

CuS_{TE}X 宏集手册

Longaster

2022 年 8 月 18 日 v0.0.1

CUSTEX

总目录

总目录	i	2.6.1 初始化设置	21
第一章 概述	1	2.6.2 编号	21
第二章 文档接口	1	2.6.3 格式	22
§ 1 util 模块	2	2.6.4 间距和缩进	22
§ 2 页面布局, layout 模块	3	2.6.5 杂项	23
2.2.1 页面尺寸	4	2.6.6 目录	23
2.2.2 主体尺寸	5	§ 7 buffer 模块	24
2.2.3 边距	7	第三章 编程接口	24
2.2.4 原有长度	7	§ 1 L ^A T _E X 2 _ε 的钩子机制	24
2.2.5 页眉页脚	8	§ 2 util 模块	25
2.2.6 杂项	9	3.2.1 psr, 处理器	28
2.2.7 设置页眉页脚	9	§ 3 struct 模块	29
§ 3 盒子和填充, box 模块	10	§ 4 L ^A T _E X 2 _ε 的 mark 机制	31
2.3.1 Framed	10	第四章 库的文档接口	32
2.3.2 Filler	11	§ 1 doc 库	32
2.3.3 多栏文字	15	§ 2 bnf 库	35
2.3.4 旋转的盒子	17	§ 3 ref 库	37
§ 4 背景, bgfg 模块	17	索引	39
§ 5 索引, index 模块	18	代码索引	39
§ 6 文档结构, struct 模块	20		

CuS_TE_X

第一章 概述

目前 CuSTeX 还处于早期的开发状态中，很多功能还并不完善。

CuSTeX (CuSLaTeX) 宏集意为 Chinese User Scheme TeX (LaTeX)，为中文 LaTeX 用户定制的文档类框架。CuSTeX 支持 XeLaTeX、LuaLaTeX、upLaTeX、ApLaTeX (pLaTeX-ng) 等多种编译方式，其中 LuaLaTeX、upLaTeX、ApLaTeX 还支持竖排。

使用 CuSTeX 可以方便地设置标题、目录、页面样式 (页面几何元素、页眉页脚等)、图表、背景、水印、边注、脚注、列表、索引、术语表等文档元素，具有强大的可定制性。CuSTeX 原生兼容 pgf 和 tcolorbox，加载这两个宏包或使用 pgf 库可实现更多的功能 [TODO]。

CuSTeX 通过模块 (module) 和库 (library) 来实现诸多功能。其中模块是核心部分，CuSTeX 将自动加载它们；库是提供额外功能的，用户可以选择是否加载它们。库可能依赖其它模块和库，但模块不会依赖库。

模块和库均可能加载其它宏包，一般情况下，CuSTeX 会自动加载这些模块并处理好它们的依赖和兼容性，当用户需要加载其它宏包时，最好通过 CuSTeX 的宏包加载机制来加载它们 [TODO]。

CuSTeX 还很好的支持和适配了通用驱动 (generic driver)，这是 LaTeX 2_ε2022-06-01 中的新功能。

不兼容 beamer。

第二章 文档接口

CuSTeX 定义的命令有的用于文档中，有的则是面向开发者，本章描述那些在文档中可能使用到的接口。

Logo。输出 CuSTeX，CuSLaTeX。

\CuSTeX
\CuSLaTeX

\cussetup

```
\cussetup {⟨key-vals⟩}
\cussetup [⟨key path⟩] {⟨key-vals⟩}
\cussetup {
  ⟨key path1⟩ = {⟨key-vals1⟩} ,
  ⟨key path2⟩ = {⟨key-vals2⟩} ,
  ...
}
```

键值设置命令。

CuSTeX 的不同模块使用不同的 ⟨key path⟩，一般情况下，这些模块会提供自己的键值设置接口，为了使用 \cussetup 来设置这些键值，需要指定 ⟨key path⟩。

```
\cussetstyle [⟨key path⟩] {⟨key⟩} {⟨key-vals⟩}
\cussetstyle * [⟨key path⟩] {⟨key⟩} {⟨code⟩}
```

\cussetstyle

自定义键。

带 * 的可使用一个参数，它代表键传入的值。

§ 1 util 模块

util 模块

<code>\Replicate</code> *	<code>\Replicate {<num expr>} {<code>}</code>
---------------------------	---

重复 `<code>` `<num expr>` 次。

<code>\MapClist</code> ☆	<code>\MapClist {<comma list>} {<tokens>}</code>
<code>\MapList</code> ☆	<code>\MapList {<list>} {<tokens>}</code>

`\MapClist` 使用 `<tokens>` 迭代逗号分隔的列表 `<comma list>`，它将 `<tokens>` 置于列表项之前。

`\MapList` 使用 `<tokens>` 迭代记号列表 `<list>`，它将 `<tokens>` 置于列表项之前。

<code>\IterateClist</code>	<code>\IterateClist {<comma list>} {<inline code>}</code>
<code>\IterateList</code>	<code>\IterateList {<list>} {<inline code>}</code>

`\IterateClist` 使用 `<inline code>` 迭代逗号分隔的列表 `<comma list>`，`<inline code>` 可带一个参数 `#1`，它为当前迭代项。

`\IterateList` 使用 `<inline code>` 迭代记号列表 `<list>`，`<inline code>` 可带一个参数 `#1`，它为当前迭代项。

$\begin{array}{c} \$ \backslash \text{MapClist}\{1,2,3,n\}\{a_ \} \$ \backslash \text{quad} \$ \backslash \text{IterateClist}\{1,2,3\}\{a_ \{ \#1 \} + \} a_n \$ \\ \dots\dots\dots \\ a_1 a_2 a_3 a_n \quad a_1 + a_2 + a_3 + a_n \end{array}$	例 1
--	------------

<code>\IterateThread</code>	<code>{<comma list₁>} {<comma list₂>} {<inline code>}</code>
<code>\IterateThread *</code>	<code>{<comma list₁>} {<comma list₂>} {<inline code>}</code>
<code>\IterateThread</code>	<code>[<n>] {<comma list₁>} ... {<comma list_n>} {<inline code>}</code>
<code>\IterateThread *</code>	<code>[<n>] {<comma list₁>} ... {<comma list_n>} {<inline code>}</code>
<code>\IterateThread</code>	<code>[<n>] {<comma list₁>} ... {<comma list_n>} {<middle>} {<inline code>}</code>
<code>\IterateThread</code>	<code>[<n>] {<comma list₁>} ... {<comma list_n>} {<middle>} [<last>] {<inline code>}</code>
<code>\IterateThread *</code>	<code>[<n>] {<comma list₁>} ... {<comma list_n>} {<middle>} [<last>] {<inline code>}</code>

使用 `<inline code>` 迭代这 n 个 `<comma list>`，`<inline code>` 可接受 $n + 1$ 个参数，其中第一个参数为索引，其后的参数分别为诸列表的当前迭代项。当某一个列表结束时迭代终止，多余的项被移除。 n 的可选值为 1–7，即最多可使用 7 个列表。

使用 `<middle>` 来分隔各项，最后两项用 `<last>` 分隔，默认与 `<middle>` 一致。如未给出，则为空，即不在两项之间插入其它符号。

带 `*` 的版本保留空项和每项前后的空格，不带 `*` 的则不保留。

若某个 `<comma list>` 为单个记号，则将其展开一次。这样，可以使用一个宏保存列表项。

$\begin{array}{c} \$ \backslash \text{IterateThread}\{a+b,c+d,e+f\}\{A+B,C+D,E+F\}\{\dfrac{\#2}{\#3}\geq\} 0 \$ \backslash \text{par} \\ \$ \backslash \text{IterateThread} \{a+b, ,e+f\}\{A+B,C+D, \}\{\dfrac{\#2}{\#3}\geq\} 0 \$ \backslash \text{par} \\ \$ \backslash \text{IterateThread} *\{a+b, ,e+f\}\{A+B,C+D, \}\{\dfrac{\#2}{\#3}\geq\} 0 \$ \backslash \text{par} \\ \dots\dots\dots \\ \frac{a+b}{A+B} \geq \frac{c+d}{C+D} \geq \frac{e+f}{E+F} \geq 0 \end{array}$	例 2
--	------------

$$\frac{a+b}{A+B} \geq \frac{e+f}{C+D} \geq 0$$

$$\frac{a+b}{A+B} \geq \frac{e+f}{C+D} \geq 0$$

```
$ \IterateThread[2]{1,2,3,n}{n,n-1,n-2,1}[+][+\cdots+]{a_{#2}\cdot b^{#3}}$
%= $ a_1\cdot b^{n+a_2}\cdot b^{n-1}+a_3\cdot b^{n-2}+\cdots+a_n\cdot b^1 $
.....
a_1 \cdot b^n + a_2 \cdot b^{n-1} + a_3 \cdot b^{n-2} + \cdots + a_n \cdot b^1
```

```
\ucchar {<unicode slot>}
\ucchars {<unicode slots>}
```

```
\ucchar ☆
\ucchars ☆
```

展开为 `<unicode slot>` 对应的 Unicode 字符。`<unicode slots>` 为空格分隔的 Unicode 代码点。

```
\ucchar{"5982}: %
\ucchars{"75 "74 "69 "6C "6A21 "5757}。
.....
如: util 模块。
```

例 4

相当于 `\kern\z@`。

```
\zkern
```

```
\Verbatimize {<balanced tokens>}
\Verbatimize * <token> <tokens> <token>
```

```
\Verbatimize
```

以 `verbatim` 的形式输出 `<balanced tokens>` 或 `<tokens>`。

带 `*` 的版本作用与 `\verb` 类似, 由一对 `<token>` 包裹, 也支持一对 `{ }` 包裹。只是它仍然使用当前字体。不能作为一个命令的参数。

不带 `*` 的版本可以作为另一个命令的参数, 但如下几个字符必须使用转义的形式: `##%` `{}` `\`, 即, 使用 `\#\$\%\ \{\}`。

```
\ifPageOdd {<true>} {<false>}
```

```
\ifPageOdd
\ifAbsPageOdd
```

判断当前页码是否为奇数。`\ifAbsPageOdd` 仅在 `shipout` 时有效 (如在 `shipout/foreground`, `shipout/background`, `shipout/after` 钩子中)。

平常使用时并不一定准确, `ref` 库改进了这一点, 见第 4.3 节。

§ 2 页面布局, layout 模块

layout 提供页面布局的相关接口。

```
\setuplayout {<layout key-val>}
\setuplayout [(<preset name>)] {<layout key-val>}
\setuplayout * [(<preset name>)] {<layout key-val>}
```

```
\setuplayout
```

设置布局。

第一个用法为直接设置页面布局。第二个除了设置布局外, 还将这个布局保存下来, 可供后续重复使用。第三个则仅保存布局, 而不设置这个布局。

可以在文档中间改变布局, 纸张大小也可改变。

键值接口大都直接使用 `geometry` 宏包的接口。具体用法说明可参见其说明文档。如未作说明, 则与 `geometry` 宏包提供的接口用法相同。

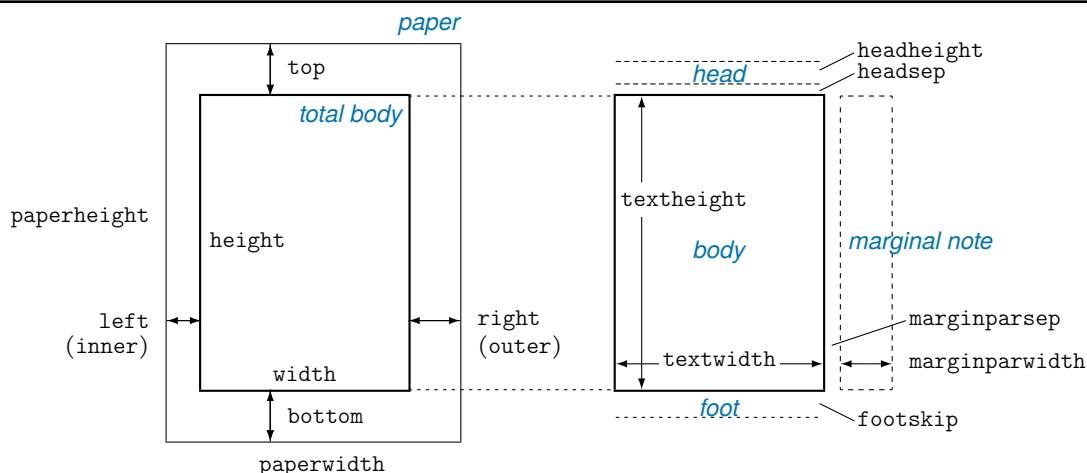


图 2.1: 长度变量

2.2.1 页面尺寸

layout/papername layout/paper	papername paper = {<papername>}
	设置纸张大小。<papername> 为预定义的纸张名，大小写无关。
layout/papersize layout/paperwidth layout/paperheight	papersize = {<宽>,<高>} 或 {<宽>:<高>} 或 {<长度>} paperwidth = {<宽>} paperheight = {<高>}
	设置纸张大小。

名称	宽 × 高	名称	宽 × 高	名称	宽 × 高
A0	841mm × 1189mm	B0	1000mm × 1414mm	C0	917mm × 1297mm
A1	594mm × 841mm	B1	707mm × 1000mm	C1	648mm × 917mm
A2	420mm × 594mm	B2	500mm × 707mm	C2	458mm × 648mm
A3	297mm × 420mm	B3	353mm × 500mm	C3	324mm × 458mm
A4	210mm × 297mm	B4	250mm × 353mm	C4	229mm × 324mm
A5	148mm × 210mm	B5	176mm × 250mm	C5	162mm × 229mm
A6	105mm × 148mm	B6	125mm × 176mm	C6	114mm × 162mm
b0j	1030mm × 1456mm	n0kai	787mm × 1092mm	b0kai	889mm × 1194mm
b1j	728mm × 1030mm	n2kai	787mm × 546mm	b2kai	889mm × 597mm
b2j	515mm × 728mm	n4kai	389mm × 546mm	b4kai	444mm × 597mm
b3j	364mm × 515mm	n6kai	370mm × 520mm	b8kai	420mm × 285mm
b4j	257mm × 364mm	n8kai	260mm × 370mm	b16kai	210mm × 285mm
b5j	182mm × 257mm	n16kai	185mm × 260mm	b32kai	142mm × 210mm
b6j	128mm × 182mm	n32kai	185mm × 130mm	screen	225mm × 180mm
ANSIA	8.5in × 11in	ANSID	22in × 34in	letter	8.5in × 11in
ANSIC	17in × 22in	ANSIE	34in × 44in	legal	8.5in × 14in
ANSIB	11in × 17in			executive	7.25in × 10.5in

表 2.1: 预定义的纸张名


```
paperorientation|orientation = <landscape|portrait>
landscape
portrait
direction = <bigwidth|bigheight|normal|inverse>
```

