hfill` 可能会强迫出现适当的断词点。

### §3.6.5 其它与断词有关的内容

$\text{\TeX}$  需要用来进行断词的信息保存在一个 `hyphen.tex` 的内部文件中, 其中包含一组供特殊断词算法所用的字母组合。这个列表是基于所考虑语言的统计研究结果的。这个文件中也包含一些例外, 算法不能正确地对这些单词进行断开。  $\text{\TeX}$  首先在这个例外列表中搜索, 根据在其中可以找到的模式断词; 如果在其中没有找到, 那它就用它的算法来断词。对于其它语言, 那就必须给出不同的字母组合列表和例外列表了。

在文件 `hyphen.tex` 的尾部可以找到例外列表, 其由前面给出的命令 `\hyphenation` 组成。这个列表中的每一项都可由文本编辑器展开。在一个大型计算中心, 只有被授权的  $\text{\TeX}$  用户才可以对它进行修改。而在 PC 机上, 用户必须查阅  $\text{\TeX}$  安装手册, 看看 `hyphen.tex` 文件在哪儿。

每次对断词文件进行改变, 对就必须生成一个新的格式文件。这是一种针对于  $\text{\LaTeX}$  的主要宏定义文件, 它以可快速输入的紧致方式存贮。关于安装  $\text{\LaTeX 2}_\epsilon$  的更多信息见 D.4 节。

顺便提一下, 你可以用下面的命令在屏幕上查看  $\text{\TeX}$  断词算法的结果:

```
\showhyphens{ 单词表 }
```

这会打印出  $\text{\TeX}$  对 单词表 中所有单词的断开结果。例如, 给出:

```
\showhyphens{interrelations penumbra summation}
```

那在屏幕上就会显示出

```
in-ter-re-la-tions penum-bra sum-ma-tion
```

### §3.6.6 多语言文本中的断词

在  $\text{\TeX 2.99}$  或更低的版本中, 在一个 `plain.fmt` 或 `lplain.fmt` 中只允许有一组用于断词的字母组合。这些版本的  $\text{\LaTeX}$  程序可适用于英文或者另一种语言。在一个文档中不可能把多种语言的断词算法混用。

从  $\text{\TeX}$  的 3.0 版本起, 在格式中可以包含多个断开列表, 这样就可以在一

个文档中切换到不同的断开框架中,所使用的就是新的 TeX 命令 `\language`。这一命令也可以用于与语言相应的改编中,把某些特定的英文单词翻译输出(如 ‘Contents’),以简化重音和标点符号,以及修改日期命令 `\today` 的定义。在 C.3.3 和 D.1.4 节给出了多语言和 `\language` 命令的更多信息。