不可设置值
不可设置值

```
layout/paperorientation
layout/orientation
layout/landscape
layout/portrait
layout/direction
```

设置纸张方向。使用 portrait 时, 纸张高度大于宽度。landscape 则反之。

direction 的 bigheight 和 normal 相当于 portrait, bigwidth 和 inverse 相当于 landscape。

使用 papername 等选项时, 将自动设置纸张方向, 使得实际纸张宽高与所给一致。

设置 *layout* 部分大小。

见 geometry 宏包文档。

```
layout/layout
layout/layoutname
layout/layoutwidth
layout/layoutheight
layout/layoutsize
layout/layoutoffset
layout/layoutoffset
```

2.2.2 主体尺寸

此小节与 geometry 对应部分的用法和作用相同。

```
hscale = <{正实数}>
vscale = <{正实数}>
scale = <{hscale},<vscale}> 或 <{正实数}>
```

初始值: **0.7**
初始值: **0.7**

```
layout/hscale
layout/vscale
layout/scale
```

设置 *total part* 部分的宽高与纸张宽高的比率。

```
totalwidth |width = <{长度}>
totalheight|height = <{长度}>
total = <{totalwidth},<totalheight}> 或 <{长度}>
```

```
layout/totalwidth
layout/width
layout/totalheight
layout/height
layout/total
```

设置 *total part* 部分的宽高。

```
textwidth = <{长度}>
textheight = <{长度}>
body = <{textwidth},<textheight}>
text = <{长度}>
```

```
layout/textwidth
layout/textheight
layout/body
layout/text
```

设置 \textwidth、\textheight, 即 *body* 部分的宽高。

```
lines = <{行数}>
```

```
layout/lines
```

根据 <行数> 设置 textheight。<行数> 一般为正整数。

```
includehead = <true|false>
includefoot = <true|false>
includeheadfoot|includehf = <true|false>
```

初始值: **false**
初始值: **false**

```
layout/includehead
layout/includefoot
layout/includeheadfoot
layout/includehf
```

控制是否将页眉 (\headheight、\headsep)、页脚 (\footskip) 计入 *total part* 部分中。

```
includemarginpar|includemp = <true|false>
```

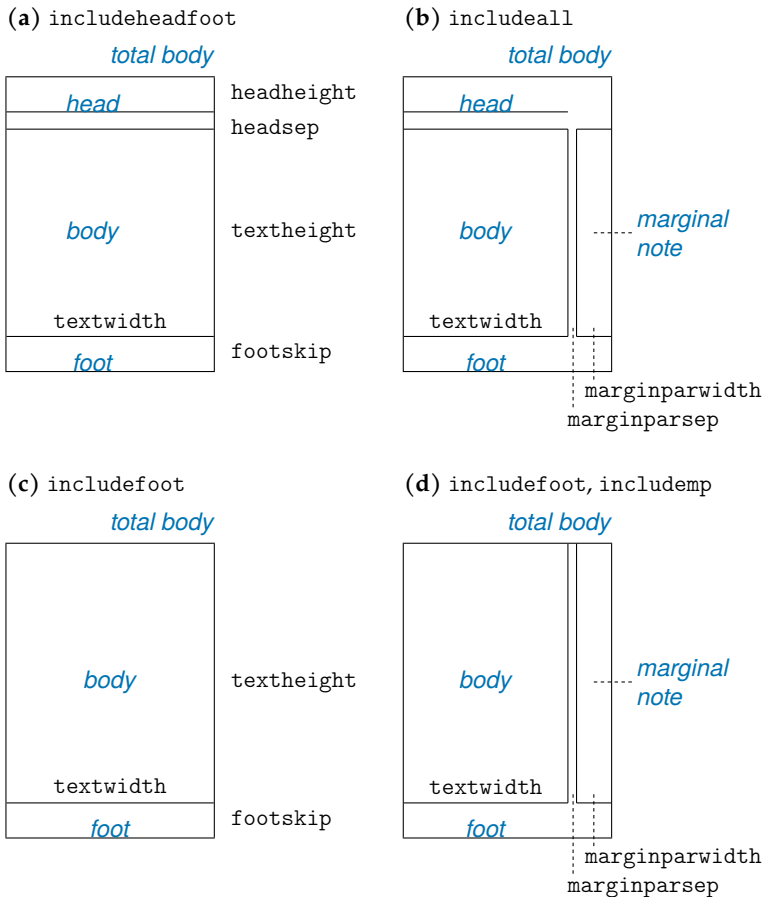
初始值: **false**

```
layout/includemarginpar
layout/includemp
```

控制是否将旁注 (\marginparwidth、\marginparsep) 计入 *body* 部分中。

<div>layout/includeall</div>	<div>includeall = $\langle \text{true} \text{false} \rangle$</div> <div>初始值: false</div> <div>设置 includeheadfoot 及 includemarginpar.</div>
<div>layout/ignorehead</div> <div>layout/ignorefoot</div> <div>layout/ignoreheadfoot</div> <div>layout/ignorehf</div>	<div>ignorehead = $\langle \text{true} \text{false} \rangle$</div> <div>ignorefoot = $\langle \text{true} \text{false} \rangle$</div> <div>ignoreheadfoot ignorehf = $\langle \text{true} \text{false} \rangle$</div> <div>初始值: false</div> <div>初始值: false</div> <div>在计算垂直方向的尺寸时, 不考虑页眉、页脚。但不修改页眉页脚的尺寸。</div>
<div>layout/ignoremarginpar</div> <div>layout/ignoremp</div>	<div>ignoremarginpar ignoremp = $\langle \text{true} \text{false} \rangle$</div> <div>初始值: false</div> <div>在计算水平方向的尺寸时, 不考虑旁注的尺寸。但不修改旁注的尺寸。</div>
<div>layout/ignoreall</div>	<div>ignoreall = $\langle \text{true} \text{false} \rangle$</div> <div>初始值: false</div> <div>设置 ignoreheadfoot 及 ignoremarginpar.</div>
<div>layout/heightrounded</div>	<div>heightrounded = $\langle \text{true} \text{false} \rangle$</div> <div>初始值: false</div> <div>如果设置为真, 则将 textheight 设置为不小于原 textheight 且满足关系: $n \times \backslash \text{baselineskip} + \backslash \text{topskip}$的最小值。</div>

图 2.2
不同模式下的 *total body*。(a) includeheadfoot, (b) includeall, (c) includefoot 及 (d) includefoot, includemp。如果 reversemarginpar 设置为真, 则交换 *marginal note* 与 *body* 的位置。如果设置了 twoside, 则依据奇偶页交换 *marginal note*。



```
hdivide = {\left margin},\langle width\rangle,\langle right margin\rangle\}
vdivide = {\top margin},\langle height\rangle,\langle bottom margin\rangle\}
divide = {\langle length_1\rangle},\langle length_2\rangle,\langle length_3\rangle\}
```

设置两个值, 将另一个留空或*。

```
layout/hdivide
layout/vdivide
layout/divide
```

2.2.3 边距

```
lmargin|leftmargin |left |inner = {\langle 内侧边距\rangle}
rmargin|rightmargin|right|outer = {\langle 外侧边距\rangle}
hmargin|horizontalmargin = {\langle inner\rangle},\langle outer\rangle\} 或 {\langle 水平边距\rangle}
```

设置内外侧边距。注意, 不论是否使用 twoside, 它们的含义都是相同的。

```
layout/leftmargin
layout/left
layout/lmargin
layout/inner
layout/rightmargin
layout/right
layout/rmargin
layout/outer
layout/hmargin
layout/horizontalmargin
```

```
tmargin|topmargin |top = {\langle 顶部边距\rangle}
bmargin|bottommargin|bottom = {\langle 底部边距\rangle}
vmargin|verticalmargin = {\langle top\rangle},\langle bottom\rangle\} 或 {\langle 垂直边距\rangle}
```

设置上下边距。

```
layout/topmargin
layout/top
layout/tmargin
layout/bottommargin
layout/bottom
layout/bmargin
layout/verticalmargin
```

```
hmarginratio|horizontalmarginratio = {\langle inner ration\rangle}:\langle outer ratio\rangle\}
vmarginratio|verticalmarginratio = {\langle top ratio\rangle}:\langle bottom ratio\rangle\} 初始值: 2:3
marginratio = {\langle hmargin ratio\rangle},\langle vmargin ratio\rangle\} 或 {\langle margin ratio\rangle}
```

设置内外边距、上下边距的比率。

使用 oneside 时 hmarginratio 初始为 1:1, 使用 twoside 时 hmarginratio 初始为 2:3。

```
layout/horizontalmarginratio
layout/hmarginratio
layout/verticalmarginratio
layout/vmarginratio
layout/marginratio
```

```
hcentering = \langle true|false\rangle
vcentering = \langle true|false\rangle 初始值: false
centering = \langle true|false\rangle
```

设置 hmarginratio、vmarginratio 为 1:1。

```
layout/hcentering
layout/vcentering
layout/centering
```

```
twoside 不可设置值
asymmetric 不可设置值
reversemarginpar|reversemp = \langle true|false\rangle 初始值: false
```

设置左右边距根据奇偶页进行切换。asymmetric 并不实际切换, 而是修改长度, 见 geometry 宏包文档。

```
layout/twoside
layout/asymmetric
layout/reversemarginpar
layout/reversemp
```

```
bindingoffset = {\langle 长度\rangle}
```

从内侧移除 \langle 长度 \rangle 。

```
layout/bindingoffset
```

2.2.4 原有长度

本小节描述几个 L^AT_EX 2_ε 原有的长度变量。

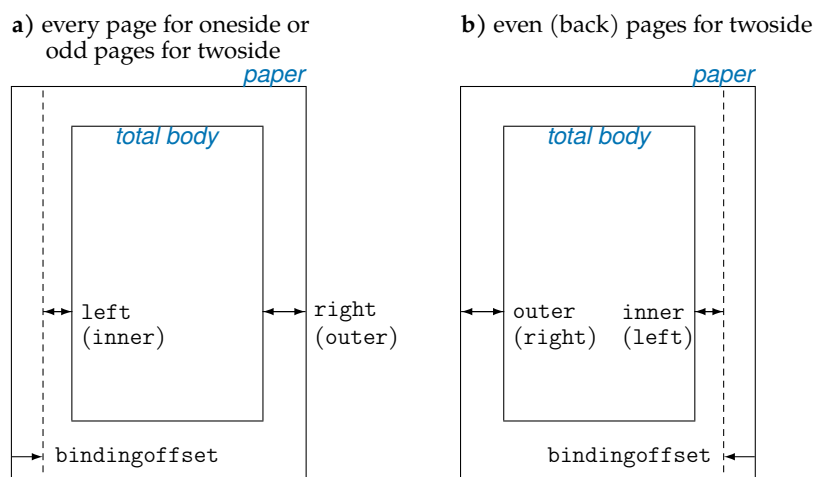
```
footnotesep = {\langle 弹性长度\rangle}
```

设置 \skip\footins, 即正文底部与脚注顶部的距离。

```
layout/footnotesep
```

图 2.3

The option `bindingoffset` adds the specified length to the inner margin. Note that `twoside` option swaps the horizontal margins and the marginal notes together with `bindingoffset` on even pages (see **b**), but `asymmetric` option suppresses the swap of the margins and marginal notes (but `bindingoffset` is still swapped).



`layout/marginparwidth`
`layout/marginpar`
`layout/marginparsep`
`layout/nomarginpar`
`layout/nomp`

`marginparwidth|marginpar` = {<长度>}
`marginparsep` = {<长度>}
`nomarginpar|nomp`

不可设置值

设置旁注宽度及旁注与正文的距离。`nomarginpar` 将它们设置为 0pt。

`layout/columnsep`
`layout/twocolumn`
`layout/onecolumn`

`columnsep` = {<长度>}
`twocolumn`
`onecolumn`

不可设置值

不可设置值

设置 `\columnsep`, 即两栏之间的距离。

`layout/hoffset`
`layout/voffset`
`layout/offset`

`hoffset` = {<长度>}
`voffset` = {<长度>}
`offset` = {<hoffset>,<voffset>} 或 {<长度>}

设置 `\hoffset`、`\voffset`。

2.2.5 页眉页脚

`layout/headheight`
`layout/head`
`layout/headsep`

`head|headheight` = {<长度>}
`headsep` = {<长度>}
`nohead`

`headheight` 设置 `\headheight`, 即页眉的高度。

`headsep` 设置 `\headsep`, 即页眉与正文之间的距离。

`nohead` 将它们设置为 0pt。

`layout/footskip`
`layout/foot`
`layout/nofoot`

`footskip|foot` = {<弹性长度>}

设置 `\footskip`, 即正文最后一行的基线与页脚基线的距离。

`nofoot` 将它设置为 0pt。

`layout/noheadfoot`
`layout/nohf`

`noheadfoot|nohf`

不可设置值

同时设置 `nohead` 和 `nofoot`

`layout/headoffset`
`layout/footoffset`
`layout/hfoffset`

`headoffset` = {<长度>}
`headoffset` = [<位置>] {<长度>}

初始值: 0pt

设置页眉页脚偏移量。

<位置> 为 O、E 与 L、C、R 的组合。这五个值分别代表奇偶、左中右。不区分大小写。

若 <长度> 为正值, 则相较于 `textwidth` 伸长 <长度>。否则, 缩短 <长度>。

此选项在直排文档中可能无效。

2.2.6 杂项

本小节列出其它几个选项。未列出的选项请参考 `geometry` 宏包文档。

<code>showframe = {true false}</code>	初始值: <code>false</code>	<code>layout/showframe</code>
<code>showcrop = {true false}</code>	初始值: <code>false</code>	<code>layout/showmarking</code>
<code>showmarking marking = {true false}</code>	初始值: <code>false</code>	<code>layout/marking</code>

`showframe` 显示各部分的外框。`showcrop` 在 `layout` 四角显示裁剪标记。`marking` 在各部分着以彩色背景。

<code>preset name = {<preset name>}</code>	<code>layout/preset</code>
使用预设值 <code><preset name></code> 。	<code>layout/name</code>

2.2.7 设置页眉页脚

本小节设置页眉页脚内容的接口。关于设置页眉页脚位置和高度的接口, 见第 2.2.5 小节。

本节所述内容可能在直排文档中不可用。

本节所述的功能主要通过 `fancyhdr` 实现。

<code>\usepagestyle {<pagestyle>}</code>	<code>\usepagestyle</code>
--	----------------------------

使用页眉页脚的样式 `<pagestyle>`。

有一个预定义的样式 `totalempy`, 它将页眉页脚设置为空, 并将页眉页脚横线的厚度设为 0pt。

<code>\setpagestyle {<pagestyle>} {<code>}</code>	<code>\setpagestyle</code>
<code>\setpagestyle {<pagestyle_1>} [<pagestyle_2>] {<code>}</code>	
<code>\setpagestyle * {<pagestyle_1>} [<pagestyle_2>] {<code>}</code>	

设置样式 `{<pagestyle>}`, 或基于样式 `<pagestyle_2>` 设置 `<pagestyle_1>`。

带 * 的, 仅设置而不使用。不带 * 的, 还会立刻使用该样式。

<code>\sethead {<code>}</code>	<code>\sethead</code>
<code>\sethead [<位置>] {<code>}</code>	<code>\setfoot</code>
<code>\setcenterhead {<奇偶页>}</code>	<code>\setheadfoot</code>
<code>\setcenterhead [<偶数页>] {<奇数页>}</code>	<code>\setleftthead</code>
	<code>\setcenterthead</code>
	<code>\setrightthead</code>
	<code>\setlefttfoot</code>
	<code>\setcentertfoot</code>
	<code>\setrighttfoot</code>

设置页眉页脚的内容。

`<位置>` 为 O、E、L、C、R、H、F 此三类的组合。这七个值分别代表奇偶、左中右、页眉页脚。不区分大小写。

如某一类未给出, 则视为该类的全部值都给出。但 `\sethead`、`\setfoot` 分别为 H、F。

例如, 在 `\sethead` 中, L 代表 OLF, ELF。

它们可以直接用在导言区和正文中, 将修改本页及其后面页面的页眉页脚。但最好用于 `\setpagestyle` 命令中, 统一设置页眉页脚。

<code>\setheadrulewidth {<长度表达式>}</code>	<code>\setheadrulewidth</code>
	<code>\setfootrulewidth</code>

设置页眉、页脚横线的厚度。

(即宏 `\headrulewidth`、`\footrulewidth` 的值。)


```
frame = {\code}
frame* = {\code width 1 parameter}
```

frame 设置盒子外框。

first, middle, last 设置分页盒子的前、中、后三部分的外框。

whole 设置未分页盒子的外框。

`\code` 其后可接一个参数, 这个参数为分页后的盒子。`\code with 1 parameter` 显式给出变量 #1。

```
init = {\code}
```

盒子中执行的初始化代码。

```
width = {\长度表达式}          初始值: \textwidth
ratio = {\数值表达式}          初始值: 1
```

ratio 设置盒子内容 (含边框) 占 width 的比率。

```
align = (left|center|right|inner|outer)  初始值: center
```

设置水平对齐方式。当 $0 < (ratio) < 1$ 时才有效。

```
frame/frame
frame/frame*
frame/first
frame/first*
frame/middle
frame/middle*
frame/last
frame/last*
frame/whole
frame/whole*
```

```
frame/init
```

```
frame/width
frame/ratio
```

```
frame/align
frame/left
frame/center
frame/right
frame/inner
frame/outer
```

```
\begin{Framed}[ratio=.8,center,
  rule-width=2pt,
  frame={\setlength{\fboxsep}{\cusframesep}%
        \setlength{\fboxrule}{\cusframerule}%
        \fcolorbox{purple}{cyan!50}}]
\zhlipsum[9][name=zhufu]
\end{Framed}
```

例 6

我很悚然, 一见她的眼钉着我的, 背上也就遭了芒刺一般, 比在学校里遇到不及豫防的临时考, 教师又偏是站在身旁的时候, 惶急得多了。对于魂灵的有无, 我自己是向来毫不介意的; 但在此刻, 怎样回答她好呢? 我在极短期的踌躇中, 想, 这里的人照例相信鬼, 然而她, 却疑惑了, ——或者不如说希望: 希望其有, 又希望其无……, 人何必增添末路的人的苦恼, 一为她起见, 不如说有罢。

```
ignore-warnings          不可设置值
```

忽略部分警告。

```
frame/ignore-warnings
```

2.3.2 Filler

“filler” 是用以填充空间的那部分内容。如 \LaTeX 2_ϵ 的 `\hrulefill` 是用水平直线填充, `\dotfill` 是用句点填充, `\hspace` 是用空白填充。

box 提供了几个创建 filler 的命令。

\dashfiller

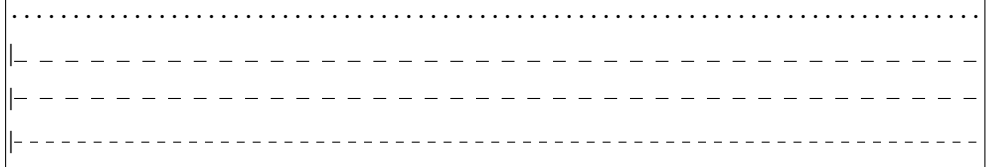
```
\dashfiller
\dashfiller {<filler width>}
\dashfiller [<raise>] [<sep width>] [<rule width>]
\dashfiller [<raise>] {<filler width>} [<sep width>] [<rule width>]
```

使用虚线填充, 虚线宽和虚线间的距离近似为 $\langle sep\ width \rangle$, 使得虚线充满 $\langle filler\ width \rangle$ 。

- $\langle filler\ width \rangle$ 为总宽度, 默认值为 $\backslash linewidth$ 。
- $\langle raise \rangle$ 为虚线升高的高度, 默认为 0pt。
- $\langle sep\ width \rangle$ 为虚线宽和虚线间的距离, 默认为 1ex。
- $\langle rule\ width \rangle$ 为虚线的厚度, 默认为 0.4pt。

```
\noindent\llap{\}\dashfiller \par % 总长为 \linewidth
\noindent\llap{\}\dashfiller [.5ex] \par % 升高 .5ex
% 升高 .5ex, 宽 3pt, 注意第二个可选参数前不能有空格
\noindent\llap{\}\dashfiller [.5ex][3pt] \par
```

例 7

\filler

```
\filler [<filler key-val>]
```

使用给定内容填充。

filler/size
filler/size*

```
size = {<skip expr>}
size* = {<长度>}
```

设置填充的长度。size* 填充的长度是弹性的。

注意在行间数学模式中 (equation、align 等环境) 弹性的那部分长度无效。

filler/space
filler/hspace
filler/hspace*
filler/vspace
filler/vspace*
filler/not-space

```
space = {<code>}
hspace = {<skip expr>}
hspace* = {<skip expr>}
vspace = {<skip expr>}
vspace* = {<skip expr>}
not-space
```

不可设置值

使用空白填充。使用它后, 其它选项无效。

$\langle code \rangle$ 是填充的内容, 如 $\backslash hspace\{1cm\}$, $\backslash vspace*\{1cm\}$ 。

hspace 相当于设置 $space=\backslash hspace\{<skip\ expr>\}$ 。

hspace* 相当于设置 $space=\backslash hspace*\{<skip\ expr>\}$ 。

vspace 相当于设置 $space=\backslash vspace\{<skip\ expr>\}$ 。

vspace* 相当于设置 $space=\backslash vspace*\{<skip\ expr>\}$ 。


由于用 space 填充的优先级最高, 若设置使用 space 填充后, 要使用其它类型来填充, 需使用 not-space 或将 space 设置为空。若后续仍设置了 space, 则仍会使用 space 填充。

```
左 \fbox{\strut \filler [hspace=5cm]} 右间隔约 5cm。
```

例 8

```
左 \filler[space] 右拉开。
```


左 `\filler[space]` 中 `\filler[space]` 右拉开。

左  右间隔约 5cm。
左 右拉开。
左 中 右拉开。

`color = {\color expr}`

`filler/color`

设置颜色 `cusfiller`, 即填充的颜色。

`content = {\content}`

`box = {\content}`

`box* = {\langle 长度表达式 \rangle {\content}}`

`clear-content`

不可设置值

`clear-box`

不可设置值

`not-content`

不可设置值

`filler/content`

`filler/box`

`filler/box*`

`filler/clear-box`

使用长 `\langle 长度表达式 \rangle` 的 `\content` 填充。

`content` 和 `box` 选项基本一致, 只是 `content` 会自动设置颜色, 而 `box` 则需使用 `\color` 或 `\color_select:n` 来设置颜色。

使用 `content` 将使用 `\content` 的自然宽度, 而 `box*` 则使用指定的宽度。

当设置了 `box` 或 `box*` 后, `content` 无效, 除非使用 `clear-box` 清除 `box`。

`dash|sep = {\langle dash length \rangle}`

初始值: 0pt

`filler/dash`

`rule = {\langle rule width \rangle}`

初始值: 0.4pt

`filler/sep`

`raise = {\langle raise height \rangle}`

初始值: 0pt

`filler/rule`

`full = {\true|false}`

初始值: false

`filler/raise`

`is-dash`

不可设置值

`filler/full`

`filler/is-dash`

使用虚线填充。

虚线宽和虚线间距为 `\langle dash length \rangle`, 厚度为 `\langle rule width \rangle`, 升高 `\langle raise height \rangle`。

若 `\langle dash length \rangle` 为 0pt, 则使用实线填充。

如果设置 `full` 为真, 则相当于 `\dashfiller`。

`solid`

不可设置值

`filler/solid`

`dashed = {\langle 长度 \rangle}`

`filler/dashed`

`dotted = {\langle 间距 \rangle}`

`filler/dotted`

`cdotted = {\langle 间距 \rangle}`

`filler/cdotted`

使用实线、虚线、句点或 `\cdot` 填充。

```
\def\BL{\noindent\llap{\}}%
\BL \filler[color=red, solid, rule=2pt] \par
\BL \filler[color=red, dashed, rule=0.5ex] \par
\BL \filler[color=red, dashed, rule=0.5ex, full] \par
\BL \filler[color=red, dotted] \par
\BL \filler[color=red, cdotted] \par
\BL \filler[color=red, cdotted=1cm, align] \par % 每个点都是对齐的
\BL \filler[color=red, cdotted=1cm, center] \par % 每个点的间距都是 1cm
\BL \filler[color=red, cdotted=1cm, spread] \par % 每个点的间距都相等, 可能超过1cm
```

例 9





filler/type	type = <align center spread>	初始值: align
filler/align	align	不可设置值
filler/center	center	不可设置值
filler/spread	spread	不可设置值

构造填充的方式。

- align: 每个同种 filler 都是无限长的对齐的填充中的一部分, 因此, 它们处处都是对齐的;
- center: 把用以填充的盒子紧挨着排列, 两头留下相等的空白;
- spread: 把多余的空白均匀地分布在盒子中间及两侧。

T_EXhackers note: 实际这三种方式分别对应 \leaders、\cleaders、\xleaders。

\atleastfiller	<pre>\atleastfiller <{dim expr}> \atleastfiller [<{filler key-val}>] <{dim expr}></pre>
----------------	---

填充的长度至少为 <dim epxr>, 太短的将自动断行。

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me.%
\atleastfiller[cdotted=1em]{5cm}断行。

我能吞下玻璃而不伤身体。 \atleastfiller[cdotted=1em]{5cm}不断。

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me.
. 断行。
我能吞下玻璃而不伤身体。 不断。

例 10

\breakablefiller	<pre>\breakablefiller \breakablefiller [<{filler key-val}>]</pre>
------------------	---

可自动断行的 filler。

\framebox[3cm]{可断} \breakablefiller[cdotted=1em] \framebox[3cm]{模式。 例 11
\framebox[7cm]{可断} \breakablefiller[cdotted=1em] \framebox[7cm]{模式。}

可断 模式。
可断
. 模式。

下例展示了制作多行填充的例子。

<pre>\newcommand\filllines[4][]{\% filler key-val, before, lines, after #2\filler[#1]% \Replicate{#3-1}{\break \rule{0pt}{0.7\baselineskip}\filler[#1]}}</pre>	例 12
--	------

```
#4\par}}
```

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me.

```
\filllines{\linespread{2}\selectfont}{3}{。 \hspace*{1em}}
```

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me.

```
\filllines[color=red,dotted]{\linespread{2}\selectfont}{3}{。 \hspace*{1em}}
```

% 整行

```
\filllines [raise=-.5ex]{\linespread{2}\selectfont \noindent\strut}{3}{
```

```
↪ 整行。 \hspace*{1em}}
```

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me. _____

我能吞下玻璃而不伤身体, I can eat glass, it dosen't hurt me.

_____ 整行。

2.3.3 多栏文字

CuTeX 中排版多栏文字有两种方式, 本节描述其中一种, 使用 `multicol` 宏包实现。

关于每个内部变量的详细用法, 可以参考 `multicol` 宏包文档。

```
\startmulticolumns [(multicolumns key-val)]  
  (content)  
\stopmulticolumns
```

```
\startmulticolumns  
\stopmulticolumns
```

将 `(content)` 以多栏排版。

`columns|cols = {⟨整数表达式⟩}`

初始值: 2

```
multicolumns/columns  
multicolumns/cols
```

设置多栏数。也可不必写出键名, 只写数字。可用的栏数为 1–20。

`outer-sep = {⟨skip expr⟩}`

初始值: 12.0pt plus 4.0pt minus 3.0pt

```
multicolumns/outer-sep
```


设置 `\multicolsep`, 即多栏文字与上下文的间距。

`column-sep|sep = {⟨length⟩}`

初始值: 10pt

```
multicolumns/column-sep  
multicolumns/sep
```

设置 `\columnsep`, 即多栏文字两栏的间隙。

<div> <div>multicolumns/first-minimal</div> <div>multicolumns/last-minimal</div> </div>	<div> <div>first-minimal = {\<pre length>}</div> <div>last-minimal = {\<post length>}</div> </div> <div> <div>初始值: 50pt</div> <div>初始值: 20pt</div> </div>
<p>如果多栏开始的那一页不足 $\langle pre\ length\rangle$, 则多栏将在新的一页开始。</p> <p>如果多栏结束的那一页不足 $\langle post\ length\rangle$, 则多栏将在新的一页结束。</p> <p><code>first-minimal</code> 设置 <code>\premulticols</code>, <code>last-minimal</code> 设置 <code>\postmulticols</code>。</p>	
<div> <div>multicolumns/heading</div> </div>	<div> <div>heading = {\<content>}</div> </div> <div> <p>设置在多栏文字之前的横跨所有栏的文字。可以使用 <code>\section</code> 等。它与其后的多栏文字保持在同一页。</p> </div>
<div> <div>multicolumns/rule</div> <div>multicolumns/rule-color</div> </div>	<div> <div>rule = {\<length>}</div> <div>rule-color = {\<color>}</div> <div>rule-color = [\<color mode>] {\<color>}</div> </div> <div> <div>初始值: 0pt</div> </div> <div> <p>设置 <code>\columnseprule</code>、<code>\columnseprulecolor</code>, 即多栏间竖线的宽度和颜色。</p> </div>
<div> <div>multicolumns/flush</div> <div>multicolumns/aligned</div> <div>multicolumns/ragged</div> <div>multicolumns/not-aligned</div> </div>	<div> <div>flush aligned</div> <div>ragged not-aligned</div> </div> <div> <div>不可设置值</div> <div>不可设置值</div> </div> <div> <p>控制多栏文字的尾部是否对齐。分别设使用 <code>\flushcolumns</code> 和 <code>\raggedcolumns</code>。</p> <p>初始为 <code>aligned</code>, 将使各栏头部和尾部的基线尽量对齐。</p> </div>
<div> <div>multicolumns/balanced</div> <div>multicolumns/not-balanced</div> </div>	<div> <div>balanced</div> <div>not-balanced</div> </div> <div> <div>不可设置值</div> <div>不可设置值</div> </div> <div> <p>在末页文字的处理上, 有两种方式, 一种为文字尽量往上排, 而将下方留空, 这也是默认的方式; 另一种为文字尽量往左排 (右排), 而将右边 (左边) 留空, 也就是将空白留在末尾的几栏上。前者为 <code>balanced</code>, 后者为 <code>not-balanced</code>。</p> </div>
<div> <div>multicolumns/columns*</div> <div>multicolumns/cols*</div> </div>	<div> <div>columns* cols* = {\<栏数>}</div> </div> <div> <p>它在设置栏数的同时还设置 <code>not-balanced</code>。</p> <p>注意 <code>columns</code> 并未决定使用 <code>balanced</code> 还是 <code>not-balanced</code>。</p> </div>
<div>  </div>	
<div> <div>multicolumns/addto-baselineskip</div> </div>	<div> <div>addto-baselineskip = {\<length>}</div> </div> <div> <p>设置 <code>\multicolbaselineskip</code>。</p> </div>
<div> <div>multicolumns/tolerance</div> <div>multicolumns/pretolerance</div> </div>	<div> <div>tolerance = {\<int expr>}</div> <div>pretolerance = {\<int expr>}</div> </div> <div> <div>初始值: 9999</div> </div> <div> <p>设置 <code>\multicoltolerance</code>、<code>\multicolpretolerance</code>。</p> </div>
<div> <div>multicolumns/collectmore</div> <div>multicolumns/minrows</div> <div>multicolumns/unbalance</div> <div>multicolumns/column-badness</div> <div>multicolumns/final-column-badness</div> </div>	<div> <div>collectmore = {\<int expr>}</div> </div> <div> <p>设置 <code>collectmore</code>, <code>minrows</code>, <code>unbalance</code>, <code>columnbadness</code>, <code>finalcolumnbadness</code> 计数器。</p> </div>
<div> <div>multicolumns/top-fuzz</div> <div>multicolumns/bottom-fuzz</div> </div>	<div> <div>top-fuzz = {\<dim expr>}</div> <div>bottom-fuzz = {\<dim expr>}</div> </div> <div> <div>初始值: 0pt</div> <div>初始值: 2pt</div> </div> <div> <p>设置 <code>\multicolovershoot</code>、<code>\multicolundershoot</code>。</p> </div>

```
v-fuzz = {\length}
```

v-fuzz 设置 top-fuzz 和 bottom-fuzz。h-fuzz 设置 \hfuzz。

```
multicolumns/v-fuzz
```

```
multicolumns/h-fuzz
```

```
overflow = {\dim expr}
```

初始值: 12pt

设置 \maxbalancingoverflow。

```
multicolumns/overflow
```

```
left-to-right|LR
```

不可设置值

```
right-to-left|RL
```

不可设置值

使用 \LRmulticolcolumns 或 \RLmulticolcolumns。默认为 left-to-right。

```
multicolumns/left-to-right
```

```
multicolumns/LR
```

```
multicolumns/right-to-left
```

```
multicolumns/RL
```

2.3.4 旋转的盒子

CuTeX 封装了 rotating 宏包, 提供了旋转的盒子。

```
\startrotate [\rotate key-val]
```

```
<content>
```

```
\stoprotate
```

```
\Rotate [\rotate key-val] {\content}
```

```
\startrotate
```

```
\stoprotate
```

```
\Rotate
```

将 <content> 旋转显示。

旋转的盒子有两种方式, 一种为保留旋转后的盒子的大小, 另一种设置旋转后的盒子大小为 0。

```
turn = {\number}
```

```
rotate|nospaceturn = {\number}
```

```
sideways
```

不可设置值

```
rotate/turn
```

```
rotate/nospaceturn
```

```
rotate/rotate
```

```
rotate/sideways
```

将盒子旋转 <number> 度。一般是逆时针旋转。

turn 使用第一种方式, rotate 使用第二种方式。sideways 相当于 turn=90。

\startrotate ... \stoprotate 默认使用 turn, \Rotate 默认使用 rotate。

```
float = {\float type}
```

```
float* = {\float type}
```

```
figure
```

不可设置值

```
figure*
```

不可设置值

```
rotate/float
```

```
rotate/float*
```

```
rotate/figure
```

```
rotate/figure*
```

```
rotate/table
```

```
rotate/table*
```

将 <content> 看作在浮动环境 <float type> 内, 并将其旋转 90 度。旋转后的内容占据一整个页面。

带 * 类似于带 * 的浮动环境。

也可不写出 float 或 float* 键名, 直接写 <float type> 或 <float type>*。

§ 4 背景, bgfg 模块

bgfg 是对 shipout 钩子的简单封装。

关于“钩子”机制, 第 3.1 节对其作了简单的介绍, 更详细的用法请参考: lthooks.pdf。

本手册前几页的水印使用如下代码实现:

```
\background + [./watermark]{%
  \rotatebox{45}{\color{gray!30}\fontsize{100}{0}%
    \sffamily \CuTeX}}
```

例 13

```
% 使用如下代码即可删除此背景
\removebackground[./watermark]
```

```
\foreground
\background
```

```
\foreground    {<content>}
\foreground +  {<content>}
\foreground (<位置>) {<content>}
\foreground [<hook label>] {<content>}
\foreground + (<位置>) [<hook label>] {<content>}
```

将 $\langle content \rangle$ 放置在前景或背景中。

- $\langle content \rangle$ 为要放置的内容, 该内容将完整地嵌于页面内;
- $\langle 位置 \rangle$ 是 $\langle content \rangle$ 要放置的位置, 为两个字符, 前一个为水平位置; 后一个为垂直位置。水平位置包括: 左 (l)、右 (r)、内侧 (i)、外侧 (o); 垂直位置包括: 顶部 (t)、底部 (b); 它们的组合也就是 *layout* 的四个角。此外还有一个 cm, 它表示 *layout* 的正中心, 这也是默认值;
- $\langle hook label \rangle$ 为 hook 的 label; 此参数与 $\langle 位置 \rangle$ 的先后位置可交换;
 $\backslash foreground$ 默认为 ./fg, $\backslash background$ 默认为 ./bg;
 关于 hook label 的作用, 请参考第 3.1 节或 lthooks.pdf;
- 默认情况下 $\langle content \rangle$ 仅添加到当前页, 使用 + 可将 $\langle content \rangle$ 添加到往后各页。

除了上述两个命令外, 还提供了两个设置背景图片的命令。

```
\foregroundpicture
\backgroundpicture
```

```
\foregroundpicture    {<图片>}
\foregroundpicture +  {<图片>}
\foregroundpicture *  {<图片>}
\foregroundpicture + * {<图片>}
\foregroundpicture + * (<位置>) [<hook label>] {<图片>}
```

将 $\langle 图片 \rangle$ 添加到前景或背景中。

+, $\langle 位置 \rangle$ 、 $\langle hook label \rangle$ 的用法如前所述。

不带 * 的图片被伸缩到与 *layout* 同宽高。而带星号的则仅缩放宽度, 保持纵横比例不变。

也可直接使用 $\backslash background$ 放置背景图片。

```
\background(ob){\includegraphics[width=\marginparwidth]{ctanlion.pdf}} 例 14
```

如本页底部图片所示。

```
\removeforeground
\removebackground
```

```
\removeforeground
\removeforeground [<hook label>]
```

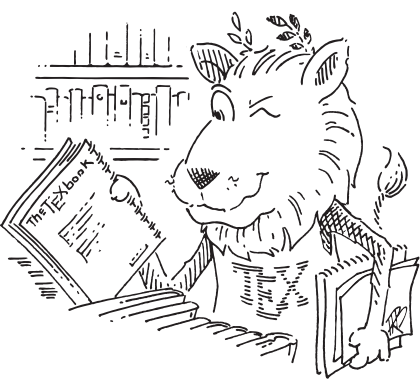
移除前景或背景。

§ 5 索引, index 模块

CuS_TE_X 提供了添加多个索引的方法。并且能够自动编译索引文件。

目前暂未提供 splitidx 宏包的功能, 也不与其兼容。

应该与 glossaries 宏包兼容。



```
\newindextype [<index keys>] {<index type>}
\setupindex   [<index type list>] {<index keys>}
```

```
\newindextype
\setupindex
```

`\newindextype` 创建一个新的索引 `<index type>`。`\setupindex` 配置 `<index type list>`。`<index type>` 可以使用 `\empty` 作为名称, 此时它的名称为空。

```
\makeindex
\makeindex [<index keys>]
\makeindex [<index keys>] [<index type>]
```

```
\makeindex
```

LaTeX 原有的接口。默认创建名称为空的索引。

`<index keys>` 不同于 CuTeX 中其它的键值选项, 仅具有类似的接口。

- `filename`: 索引文件名, 一般以 `idx` 结尾, 如果未设置, 则为:
`\jobname<index type>.idx`;
- `output`: 编译后的索引文件, 一般以 `ind` 结尾, 如果未设置, 则将 `filename` 的后缀修改为 `ind` 作为输出文件名;
- `name`: 如果设置, 则应与 `<index type>` 一致;
- `title`: 索引的标题, 如 `\indexname`;
- `program`: 编译索引的可执行程序; 如 `makeindex`、`makeglossaries`;
- `options`: 编译索引时的选项, 索引文件名和输出文件名将自动添加;
- `exec`: 终端中实际编译索引执行的代码, 如果未设置, 则组合 `program` 及 `options`;
- `auto`: 布尔值, 是否自动编译索引文件;
- `multi`: 多栏选项 (`<multicolumns key-val>`);
- `heading*`: 标题命令, 如 `\chapter[numbering=false]`;
- `init`: 索引开头的初始化设置; 索引文件不存在时不会执行;
- `prologue`: 索引开头的文字; 索引文件不存在时不会输出。

```
\newindextype[
  filename=\jobname.docusage.idx,
  output=\jobname.docusage.ind,
  exec={makeindex -s gind.list -o \jobname.docusage.ind
  ↪ \jobname.docusage.idx},
  title={代码索引}, heading*={\section},
  multi={2, ragged, sep=1em, outer-sep=0pt},
  auto=true
]{docusage}
```

例 15

```
\setindexinit {<code>}
\setindexinit [<index type>] {<code>}
\setindexprologue {<content>}
\setindexprologue [<index type>] {<content>}
```

```
\setindexinit
\setindexprologue
```

设置索引开头的初始化设置、设置索引开头的文字。只要使用了 `\printindex`, 当索引不存在时它们也会执行或输出。

默认设置名称为空的索引。

在初始化代码中还可以重定义索引环境 `theindex`。

```
\index
\printindex
```

```
\index {<index item>}
\index [<index type>] {<index item>}
\printindex
\printindex [<index keys>]
```

`\index` 添加索引项 `<index item>` 到索引 `<index type>` 中, 默认添加到名称为空的索引中; `\printindex` 输出索引, 以 `name` 键标识要输出的索引, 否则输出名称为空的索引。`<index keys>` 为前述的键。

§ 6 文档结构, **struct** 模块

struct 模块提供了创建目录和章节标题的方法。参考了 `titlesec`, `titletoc`, `ctex-heading`, `etoc` 等宏包的实现, 并自动阻止加载这些宏包。

章节标题样式的设置与 `ctexheading` 宏包也即 CTeX 文档类的接口基本一致, 但扩充了几个选项, 并且可以定义新的标题。

```
\definetttitle
```

```
\definetttitle {<title command>} {<title key-val>}
\definetttitle {<title command>} [<title class>] {<title key-val>}
```

定义新的章节标题命令 `<title command>`, 以 `<title class>` 作为基准类。

标题的使用方式见下方预定义的几个章节标题命令。

标准的 L^AT_EX book 类中, 章节标题可分为三种, 一种是以 `\part` 为代表的, CusTeX 将其命令为 `page` 类, 一种是以 `\chapter` 为代表的, CusTeX 将其命名为 `top` 类, 另一种则是以 `\section` 为代表, CusTeX 将其命名为 `normal` 类。¹

本模块预先定义了一些章节命令, 它们与 `ctexbook` 文档类的效果基本一致。

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

```
\part {<标题>}
\part * {<标题>}
\part [<list entry>] {<标题>}
\part [<title key-val>] {<标题>}
\part * [<title key-val>] {<标题>}
\part [<title key-val>] [<list entry>] {<标题>}
```

与标准的章节标题命令相比, 增加了 `<title key-val>` 可选项, 用于暂时修改样式。

`<标题>` 为实际显示的标题, `<list entry>` 为目录、页眉等内容中的文字, 它在带星号的命令中无效。

若要修改它们的样式, 一般仅需使用下述的 `\setuptitle` 修改键值选项, 而无需重新定义它们。

```
\setuptitle
```

```
\setuptitle {<title key-val>}
\setuptitle [<title list>] {<title key-val>}
```

设置标题的样式。`<title list>` 为章节标题命令名称的列表, 如: `[chapter, section]`, 而非标题命令的列表。

以下几节描述了章节标题中可用的键值选项, 它们均可以被设置, 但并不一定在所有的标题类中都有效。这里的 ... 代表各章节标题命令的名称。

¹实际上, 这些名称基本沿用了 `titlesec` 宏包的名称。

2.6.1 初始化设置

初始化设置选项可以在导言区修改任意次，但不可在正文设置。

```
number-from    = {(计数器)}
number-within  = {(计数器)}
number-without = {(计数器)}
```

```
title/.../number-from
title/.../number-within
title/.../number-without
```

设置章节命令的计数器。

`number-from` 设置章节命令所使用的计数器，默认为使用自己的计数器，这计数器与章节命令同名。

`number-within` 设置章节计数器随该计数器的递增而清零。`number-without` 取消对应的设置。

2.6.2 编号

```
\setsecnumdepth {(整数或层级名称)}
```

```
\setsecnumdepth
```

设置对章节标题进行编号的层次数。可以是整数或层级名称。

CuTeX 预先定义了一些层级名称，如表 2.2 所示。

```
numbering = {true|false}
```

初始值: **true**

```
title/.../numbering
```

控制是否对不带星号的命令编号。当此选项设置为 `false` 时，除了不带编号，其余功能均与正常标题一致。

注意，章节是否编号还受到 `secnumdepth` 计数器的控制，可以通过上述的 `\setsecnumdepth` 命令来设置。例如，对于 `\section` 而言，其默认的章节层级为 1（对于预定义的几个章节标题，其默认的章节层级与同名层级名称的层级相同，见表 2.2，可通过 `level` 键来修改层级）。因此 `\section` 会被编号当且仅当 `secnumdepth` 不小于 1，且其 `numbering` 键为 `true`，并且使用不带星号的章节命令。

表 2.2: 层级名称

层级	名称
-1	part
0	chapter
1	section
2	subsection
3	subsubsection
4	paragraph
5	subparagraph
4	sub3section
5	sub4section

```
name = {(名字的前半部分),(名字的后半部分)}
name = {(名字的前一部分)}
```

```
title/.../name
```

设置章节的名字。所谓“章节的名字”，可以分为前后两部分，即章节编号前后的词语，两个词之间用一个半角逗号分开；也可以只有一部分，表示只有章节编号之前的名字。

```
number = {(数字输出命令)}
```

```
title/.../number
```

设置章节编号的数字输出格式。(数字输出命令) 通常是对应章节编号计数器的输出命令，如 `\thesection` 或 `\zhnum{chapter}` 之类的。

`number` 选项定义的同时将控制对章节计数器的交叉引用。在引用计数器时，记录在 L^AT_EX 辅助文件中的是 `number` 选项的定义。但是，`number` 选项不会影响计数器本身的输出。

```
level = {(整数或层级名称)}
```

```
title/.../level
```

设置章节标题的层级。将影响是否对标题进行编号。

2.6.3 格式

和 CT_EX 宏集一样, CuST_EX 也提供了提供了 `numberformat`, `nameformat`, `titleformat`, `format` 这几个选项用来控制章节标题的格式。它们的作用范围如图 2.4 所示。

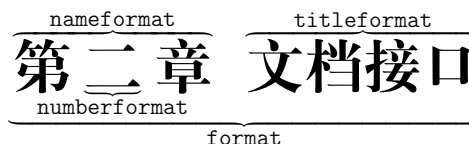


图 2.4: `numberformat`, `nameformat`, `titleformat`, `format` 几个选项的作用范围

```
title/.../format
title/.../format+
title/.../nameformat
title/.../nameformat+
title/.../numberformat
title/.../numberformat+
title/.../titleformat
title/.../titleformat+
```

```
format = {<格式代码>}
format+ = {<格式代码>}
```

设置相应部分的格式。带 + 的用于在原有的格式后增加代码。注意, + 与选项之间不能留有空白, 不能写成 `format += {..}`, 以下同。

```
title/.../aftername
title/.../aftername+
title/.../aftertitle
title/.../aftertitle+
```

```
aftername = {<code>}
aftername+ = {<code>}
```

用于将 `<code>` 插入到相应的部分之后。带 + 的用于在原有的格式后增加代码。

```
title/.../pagestyle
```

```
pagestyle = {<pagestyle>}
```

page (如 `\part`) 和 top (如 `\chapter`) 标题类还可以设置该标题所在页的页面样式。在 `normal` 标题类中无效。关于页面样式的相关内容, 见第 2.2.7 小节。

2.6.4 间距和缩进

```
title/.../runin
```

```
runin = <true|false>
```

初始值: `false`

用于指定是否是标题与其后的文字排在同一行。仅对 `normal` 类有效。

```
title/.../hang
```

```
hang = <true|false>
```

初始值: `false`

设置是否对章节标题实施悬挂缩进 (缩进的宽度为名字宽度和 `indent` 选项设置的宽度之和)。设置该选项为 `true` 时必须恰当地设置 `aftername` 选项。

若设置了 `runin` 为 `true`, 则该选项无意义。

```
title/.../indent
```

```
indent = {<缩进间距>}
```

设置章节标题本身的首行缩进。如果这缩进值是相对单位, 则缩进间距的大小是相对于 `format` 中指定的字号大小。

```
title/.../beforeskip
title/.../afterskip
```

```
beforeskip = {<skip expr>}
```

设置章节标题前后的垂直间距。若 `runin` 选项为 `true`, 则设置的是水平间距。

```
title/.../fixskip
```

```
fixskip = <true|false>
```

初始值: `false`

默认情况下, 章节标题除了由 `beforeskip` 和 `afterskip` 选项设置的垂直间距外, 还会有其它一些多余的间距, `fixskip` 用于指定是否抑制这些多余的间距。

```
break = {\code}
break+ = {\code}
```

```
title/.../break
title/.../break+
```

break 选项用于控制章节标题与之前正文的分隔关系。一般用于设置是否在标题之前分页或者设置行间罚点。

例如, 若当前页剩余高度小于正文高度的一半时, 则另起一页输出 \section 标题:

```
\usepackage{needspace}
\setuptitle [section] { break = \Needspace{0.5\textheight} }
```

例 16

```
afterindent = <true|false>
```

```
title/.../afterindent
```

afterindent 选项用于设置章节标题后首段的缩进。

2.6.5 杂项

```
tocline = {\<格式定义>}
```

```
title/.../tocline
```

定义章节标题在目录文件中的格式, <格式定义> 有两个参数: 参数 #1 是 part、chapter 等名字, 参数 #2 是标题内容。

初始值是: \titlenumberline{#1}#2。其含义为, 若标题没有名字, 则不输出编号。

```
mark = {\<mark code>}
```

```
title/.../mark
```

写入标记文本。<mark code> 其后跟一个参数, 在 \chapter 中, 为 \chaptermark, 在 \section 中, 为 \sectionmark。初始情况下, 不写入标记。

```
style = {\<style>}
```

```
title/.../style
```

设置已有的样式。

```
\titleifname {\<有名字时的内容>} {\<无名字时的内容>}
```

```
\titleifname
```

根据当前章节有无名字展开得到不同内容 (通常是格式命令)。

每一个标题都有一个对应的 \titlethe<title> 命令, 表示当前实际输出的章节标题的名字。如现在 \titlethechapter 为 “第二章”。

2.6.6 目录

CuS_TE_X 重新实现了目录的制作方式, 将每个目录项看成是一个个数据, 不同于标准的目录制作方式, 因此可能存在与其它宏包不兼容的情况。

在 CuS_TE_X 中, 目录被称为 “combined list”, 这是 ConT_EXt 中的称呼。

\enablecombinedlist 启用目录。可用于导言区或文档最开头。如果未使用此命令则其它目录命令不可用。

```
\enablecombinedlist
```

输出标准的目录、图片和表格目录。

```
\tableofcontents
\listoffigures
\listoftables
```

\standardplaincombinedlist
\multicolplaincombinedlist

\standardplaincombinedlist {<title>} {<cbl type>}
\multicolplaincombinedlist [<multicolumns key-val>] {<title>} {<cbl type>}

\standardplaincombinedlist 输出默认的目录, 该目录的类型是 <cbl type>, 并以 <title> 作为标题。如果 <title> 为空, 则不输出标题。

\multicolplaincombinedlist 输出多栏目录, 该目录的类型是 <cbl type>, 并以 <title> 作为标题。如果 <title> 为空, 则不输出标题。<multicolumns key-val> 设置多栏选项。

目前暂未提供修改目录样式的接口, 待后续版本更新。关于目录的内部数据结构, 见第 3.3 节。

§ 7 buffer 模块

未完成。[**TODO**]

第三章 编程接口

本章描述 CuS_TE_X 提供的编程接口。

\CUSProvideLibrary
\CUSProvideExplLibrary

\CUSProvideLibrary {<库名>} [<描述>]
\CUSProvideExplLibrary {<库名>} {<日期>} {<版本>} {<描述>}

提供库文件。库文件名必须是: “cus.library.<库名>.tex”。

\CUSDependency

\CUSDependency {<key-val>}

处理库依赖。

dependency/package
dependency/module
dependency/library
dependency/disable

package = {<comma list>}
module = {<comma list>}
library = {<comma list>}
disable = {<comma list>}

\CUSDependency 可用的键值选项。

\CUSLoadLibrary
\CUSIfLibraryAtLeast

\CUSLoadLibrary {<库名>} [<日期>]
\CUSIfLibraryAtLeast {<库名>} {<日期>} {<true code>} {<false code>}

§ 1 L^AT_EX 2_ε 的钩子机制

本节简述 L^AT_EX 2_ε 的钩子机制。更详细的说明见 lthooks.pdf。

“钩子” (hook) 是在命令或环境的定义中可以添加其它代码的位置。

\UseHook

\UseHook {<hook>}

在命令或环境中执行 <hook>。

```
\AddToHook {<hook>} [<label>] {<code>}
```

```
\AddToHooks
```

将 $\langle code \rangle$ 添加到 $\langle hook \rangle$ 中, 标记为 $\langle label \rangle$ 。 $\langle hook \rangle$ 可以未被定义。

如果未指定 $\langle label \rangle$, 则使用默认的 label。如果 $\backslash\text{AddToHook}$ 用在宏包或文档类中, 它就是宏包或文档类名, 否则, 它是 top-level。

```
\RemoveFromHook {<hook>} [<label>]
```

```
\RemoveFromHook
```

移除标记了 $\langle label \rangle$ 的 $\langle hook \rangle$ 。若 $\langle label \rangle$ 未指定, 则使用默认的 label。

```
\AddToHookNext {<hook>} {<code>}
\ClearHookNext {<hook>}
```

```
\AddToHookNext
\ClearHookNext
```

将 $\langle code \rangle$ 添加到 $\langle hook \rangle$ 的下一调用中。在正常的 $\langle hook \rangle$ 代码执行完毕后再执行 $\langle code \rangle$ 。

§ 2 util 模块

```
\getlabelinfo {<info>} {<label>}
```

```
\getlabelinfo
\cus_get_label:nn
```

获取 $\langle label \rangle$ 的相关信息。 $\langle label \rangle$ 可以为空, 代表最近写入的那个 label。

可获得的信息 $\{<info>\}$ 为:

- name, $\langle label \rangle$ 的值;
- page, 获得 $\langle label \rangle$ 所在页的 $\backslash\text{thepage}$ 值, 若 $\langle label \rangle$ 不存在则为 0;
- data, 获得 $\langle label \rangle$ 中的第一个数据项, 若 $\langle label \rangle$ 不存在则为 $\backslash\text{c_novalue_tl}$, 可使用 $\backslash\text{IfNoValueTF}$ 或 $\backslash\text{tl_if_novalue:nTF}$ 判断;
- anchor, 获得链接到 $\langle label \rangle$ 所在位置的锚点, 若 $\langle label \rangle$ 不存在或未加载 hyperref 则为 Doc-Start;
- pageanchor, 获得链接到 $\langle label \rangle$ 所在页的锚点, 若 $\langle label \rangle$ 不存在或未加载 hyperref 则为 Doc-Start。

```
\cus_new_label_now:nnnnnn {<label>} {<data>} {<thepage>}
{<current label name>} {<anchor>} {<extra>}
```

```
\cus_newlabel_now:nnnnnn
\cus_newlabel_now:xxxxxx
\cus_newlabel_shipout:nnnnnn
\cus_newlabel_shipout:xxxxxx
\cus_newlabel_shipout_x:nnnnnn
\cus_newlabel_shipout_x:xxxxxx
```

定义一个新的 $\langle label \rangle$ 。当未加载 hyperref 宏包时, 后三个参数无效。

$\langle label \rangle$ 、 $\langle thepage \rangle$ 、 $\langle current label name \rangle$ 、 $\langle anchor \rangle$ 总是立即展开。

```
\cus_split_bracket_head_default:nn {<tl>} {<default>}
\cus_split_bracket_head:n {<tl>}
```

```
\cus_split_bracket_head_default:nn *
\cus_split_bracket_head:n *
```

解析 $\langle tl \rangle$ 。判断其是否以一对方括号 $[]$ 起始, 若是则将方括号后的内容放入一个集合中 $\{ \}$ 。方括号不可直接嵌套, 需放入组中。

```
\ttfamily \ExplSyntaxOn
\exp_args:Ne \tl_to_str:n
{
  1. \cus_split_bracket_head:n { }
  \space 2. \cus_split_bracket_head:n { tail }
  \space 3. \cus_split_bracket_head:n { [bracket]tail }
  \space 4. \cus_split_bracket_head:n { [bracket] }
}
\ExplSyntaxOff
```

例 17

```
.....
1. [{}]{ } 2. [{}]{tail} 3. [{bracket}]{tail} 4. [{bracket}]{ }
```

\cus_split_bracket_tail_default:nn *	\cus_split_bracket_tail_default:nn {<tl>} {<default>}
\cus_split_bracket_tail:n *	\cus_split_bracket_tail:n {<tl>}

解析 <tl>。判断其是否以一对方括号 ([]) 结尾，若是则将方括号前的内容放入一个集合中 ({ })。方括号不可直接嵌套，需放入组中。

```
\ttfamily \ExplSyntaxOn
\exp_args:Nc \tl_to_str:n
{
  1. \cus_split_bracket_tail:n { }
  \space 2. \cus_split_bracket_tail:n { head }
  \space 3. \cus_split_bracket_tail:n { head[bracket] }
  \space 4. \cus_split_bracket_tail:n { [bracket] }
}
\ExplSyntaxOff
.....
1. {}[{}] 2. {head}[{}] 3. {head}[{bracket}] 4. {[bracket]}[{}]
```

例 18

\cus_if_preamble_p: *	\cus_if_preamble:TF {<true code>} {<false code>}
\cus_if_preamble:TF *	
\cus_if_document_p: *	
\cus_if_document:TF *	
\cus_if_after_documentclass_p: *	
\cus_if_after_documentclass:TF *	

测试是否在导言区、正文或在 \documentclass 后。

\cus_if_head_int:TF	\cus_if_head_int:TF {<tl>} {<true code>} {<false code>}
---------------------	---

测试 <tl> 是否以数字起始。

```
\ExplSyntaxOn
1. \cus_if_head_int:nTF { 2022 } { T } { F }
\ 2. \cus_if_head_int:nTF { +2022 } { T } { F } % 正数
\ 3. \cus_if_head_int:nTF { -2022 } { T } { F } % 负数
\ 4. \cus_if_head_int:nTF { '3746 } { T } { F } % 8进制数
\ 5. \cus_if_head_int:nTF { "7E6 } { T } { F } % 16进制数
\ 6. \cus_if_head_int:nTF { Notnu } { T } { F }
\ 7. \cus_if_head_int:nTF { \par } { T } { F }
\ 8. \cus_if_head_int:nTF { \c_true_bool } { T } { F } % \char
\ 9. \cus_if_head_int:nTF { \c_one_int } { T } { F } % int (count)
\ 10. \cus_if_head_int:nTF { \tracingmacros } { T } { F }
\ 11. \cus_if_head_int:nTF { \the\tracingmacros } { T } { F } % 展开为整数
\ 12. \cus_if_head_int:nTF { \baselineskip } { T } { F }
\ 13. \cus_if_head_int:nTF { \number\baselineskip } { T } { F } % 展开为整数
\ExplSyntaxOff
.....
1.T 2.T 3.T 4.T 5.T 6.F 7.F 8.T 9.T 10.F 11.T 12.F 13.T
```

例 19

\cus_exp_args:Nd	\cus_exp_args:Nd <function> {<tokens>}
\cus_exp_last_unbraced:Nd	

展开 <tokens> 两次。

```
\ExplSyntaxOn
\cus_exp_args:Nd \tl_to_str:n
```

例 20

```
{ \prg_replicate:nn { 5 } { \CuTeX } }
\ExplSyntaxOff
.....
\CuTeX \CuTeX \CuTeX \CuTeX \CuTeX
```

```
\ExplSyntaxOn
\cus_exp_last_unbraced:Nd \tl_to_str:n
{ \char_generate:nn { ` \ } { 1 } } \token_to_str:N }
\ExplSyntaxOff
.....
\token_to_str:N
```

例 21

```
\cus_parse_range:nnnN {<最小值>} {<最大值>} {<range list>} <sequence>
\cus_parse_range:nnnN {<最大值>} {<range list>} <sequence>
```

```
\cus_parse_range:nnnN
\cus_parse_range:(nnvN|nneN)
\cus_parse_range:nnN
\cus_parse_range:(nvN|neN)
```

解析一个整数列表，将其保存至 *<sequence>* 中。可使用 *->* 标记连续的范围。若范围的开始为空，则设它为 *<最小值>*，若终止为空，则设它为 *<最大值>*。

逆序的范围无效。

如果 *<最小值>* 被省略，则设它为 1。

是否检查越界值。

当设置了检查越界值时，结果的项被限制在 *<最小值>* 和 *<最大值>* 之中（含边界）。

否则，忽略这一限制，但逆序的范围仍然无效。

```
\cus_parse_range_check:
\cus_parse_range_nocheck:
```

```
\ExplSyntaxOn
\cus_parse_range:nnN { 10 } { ->2, 4->7, 8-> } \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq { ,~ } \par

\cus_parse_range:nnN { 10 } { -1->2, 4->7, 9->12, 20, 30->32 } \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq { ,~ } \par

\cus_parse_range_nocheck:
% 不检查越界
\cus_parse_range:nnN { 10 } { -1->2, 9->12, 20, 30->32 } \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq { ,~ } \par

\cus_parse_range:nnN { 10 } { -1->2, 9->12, 20, 32->30 } \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq { ,~ } \par % 逆序, 无效
\ExplSyntaxOff
.....
1, 2, 4, 5, 6, 7, 8, 9, 10
1, 2, 4, 5, 6, 7, 9, 10
-1, 0, 1, 2, 9, 10, 11, 12, 20, 30, 31, 32
-1, 0, 1, 2, 9, 10, 11, 12, 20
```

例 22

```
\cus_set_parse_range_delimiter:n {<delimiter>}
\cus_set_parse_range_default_delimiter:
```

```
\cus_set_parse_range_delimiter:n
\cus_set_parse_range_default_delimiter:
```

设置范围的分隔符为 *<delimiter>*，默认为 *->*。对分隔符的修改应该是局部的。

```
\ExplSyntaxOn
\cus_set_parse_range_delimiter:n { - }
```

例 23


```
\cus_parse_range:nnN { 10 } { 0-2, 7-12, 20, 32-30 } \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq { ,~ }
\ExplSyntaxOff
.....
1, 2, 7, 8, 9, 10
```

3.2.1 psr, 处理器

有时，一个命令需要根据不同的设置显示不同的效果，但这个命令的执行逻辑已经确定，重新修改这个命令不是一个好的做法，因为无法保证对它的修改是正确的。一般可以通过在这些命令中插入钩子（hook）或修改这个命令内部真正有效的命令来间接地修改它。后者就是处理器的主要思想。

处理器是包装过的宏。

一个处理器是在某些命令内部发挥作用的接口宏，处理器的规则是处理器真正发挥执行操作的宏，通过让处理器遵循（obey）某个规则来控制处理器以何种方式运行（这些规则也是宏）。这样我们只需定义一些规则，然后根据需让处理器遵循某个规则，从而达到不同的效果。

与使用条件判断相比，具有更一致的接口，可以增加任意个处理方式，并且在同一次展开时不必重复判断。

与直接使用宏相比，调用接口更加一致。

```
\cus_new_psr:nnn
\cus_set_psr:nnn
\cus_gset_psr:nnn
\cus_use_psr:n
```

```
\cus_new_psr:nnn {<处理器>} {<参数数目>} {<code>}
\cus_use_psr:n {<处理器>}
```

创建、使用处理器。

每个处理器的参数数目固定。在使用处理器时，其后需有对应数目的参数。

处理器以其名称唯一识别，不可定义参数数目不同的同名处理器。

```
\cus_new_psrrule:nnn
\cus_set_psrrule:nnn
\cus_gset_psrrule:nnn
```

```
\cus_new_psrrule:nnn {<处理器>} {<规则>} {<code>}
```

创建从属于〈处理器〉的规则〈规则〉。该〈规则〉可使用〈处理器〉预先定义的参数。

```
\cus_obey_psrrule:nn
\cus_gbey_psrrule:nn
```

```
\cus_obey_psrrule:nn {<处理器>} {<规则>}
```

对〈处理器〉局部或全局应用〈规则〉。注意，全局应用不是 gobey。

```
\cus_psr_if_exist:p:n *
\cus_psr_if_exist:nTF *
\cus_psrrule_if_exist:p:nn *
\cus_psrrule_if_exist:nnTF *
\cus_psr_if_compatible:p:nn *
\cus_psr_if_compatible:nnTF *
```

```
\cus_psr_if_exist:nTF {<处理器>} {<true code>} {<false code>}
\cus_psrrule_if_exist:nTF {<处理器>} {<规则>} {<true code>} {<false code>}
\cus_psr_if_compatible:nnTF {<处理器1>} {<处理器2>} {<true code>} {<false code>}
```

判断处理器、处理器的规则是否存在。或判断两个处理器是否兼容，当前，两个参数数目一致时视为兼容。

```
\cus_exec_psrrule:nn
```

```
\cus_exec_psrrule:nn {<处理器>} {<规则>}
```

直接执行〈处理器〉的〈规则〉。可用于其它规则中，相当于一个宏。其后需有对应数目的参数。

```
\cus_psr_argument_count:n *
```

```
\cus_psr_argument_count:n {<处理器>}
```

计算处理器可用的参数数目。


```
\cus_new_psrrule_eq:nnn {<处理器>} {<规则1>} {<规则2>}
```

将 <规则₁> 设置为与 <规则₂> 相等，它们从属于 {<处理器>}。

```
\cus_new_psrrule_eq_cs:nnN {<处理器>} {<规则>} <function>
```

将 <处理器> 的 <规则> 设置为与 <function> 相等。这 <function> 的参数数必须与 <处理器> 的参数数相等，且必须是非定界的变量（undelimited parameter）。

```
\cus_new_psrrule_eq:nnn
\cus_set_psrrule_eq:nnn
\cus_gset_psrrule_eq:nnn
```

```
\cus_new_psrrule_eq_cs:nnN
\cus_new_psrrule_eq_cs:nnc
\cus_set_psrrule_eq_cs:nnN
\cus_set_psrrule_eq_cs:nnc
\cus_gset_psrrule_eq_cs:nnN
\cus_gset_psrrule_eq_cs:nnc
```

§ 3 struct 模块

以下简单描述 struct 模块中目录的数据结构。

```
\addcombinedlistitem {<type>} {<cbl levels>}
```

若要添加新的目录类型，必须先声明 <type>。<cbl levels> 为这个目录类型可用的层级名称，层级名后的中括号括起的数字表示其层级 <level>，也可使用通常的 key=val 的形式。

比如，对于标准的目录 \tableofcontents，它写入的 ext 为 toc，有

```
\addcombinedlistitem{toc}
{
  part[-1],
  chapter[0],
  section[1], subsection[2], subsubsection[3],
  sub3section[4], sub4section[5],
  paragraph[4], subparagraph[5],
}
```

例 24

对于标准的 \listoffigures 和 \listoftables，有

```
\addcombinedlistitem{lof}{ figure } % figure=0
\addcombinedlistitem{lot}{ table } % table=0
```

例 25

原有的 \addcontentsline 接受三个参数，其中第一个参数为写入的文件的扩展名，在这里就是目录项的类型 <type>，其第二个参数就是这里的 <cbl level>，即层级名称，<cbl levels> 应包含这个参数的所有可能值；第三个参数就是 <entry>。

```
\getcbltypelevel {<type>} {<cbl level>}
```

```
\getcbltypelevel *
```

展开为 <type> 类型中层级名称 <cbl level> 对应的层级数。

```
\getcbltotalcounts {<type>}
```

```
\getcbltotalcounts *
```

展开为 <type> 类型的目录条目数。若 <type> 为空，则为各类型的总和。

每个类型的条目数在使用 \enablecombinedlist 时就已经确定，此后不可更改，只需常数时间即可获取（包括为空时的情况）。

每个类型的目录条目包含为如下数据：

```
\cus@type@contentsline {<type>}{<cbl count>}{<tags>}{<level>}{<entry>}{
  <thepage>}{<anchor>}\cus@type@contentsline@
```

- \cus@type@contentsline, \cus@type@contentsline@, 这两个宏标记条目的边界；

- $\langle type \rangle$ 为目录类型名；
- $\langle cbl\ count \rangle$ 为本条目在存储着所有目录项的那个列表中的位置；
- $\langle tags \rangle$ 为一个列表，它标记着这个目录项的某些信息，第一项一般为这个目录项层级的名称。在 `\chapter` 中为 `chapter`，在 `figure` 中，为 `figure`；
- $\langle level \rangle$ 为这个目录项的层级，是一个整数；
- $\langle entry \rangle$ 为这个目录项的值，通常包含着标题或 `caption`；
- $\langle thepage \rangle$ 为目录项所在页的 `\thepage`；
- $\langle anchor \rangle$ 为目录项原位置的锚点。仅在 `hyperref` 宏包加载时有效。可作为 `\hyperlink` 命令的第一个参数。

每个目录类型不仅分别存储在各自的列表中，还存储在一个统一的列表（以下称为 `cbl` 列表）中。在目前的版本中，存储顺序是按照宏的执行顺序而并不一定是文档实际输出顺序（例如，一个浮动体可能出现在其执行顺序之前）。

`cbl` 列表除了 `\cus@type@contentsline`、`\cus@type@contentsline@`、 $\langle cbl\ count \rangle$ 改为 `\cus@cbl@contentsline`、`\cus@cbl@contentsline@`、 $\langle type\ count \rangle$ 外，其它未改变。这里的 $\langle type\ count \rangle$ 为此条目在其对应类型的列表中的位置。

 $\backslash\mathrm{getcblitemname}$ *

 $\backslash\mathrm{getcblitemname} \{ \langle type \rangle \} \{ \langle count \rangle \}$

展开为存储着类型 $\langle type \rangle$ 第 $\langle count \rangle$ 项的那个宏的名称。可以使用 `\UseName`、`\@nameuse` 等获取这个目录项。也可以作为 `LATEX3` 函数的 `c` 参数，如 `\tl_show:c{\getcblitemname}{\getcbltotalcounts{}}` 在终端中显示 `cbl` 列表的最后一项。

 $\backslash\mathrm{getcblitemdata}$ *

 $\backslash\mathrm{getcblitemdata} \{ \langle type \rangle \} \{ \langle data \rangle \} \{ \langle count \rangle \}$

获取 $\langle type \rangle$ 列表第 $\langle count \rangle$ 项 $\langle data \rangle$ 的值。 $\langle type \rangle$ 为空，则获取 `cbl` 列表中的值。 $\langle data \rangle$ 可为 `type`、`count`、`tags`、`level`、`entry`、`thepage`、`anchor`。

 $\backslash\mathrm{iteratecontents}$
 $\backslash\mathrm{iteratecontents} \{ \langle type \rangle \} \{ \langle inline\ code \rangle \}$

使用 $\langle inline\ code \rangle$ 迭代 $\langle type \rangle$ 中的每一个目录项。若 $\langle type \rangle$ 为空，则迭代 `cbl` 列表。

$\langle inline\ code \rangle$ 可使用 7 个参数，分别顺序代表前述的 7 个数据值。

 $\backslash\mathrm{getcbldefaultlevellistname}$ *

 $\backslash\mathrm{getcbldefaultlevellistname} \{ \langle type \rangle \}$

展开为一个 `clist` 的名称，这个 `clist` 的前 n 项为 $\langle type \rangle$ 这个类型中层级为 0 的项的索引（即 $\langle type\ count \rangle$ 的值），最后一项为 `\getcbltotalcounts{ $\langle type \rangle$ }+1`。

`CurrentTocDefaultLevelCount`

当 `toc` 类型的目录中添加层级为 0 的条目时这个计数器递增一次。如，在使用不带星号的 `\chapter` 时将递增 1。

在此处，它的值为 4。

```
\ExplSyntaxOn
\tl_set:Nx \l_tmpa_tl { \getcbldefaultlevellistname {toc} }
\clist_item:cn { \l_tmpa_tl } { \value{CurrentTocDefaultLevelCount} } ,~
\clist_item:cn { \l_tmpa_tl } { \value{CurrentTocDefaultLevelCount} + 1 }
\ExplSyntaxOff
```

例 26

28, 34。表示本章节的所有目录项为 `toc` 类型的目录中的第 28 – 32 项。

以下代码输出本章目录。

例 27

```

\ExplSyntaxOn
\int_compare:nNnT { \getcblltotalcounts{} } > { 0 }
{
  \tl_set:Nx \l_tmpa_tl { \getcblldefaultlevellistname {toc} }
  \int_set:Nn \l_tmpa_int % 本章开始
  { \clist_item:cn { \l_tmpa_tl } { \value{CurrentTocDefaultLevelCount}
  } }
  \int_set:Nn \l_tmpb_int % 下章开始
  { \clist_item:cn { \l_tmpa_tl } { \value{CurrentTocDefaultLevelCount}
  } + 1 } }
  \int_step_inline:nnnn { \l_tmpa_int } { 1 } { \l_tmpb_int - 1 }
  { \tl_use:c { \getcblitemname {toc} {#1} } } }
}
\ExplSyntaxOff

```

第三章 编程接口	24
§ 1 L ^A T _E X 2 _ε 的钩子机制	24
§ 2 util 模块	25
3.2.1 psr, 处理器	28
§ 3 struct 模块	29
§ 4 L ^A T _E X 2 _ε 的 mark 机制	31

§ 4 L^AT_EX 2_ε 的 mark 机制

L^AT_EX 2_ε 在 2022-06-01 的发行版中引入了新的 mark 机制。本节简述这一机制，更详细的说明请参考 `ltmarks-doc.pdf`。本说明文档的源码也使用了这个新机制。

```

\NewMarkClass {<class>}
\mark_new_class:n {<class>}

```

```

\NewMarkClass
\mark_new_class:n

```

声明一个新的 mark class。仅能在导言区使用。

```

\InsertMark {<class>} {<text>}
\mark_insert:nn {<class>} {<text>}

```

```

\InsertMark
\mark_insert:nn
insertmark

```

添加 mark 到当前的垂直列中，这个 mark 包含 <text>（被完全展开）。

在不能使用浮动体的地方也无法使用这个命令，如在一个盒子中使用它们时无效。特别的，在 `multicols` 等多栏环境中使用它们将无效。

在执行 `\InsertMark`、`\mark_insert:nn` 时，将首先执行 `insertmark` 钩子。

```

\TopMark [<region>] {<class>}
\mark_use_last:nn {<region>} {<class>}

```

```

\TopMark          *
\FirstMark        *
\LastMark         *
\mark_use_top:nn  *
\mark_use_first:nn *
\mark_use_last:nn *

```

展开为 <class> 在 <region> 中相应位置的 <text>。

`\FirstMark`、`\LastMark` 分别展开为 <region> 的第一个、最后一个 <text>。

`\TopMark` 展开为上一个 <region> 的最后一个 <text>。

目前，<region> 可选值为 `page`、`previous-page`、`column`、`previous-column`。在多栏（双栏）文档中，`first-column`、`last-column` 分别代表最左列和最右列。<region> 默认为 `page`。

<code>\IfMarksEqualTF</code>	★	<code>\IfMarksEqualTF [<i>{region}</i>] <i>{<class>}</i> <i>{<pos₁}</i> <i>{<pos₂}</i> <i>{<true>}</i> <i>{<false>}</i></code>
<code>\mark_if_eq:nnnnTF</code>	★	<code>\mark_if_eq:nnnnTF <i>{<region>}</i> <i>{<class>}</i> <i>{<pos₁}</i> <i>{<pos₂}</i> <i>{<true>}</i> <i>{<false>}</i></code>
<code>\mark_if_eq:nnnnnnTF</code>	★	<code>\mark_if_eq:nnnnnnTF <i>{<region₁}</i> <i>{<class₁}</i> <i>{<pos₁}</i></code> <code><i>{<region₂}</i> <i>{<class₁}</i> <i>{<pos₂}</i> <i>{<true>}</i> <i>{<false>}</i></code>

判断两个 mark 的 *<text>* 是否完全相等 (使用 `\ifx`)。

<pos> 为 top、first、last 之一。

原有的 `\markboth`、`\markright`、`\leftmark`、`\rightmark` 仍然可用。

可用使用 `2e-left`、`2e-right`、`2e-right-nonempty class` 来获取 `\leftmark` 和 `\rightmark`。`2e-right` 与 `2e-right-nonempty` 的区别是,后者仅在 *<rightmark>* 非空时才更新。

第四章 库的文档接口

§ 1 doc 库

doc 库用于支持排版说明文档。本库移植修改自 l3doc 和 ctxdoc。

当加载本库时,将创建两个索引,名为 docusage 和 docchange,分别记录代码和版本历史。使用 `\PrintIndex`、`\PrintChanges` 分别输出代码索引和版本历史。

本库的详细用法可参考本说明文档的源码。

本库提供 `\cs`、`\cmd`、`\tn` 来排版宏。

<code>\cs</code>	<code>\cs [<i><cmd key-val></i>] <i>{<macro name>}</i></code>
<code>\cmd</code>	<code>\cmd [<i><cmd key-val></i>] <i>{<macro>}</i></code>
<code>\tn</code>	<code>\tn [<i><cmd key-val></i>] <i>{<tex macro name>}</i></code>

排版宏。`\tn` 专用于排版 T_EX 和 L^AT_EX 2_ε 的宏。

<code>doc/cmd/index</code>	<code>index = <i>{<index entry>}</i></code>	
<code>doc/cmd/module</code>	<code>module = <i>{<module name>}</i></code>	
<code>doc/cmd/no-index</code>	<code>no-index = <i><true false></i></code>	初始值: false
<code>doc/cmd/space</code>	<code>do-index = <i><true false></i></code>	
	<code>space = <i><true false></i></code>	初始值: false

`\cs`、`\cmd`、`\tn` 可用的键值选项。

`no-index` 控制是否写入索引文件。`do-index` 与 `no-index` 相反。

当在 **function**、**keyval** 和 **syntax** 环境中时,不写入索引文件。

`space` 控制是否将空格替换为 `\textvisiblespace`。

<code>\meta</code>	排版宏的参数。
<code>\Arg</code>	
<code>\oarg</code>	
<code>\parg</code>	
<code>\pkg</code>	
<code>\env</code>	
<code>\cls</code>	
<code>\opt</code>	
<code>\file</code>	
<code>\docfile</code>	

```
\begin{function} [<function key-val>] {<functions clist>}
...
\end{function}
```

function

显示函数说明。*<functions clist>* 可以是宏或者环境名。

```
\begin{keyval} [<function key-val>] {<keys clist>}
...
\end{keyval}
```

keyval

显示键值选项的说明。*<keys clist>* 为键列表。

```
\begin{syntax}
...
\end{syntax}
```

syntax

输出使用方法。

本环境中每个输入行都为输出行（一个段落），除每行首尾的空格被移除外，所有的空格都被保留下来；此外，可使用 `~` 输出一个空格的宽度。

本环境中可以使用几个特殊的字符（字符对），它们是语法糖：

- `<...>` — 在文本环境时这相当于 `\meta{...}`，数学环境时仍然为小于、大于号；但有几个例外：
- `<&...>` — 当文本环境中 `<` 紧跟 `&` 时，`...` 被视为可选值；
- `<{...}>` — 当文本环境中 `<...>` 中的内容为一个正确嵌套的组时，它被视为 `\marg{...}`；
- `&` — 其后的值被认为是初始值，每行最多应仅使用一次，与之等价的写法是：`\initialval ...`（无需花括号）；但有几个例外：
- `&*` — 当 `&` 紧跟 `*` 时，相当于 `\repinitval`，可自行设置文字；
- `&&` — 当 `&` 紧跟 `&` 时，相当于 `\forbiddenval`，表示禁止设置值；
- `&#` — 当 `&` 紧跟 `#` 时，相当于 `\automaticval`，表示如未给出将自动设置值；
- `&~` — 当 `&` 紧跟 `~` 时，相当于 `\initemptyval`；
- `&!` — 当 `&` 紧跟 `!` 时，相当于 `\resetval`，表示在对应命令或环境中其值均被重设；
- `|` — 相当于 “|” (`\orbar`)，一般用于分隔不同的可选值；
- `(...)` — 这中间的值被认为是默认值，以粗体显示，与之等价的写法是：`\defaultval{...}`。

T_EXhackers note: # 在本环境中的类别码被设置为 12 (other)。

`function`、`keyval` 和 `syntax` 环境均可使用 `\V` 命令，它和 `\Verbatimize` 一样，但以当前字体显示。

EXP
rEXP

不可设置值
不可设置值

doc/function/EXP
doc/function/rEXP

EXP 将函数标记为完全可展的 (fully expandable functions)，可同时用作 `x`、`e`、`f` 类型的参数。如 `\string`、`\cs_to_str:N`。使用 `*` 标记。

rEXP 将函数标记为受限可展的 (restricted expandable functions)，这些函数是完全可展的，但不能在 `f` 类型的参数中完全展开 (cannot be fully expanded)。如 `\seq_map_function:NN`。使用 `☆` 标记。

doc/function/TF	TF	不可设置值
doc/function/pTF	pTF	不可设置值
doc/function/noTF	noTF	不可设置值

标记函数为带有真假值参数的函数。

TF 将函数标记为带有真假参数的函数，如 `\tl_if_eq:nn`。pTF 在 TF 的基础上，还将函数标记为带有可用于 `\if_predicate:w` 的函数。noTF 在 TF 的基础上，还将函数标记为不带真假参数的函数，如 `\prop_get:NnN`。

doc/function/added	added = {<年>-<月>-<日>} 或 {<年>/<月>/<日>}
doc/function/updated	updated = {<年>-<月>-<日>} 或 {<年>/<月>/<日>}

此函数是何时添加的或最近一次修改在何时。

doc/function/label	label = {<label list>}
doc/function/label*	label* = {<label list>}

设置 `\label`。label 不会设置默认的 label，label* 会设置默认的 label。

doc/function/verb	verb	不可设置值
-------------------	------	-------

将整个 `<functions clist>` 或 `<keys clist>` 看作是一个函数或键。

doc/function/module	module = {<module name>}
---------------------	--------------------------

设置当前函数所在的模块。

doc/function/type	type = {<类型>}
-------------------	---------------

设置当前的类型，如 `function`、`environment`、`keyval`。

doc/function/path	path = {<key path>}
-------------------	---------------------

设置键值参数的键路径。

doc/function/frame	frame = {<frame key-val>}
doc/function/frame+	frame+ = {<frame key-val>}

设置外部方框盒子的选项。

texnote	<pre>\begin{texnote} ... \end{texnote}</pre>
---------	--

\csref	[<类型>] {<cs name>}
\csreflist	[<类型>] {<cs name list>}
\envref	[<类型>] {<env name>}
\envreflist	[<类型>] {<env name list>}
\keyref	[<key path>] {<key name>}
\keyreflist	[<key path>] {<key name list>}

引用命令，环境或键。对于列表的引用，可以通过 `\cus@doc@refrange` 修改分隔字符。

§ 2 bnf 库

bnf 库用于排版基于 Backus-Naur Form (BNF 范式) 的文法。

```
\begin{latexbnf}[{texbnf key-val}]
<(non-terminal)> : <(non-terminal)>
<(non-terminal)> : "<(terminal)>"
<(non-terminal)> : <(non-terminal)> | <(non-terminal)>
...
\end{latexbnf}
```

latexbnf

BNF 范式排版环境。

可使用 `:=` 代替 `:`。

当 `<` 在行首时, 被解释为定义一个新的句法。

在此环境中, `_`、`^` 相当于 `\lo`、`\hi`, 可以直接在文本中使用, 分别表示上下标。

排版时, 既可使用这种字符标记的形式, 也可使用下述的命令形式。混合使用它们也是可被接受的。

这些字符标记中的文字被正常处理。

连续使用两次: 可输出 “:”, 连续使用两次 `|` 可输出 “|”。

这些标记字符在数学模式中表示它们原本的含义。

此环境中空行被忽略了, 若要显示空行, 可以在此行使用 `\null` 或使用一个空盒子: `\mbox{}`。

本环境中还可使用 `\V`, 它相当于 `\Verbatimize`。

```
\BNFN {(non-terminal)}
\BNFT {(terminal)}
```

\BNFItem
\BNFN
\BNFI
\BNFO
\BNFT

`\BNFItem` 用于标记一个句法 (syntax) 的开始。

`\BNFN` 排版非终结符。`\BNFT` 排版非终结符, 它的使用方式和效果与 `\verb` 类似。

`\BNFI` 表示它之前的内容被定义为它之后的内容。

`\BNFO` 表示 “或者”。

除 `\BNFItem` 外, 上述命令均可在正文环境中使用。

在 `latexbnf` 环境中, 可使用 `\is` 代替 `\BNFI`, `\alt` 代替 `\BNFO`, 它们会在两侧加上空白。

本库还支持给非终结符加上超链接。

当加载了 `hyperref` 宏包后, 右侧的非终结符将链接到对应的定义处 (如果其定义存在)。

当使用字符标记时, 可使用 `\h<(non-terminal)>` 的形式显示使用。在定义的左侧使用时, 被解释为设置该非终结符的超链接位置; 在定义的右侧使用时, 被解释为链接到这个非终结符的定义处。可以显式使用 `\BNFAnchor` 或 `\BNFref` 来表示上述的两种类型。

```
hyper = {true|false}
hyper-color = {<颜色>}
```

初始值: **false**

texbnf/hyper
texbnf/hyper-color

`hyper` 控制是否使用默认使用超链接而无需显示使用 `\h`。

`hyper-color` 控制超链接的颜色。未给定时, 使用超链接默认的颜色。

`\BNFN` 超链接的使用与否也受 `hyper` 选项控制。

例 28

```

\begin{latexbnf}[hyper, hyper-color=purple]
<glue> ::= <optional signs><internal glue>
| <dimen><stretch><shrink>
<stretch> ::= "plus"<dimen> | "plus"<fil dimen> | <optional spaces>
<shrink> ::= "minus"<dimen> | "minus"<fil dimen> | <optional spaces>
<fil dimen> ::= <optional signs><factor><fil unit><optional spaces>
<fil unit> ::= "fil" | <fil unit>"l"
<muglue> ::= <optional signs><internal muglue>
| <mudimen><mustretch><mushrink>
<mustretch> ::= "plus"<mudimen> | "plus"<fil dimen> | <optional spaces>
<mushrink> ::= "minus"<mudimen> | "minus"<fil dimen> | <optional spaces>
\end{latexbnf}

```

.....

<glue> → <optional signs><internal glue>
 | <dimen><stretch><shrink>
 <stretch> → plus<dimen> | plus<fil dimen> | <optional spaces>
 <shrink> → minus<dimen> | minus<fil dimen> | <optional spaces>
 <fil dimen> → <optional signs><factor><fil unit><optional spaces>
 <fil unit> → fil | <fil unit>l
 <muglue> → <optional signs><internal muglue>
 | <mudimen><mustretch><mushrink>
 <mustretch> → plus<mudimen> | plus<fil dimen> | <optional spaces>
 <mushrink> → minus<mudimen> | minus<fil dimen> | <optional spaces>

完全等价的一个写法是：

例 29

```

\begin{latexbnf}[hyper, hyper-color=purple]
\BNFItem \BNFN{glue}\is\BNFN{optional signs}\BNFN{internal glue}
\alt\BNFN{dimen}\BNFN{stretch}\BNFN{shrink}
\BNFItem \BNFN{stretch}\is\BNFT{plus}\BNFN{dimen}\alt\BNFT{plus}\BNFN{fil}
↪ dimen}\alt\BNFN{optional spaces}
\BNFItem \BNFN{shrink}\is\BNFT{minus}\BNFN{dimen}\alt\BNFT{minus}\BNFN{fil}
↪ dimen}\alt\BNFN{optional spaces}
\BNFItem \BNFN{fil dimen}\is\BNFN{optional signs}\BNFN{factor}\BNFN{fil}
↪ unit}\BNFN{optional spaces}
\BNFItem \BNFN{fil unit}\is\BNFT{fil}\alt\BNFN{fil unit}\BNFT{l}
\BNFItem \BNFN{muglue}\is\BNFN{optional signs}\BNFN{internal muglue}
\alt\BNFN{mudimen}\BNFN{mustretch}\BNFN{mushrink}
\BNFItem
↪ \BNFN{mustretch}\is\BNFT{plus}\BNFN{mudimen}\alt\BNFT{plus}\BNFN{fil}
↪ dimen}\alt\BNFN{optional spaces}
\BNFItem
↪ \BNFN{mushrink}\is\BNFT{minus}\BNFN{mudimen}\alt\BNFT{minus}\BNFN{fil}
↪ dimen}\alt\BNFN{optional spaces}
\end{latexbnf}

```



```
format = {\code}
Tformat = {\code}
clear-all-format
```

初始值: `\ttfamily`
不可设置值

设置格式。

```
texbnf/format
texbnf/Nformat
texbnf/Iformat
texbnf/Oformat
texbnf/Tformat
texbnf/clear-all-format
```

```
Nleft = {\code}
Nright = {\code}
I = {\code}
O = {\code}
```

初始值: `\ensuremath{\langle\langle}, \langle`
初始值: `\ensuremath{\langle\rangle}, \rangle`
初始值: `\ensuremath{\langle\longrightarrow}, \rightarrow`
初始值: `\cus@mathrule, |`

设置 \BNFN、\BNFT 左右的符号, \BNFI、\BNFO 的替换符号。

```
Nleft
Nright
Tleft
Tright
N
T
O
I
```

```
label-prefix = {\前缀}
label-suffix = {\后缀}
```

初始值: `texbnf//`

```
label-prefix
label-suffix
```

当设置超链接锚点时, 会写入 \label, 使用 label-prefix 和 label-suffix 可在 \label 名中添加 <前缀> 和 <后缀>。使用 \BNFref 时, 也需正确添加它们。

§ 3 ref 库

ref 库提供交叉引用的一些额外功能。

本库改进了 \ifPageOdd 和 \ifAbsPageOdd, 使得它们在任何位置都有效。

```
\ifLabelOdd {\label} {\true} {\false}
```

```
\ifLabelOdd
```

判断这个 <label> 是否定义在奇数页。当 <label> 不存在时使用 <false>。

<label> 所在页码必须以阿拉伯数字显示, 否则使用 <false> 分支。

索引

代码索引

A	
<code>\addcombinedlistitem</code>	29
<code>\Arg</code>	32
<code>\atleastfiller</code>	14
<code>\automaticval</code>	33
B	
<code>\background</code>	18
<code>\backgroundpicture</code>	18
<code>\BNFanchor</code>	35
<code>\BNFI</code>	35
<code>\BNFItem</code>	35
<code>\BNFN</code>	35
<code>\BNFO</code>	35
<code>\BNFref</code>	35
<code>\BNFT</code>	35
<code>\breakablefiller</code>	14
C	
<code>\chapter</code>	20, 30
<code>\cls</code>	32
<code>\cmd</code>	32
<code>\cs</code>	32
<code>\csref</code>	34
<code>\csreflist</code>	34
<code>CurrentTocDefaultLevelCount</code>	30
cus commands:	
<code>\cus_exec_psrrule:nn</code>	28
<code>\cus_exp_args:Nd</code>	26
<code>\cus_exp_last_unbraced:Nd</code>	26
<code>\cus_gbey_psrrule:nn</code>	28
<code>\cus_get_label:nn</code>	25
<code>\cus_gset_psr:nnn</code>	28
<code>\cus_gset_psrrule:nnn</code>	28
<code>\cus_gset_psrrule_eq:nnn</code>	29
<code>\cus_gset_psrrule_eq_cs:nnN</code>	29
<code>\cus_if_after_documentclass:TF</code>	26
<code>\cus_if_after_documentclass_p:</code>	26
<code>\cus_if_document:TF</code>	26
<code>\cus_if_document_p:</code>	26
<code>\cus_if_head_int:TF</code>	26
<code>\cus_if_preamble:TF</code>	26
<code>\cus_if_preamble_p:</code>	26
<code>\cus_new_psr:nnn</code>	28
<code>\cus_new_psrrule:nnn</code>	28
<code>\cus_new_psrrule_eq:nnn</code>	29
<code>\cus_new_psrrule_eq_cs:nnN</code>	29
<code>\cus_newlabel_now:nnnnnn</code>	25
<code>\cus_newlabel_shipout:nnnnnn</code>	25
<code>\cus_newlabel_shipout_x:nnnnnn</code>	25
<code>\cus_obey_psrrule:nn</code>	28
<code>\cus_parse_range:nnN</code>	27
<code>\cus_parse_range:nnnN</code>	27
<code>\cus_parse_range_check:</code>	27
<code>\cus_parse_range_nocheck:</code>	27
<code>\cus_psr_argument_count:n</code>	28
<code>\cus_psr_if_compatible:nnTF</code>	28
<code>\cus_psr_if_compatible_p:nn</code>	28
<code>\cus_psr_if_exist:nnTF</code>	28
<code>\cus_psr_if_exist_p:n</code>	28
<code>\cus_psrrule_if_exist:nnTF</code>	28
<code>\cus_psrrule_if_exist_p:nn</code>	28
<code>\cus_set_parse_range_default_delimiter:</code>	27
<code>\cus_set_parse_range_delimiter:n</code>	27
<code>\cus_set_psr:nnn</code>	28
<code>\cus_set_psrrule:nnn</code>	28
<code>\cus_set_psrrule_eq:nnn</code>	29
<code>\cus_set_psrrule_eq_cs:nnN</code>	29
<code>\cus_split_bracket_head:n</code>	25
<code>\cus_split_bracket_head_default:nn</code>	25
<code>\cus_split_bracket_tail:n</code>	26
<code>\cus_split_bracket_tail_default:nn</code>	26
<code>\cus_use_psr:n</code>	28

\CUSDependency	24	\envreflist	34
\CUSIfLibraryAtLeast	24		
\CUSLoadLibrary	24	F	
\CUSProvideExplLibrary	24	\fancycenter	10
\CUSProvideLibrary	24	\file	32
\CusLaTeX	1	\filler	12
\cussetstyle	1	filler options:	
\cussetup	1	align	14
\CusTeX	1	box	13
		box*	13
D		cdotted	13
\dashfiller	12	center	14
\definetitle	20	clear-box	13
dependency options:		color	13
disable	24	content	13
library	24	dash	13
module	24	dashed	13
package	24	dotted	13
doc/cmd options:		full	13
index	32	hspace	12
module	32	hspace*	12
no-index	32	is-dash	13
space	32	not-space	12
doc/function options:		raise	13
added	34	rule	13
EXP	33	sep	13
frame	34	size	12
frame+	34	size*	12
label	34	solid	13
label*	34	space	12
module	34	spread	14
noTF	34	type	14
path	34	vspace	12
pTF	34	vspace*	12
rEXP	33	\FirstMark	31
TF	34	\forbiddenval	33
type	34	\foreground	18
updated	34	\foregroundpicture	18
verb	34	frame options:	
\docfile	32	align	11
		center	11
E		first	11
\enablecombinedlist	23	first*	11
\env	32	frame	11
\envref	34		

frame*	11	\IterateClist	2
ignore-warnings	11	\iteratecontents	30
init	11	\IterateList	2
inner	11	\IterateThread	2
last	11		
last*	11	K	
left	11	\keyref	34
middle	11	\keyreflist	34
middle*	11	keyval	33
outer	11		
outer-sep	10	L	
ratio	11	label-prefix	37
right	11	label-suffix	37
rule-width	10	\LastMark	31
sep	10	latexbnf	35
whole	11	layout options:	
whole*	11	asymmetric	7
width	11	bindingoffset	7
Framed	10	bmargin	7
function	33	body	5
		bottom	7
		bottommargin	7
		centering	7
G		columnsep	8
\getcbldefaultlevellistname	30	direction	5
\getcblitemdata	30	divide	7
\getcblitemname	30	foot	8
\getcbltotalcounts	29	footnotesep	7
\getcbltypelevel	29	footoffset	8
\getlabelinfo	25	footskip	8
		hcentering	7
I		hdivide	7
I	37	head	8
\ifAbsPageOdd	3, 37	headheight	8
\ifbotfloat	10	headoffset	8
\iffloatpage	10	headsep	8
\iffootnote	10	height	5
\ifLabelOdd	37	heightrounded	6
\IfMarksEqualTF	32	hfoffset	8
\ifPageOdd	3, 37	hmargin	7
\iftopfloat	10	hmarginratio	7
\index	20	hoffset	8
\initemptyval	33	horizontalmargin	7
\initialval	33	horizontalmarginratio	7
\InsertMark	31		
insertmark	31		

hscale	5	orientation	5
ignoreall	6	outer	7
ignorefoot	6	paper	4
ignorehead	6	paperheight	4
ignoreheadfoot	6	papername	4
ignorehf	6	paperorientation	5
ignoremarginpar	6	papersize	4
ignoremp	6	paperwidth	4
includeall	6	portrait	5
includefoot	5	preset	9
includehead	5	reversemarginpar	7
includeheadfoot	5	reversemp	7
includehf	5	right	7
includemarginpar	5	rightmargin	7
includemp	5	rmargin	7
inner	7	scale	5
landscape	5	showframe	9
layout	5	showmarking	9
layoutheight	5	text	5
layoutoffset	5	textheight	5
layoutname	5	textwidth	5
layoutoffset	5	tmargin	7
layoutsiz	5	top	7
layoutvoffset	5	topmargin	7
layoutwidth	5	total	5
left	7	totalheight	5
leftmargin	7	totalwidth	5
lines	5	twocolumn	8
lmargin	7	twoside	7
marginpar	8	vcentering	7
marginparsep	8	vdivide	7
marginparwidth	8	verticalmargin	7
marginratio	7	verticalmarginratio	7
marking	9	vmarginratio	7
name	9	voffset	8
nofoot	8	vscale	5
noheadfoot	8	width	5
nohf	8	\listoffigures	23
nomarginpar	8	\listoftables	23
nomp	8	lthooks commands:	
offset	8	\AddToHookNext	25
onecolumn	8	\AddToHooks	25
		\ClearHookNext	25

<code>\RemoveFromHook</code>	25	<code>right-to-left</code>	17
<code>\UseHook</code>	24	<code>RL</code>	17
M		<code>rule</code>	16
<code>\makeindex</code>	19	<code>rule-color</code>	16
<code>\MapClist</code>	2	<code>sep</code>	15
<code>\MapList</code>	2	<code>tolerance</code>	16
mark commands:		<code>top-fuzz</code>	16
<code>\mark_if_eq:nnnnnnTF</code>	32	<code>unbalance</code>	16
<code>\mark_if_eq:nnnnTF</code>	32	<code>v-fuzz</code>	17
<code>\mark_insert:nn</code>	31	N	
<code>\mark_new_class:n</code>	31	<code>N</code>	37
<code>\mark_use_first:nn</code>	31	<code>\newindextype</code>	19
<code>\mark_use_last:nn</code>	31	<code>\NewMarkClass</code>	31
<code>\mark_use_top:nn</code>	31	<code>Nleft</code>	37
<code>\meta</code>	32	<code>Nright</code>	37
<code>\multicolplaincombinedlist</code>	24	O	
multicolumns options:		<code>O</code>	37
<code>addto-baselineskip</code>	16	<code>\oarg</code>	32
<code>aligned</code>	16	<code>\opt</code>	32
<code>balanced</code>	16	<code>\orbar</code>	33
<code>bottom-fuzz</code>	16	P	
<code>collectmore</code>	16	<code>\paragraph</code>	20
<code>cols</code>	15	<code>\parg</code>	32
<code>cols*</code>	16	<code>\part</code>	20
<code>column-badness</code>	16	<code>\pkg</code>	32
<code>column-sep</code>	15	<code>\PrintChanges</code>	32
<code>columns</code>	15	<code>\PrintIndex</code>	32
<code>columns*</code>	16	<code>\printindex</code>	20
<code>final-column-badness</code>	16	R	
<code>first-minimal</code>	16	<code>\removebackground</code>	18
<code>flush</code>	16	<code>\removeforeground</code>	18
<code>h-fuzz</code>	17	<code>\repinitval</code>	33
<code>heading</code>	16	<code>\Replicate</code>	2
<code>last-minimal</code>	16	<code>\resetval</code>	33
<code>left-to-right</code>	17	<code>\Rotate</code>	17
<code>LR</code>	17	rotate options:	
<code>minrows</code>	16	<code>figure</code>	17
<code>not-aligned</code>	16	<code>figure*</code>	17
<code>not-balanced</code>	16	<code>float</code>	17
<code>outer-sep</code>	15	<code>float*</code>	17
<code>overflow</code>	17	<code>nospaceturn</code>	17
<code>pretolerance</code>	16	<code>rotate</code>	17
<code>ragged</code>	16	<code>sideways</code>	17

table	17	TEX and L ^A T _E X 2 _ε commands:	
table*	17	\addcontentsline	29
turn	17	\cleaders	14
S		\cus@cbl@contentsline	30
\section.....	20, 23	\cus@cbl@contentsline@	30
\setcenterfoot	9	\cus@type@contentsline	29, 30
\setcenterhead	9	\cus@type@contentsline@	29, 30
\setfoot.....	9	\dotfill	1
\setfootinit	10	\hrulefill.....	11
\setfootrule	10	\hspace	11
\setfootruleskip.....	10	\hyperlink.....	30
\setfootrulewidth.....	9	\indexname.....	19
\sethead.....	9	\jobname	19
\setheadfoot	9	\leaders	14
\setheadfootinit.....	10	\leftmark	32
\setheadinit	10	\listoffigures	29
\setheadrule	10	\listoftables	29
\setheadruleskip.....	10	\markboth	32
\setheadrulewidth.....	9	\markright.....	32
\setindexinit	19	\rightmark.....	32
\setindexprologue.....	19	\tableofcontents	29
\setleftfoot	9	\thepage	30
\setlefthead	9	\xleaders	14
\setpagestyle	9	texbnf options:	
\setrightfoot	9	clear-all-format	37
\setrighthead	9	format	37
\setsecnumdepth	21	hyper	35
\setupindex	19	hyper-color	35
\setuplayout	3	Iformat	37
\setuptitle	20	Nformat	37
\standardplaincombinedlist.....	24	Oformat	37
\startmulticolumns.....	15	Tformat	37
\startrotate	17	texnote	34
\stopmulticolumns.....	15	title/... options:	
\stoprotate	17	afterindent	23
\subparagraph	20	aftername	22
\subsection	20	aftername+	22
\subsubsection	20	afterskip	22
syntax	33	aftertitle.....	22
T		aftertitle+	22
T.....	37	beforeskip.....	22
\tableofcontents.....	23	break	23
		break+	23

fixskip	22	style	23
format	22	titleformat	22
format+	22	titleformat+	22
hang	22	tocline	23
indent	22	\titleifname	23
level	21	Tleft	37
mark	23	\tn	32
name	21	\TopMark	31
nameformat	22	Tright	37
nameformat+	22		
number	21	U	
number-from	21	\ucchar	3
number-within	21	\ucchars	3
number-without	21	\usepagestyle	9
numberformat	22		
numberformat+	22	V	
numbering	21	\V	33
pagestyle	22	\Verbatimize	3
runin	22		
		Z	
		\zkern	3