

## Tabularray 基于 L<sup>A</sup>T<sub>E</sub>X3 的表格 (Tabulars) 和阵列 (Arrays) 宏包

Author Jianrui Lyu (tolvjr@163.com)

翻译 耿楠 (nangeng@nwafu.edu.cn)

Version 2022B (2022-06-05)

Code <https://github.com/lvjrr/tabularray>

Code <https://bitbucket.org/lvjrr/tabularray>

Forum <https://github.com/lvjrr/tabularray/discussions>

Forum <https://tex.stackexchange.com/questions/tagged/tabularray>

Issue <https://github.com/lvjrr/tabularray/issues>

译文 [https://gitee.com/nwafu\\_nan/tabularray-doc-zh-cn](https://gitee.com/nwafu_nan/tabularray-doc-zh-cn)

```
\begin{tblr}{
  colspec = {rX}, colsep = 8mm, hlines = {2pt, white},
  row{odd} = {azure8}, row{even} = {gray8},
  row{1} = {6em,azure2,fg=white,font=\LARGE\bfseries\sffamily},
  row{2-Z} = {3em,font=\Large},
}
Tabularray & 基于\LaTeX3 的表格 (Tabulars) 和阵列 (Arrays) 排版宏包 \\
Author     & Jianrui Lyu (tolvjr@163.com) \\
翻译      & 耿楠 (nangeng@nwafu.edu.cn) \\
Version   & \myversion\ (\the\year-\mylpad\month-\mylpad\day) \\
Code      & \url{https://github.com/lvjr/tabularray} \\
Code      & \url{https://bitbucket.org/lvjr/tabularray} \\
Forum     & \url{https://github.com/lvjr/tabularray/discussions} \\
Forum     & \url{https://tex.stackexchange.com/questions/tagged/tabularray} \\
Issue     & \url{https://github.com/lvjr/tabularray/issues} \\
译文      & \url{https://gitee.com/nwafu_nan/tabularray-doc-zh-cn} \\
\end{tblr}
```

# 目录

<b>第一章 概述</b>	<b>3</b>
1.1 垂直间距	3
1.2 多行单元格	4
1.3 单元格对齐方式	4
1.4 单元格行合并	5
1.5 单元格行合并和列合并	7
1.6 列格式	8
1.7 行格式	9
1.8 表格横线与表格竖线	9
1.9 彩色表格	10
<b>第二章 用户接口</b>	<b>12</b>
2.1 新旧用户接口	12
2.2 表格横线和竖线	12
2.3 水平和垂直边框	17
2.4 单元格与单元格合并选项	18
2.5 行 (rows) 和列 (columns) 选项	21
2.6 colspec 和 rowspec 键	24
<b>第三章 附加接口</b>	<b>27</b>
3.1 内部参数	27
3.2 外部参数	31
3.3 表格内容宏的提前展开	32
3.4 参数默认值	33
3.5 定义新 tabularray 环境	33
3.6 定义新通用环境	33
3.7 定义新表格命令	34
3.8 奇 (odd)/偶 (even) 选择器	34

目录	2
3.9 计数器和长度	35
3.10 跟踪 Tabularray	35
<b>第四章 长表格</b>	<b>36</b>
4.1 简单示例	36
4.2 个性化模板	41
4.3 改变样式	45
4.4 定义主题	46
4.5 分页控制	46
4.6 可浮动长表格 (talltblr)	46
4.7 移除长表格题注和尾注	47
<b>第五章 使用扩展库</b>	<b>48</b>
5.1 amsmath 库	48
5.2 booktabs 库	49
5.3 counter 库	51
5.4 diagbox 库	51
5.5 functional 库	52
5.6 siunitx 库	55
5.7 varwidth 库	57
<b>第六章 历史与未来</b>	<b>59</b>
6.1 未来	59
6.2 历史	59

# 第一章 概述

在使用 `tabulararray` 宏包前，建议先熟悉如何使用传统 `tabular`、`tabularx` 和 `array` 环境排版文本和数学表格，这是因为在此需要比较 `tabulararray` 宏包的 `tblr` 环境与这些环境的区别。可以通过阅读 [LearnLaTeX](#) 和 [Overleaf](#) 学习使用  $\text{\LaTeX}$  表格。

## 1.1 垂直间距

在导言区加载 `Tabulararray` 宏包后，可以使用 `tblr` 环境排版表格 (`tabulars`) 或阵列 (`arrays`)。环境名 `tblr` 是 `tabulararray` 或 `top-bottom-left-right` 的缩写。下例说明了常规 `tabular` 环境与 `tblr` 环境排版表格的异同：

```
\begin{tabular}{lccr}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
Epsilon & Zeta & Eta & Theta \\
\hline
Iota & Kappa & Lambda & Mu \\
\hline
\end{tabular}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

```
\begin{tblr}{lccr}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
Epsilon & Zeta & Eta & Theta \\
\hline
Iota & Kappa & Lambda & Mu \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

显然，用 `tblr` 环境排版表格时，在表格各行上下会增加额外垂直间距。该间距使表格排版更为美观。当然，可以使用 `\SetTblrDefault` 命令根据需要对该垂直间距进行调整，例如，可以用如下代码取消间距：

```

\SetTblrInner{rowsep=Opt}
\begin{tblr}{lccr}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
Epsilon & Zeta & Eta & Theta \\
\hline
Iota & Kappa & Lambda & Mu \\
\hline
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

但多数情况下，使用rowsep间距将使表格排版更为美观，例如如下带有分式的表格：

```

$\begin{array}{rrr}
\hline
\frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\
\frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\
\frac{1}{3} & -\frac{2}{3} & \frac{2}{3} \\
\hline
\end{array}$

```

$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$
$\frac{2}{3}$	$-\frac{1}{3}$	$-\frac{2}{3}$
$\frac{1}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$
$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{2}{3}$

```

$\begin{tblr}{rrr}
\hline
\frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\
\frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\
\frac{1}{3} & -\frac{2}{3} & \frac{2}{3} \\
\hline
\end{tblr}$

```

$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$
$\frac{2}{3}$	$-\frac{1}{3}$	$-\frac{2}{3}$
$\frac{1}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$
$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{2}{3}$

需要说明的是，tblr环境既可以用于文本模式，也可以用于数学模式。

## 1.2 多行单元格

使用tblr环境排版表格时，不使用定宽列格式就可以实现多行单元格排版。此时，仅需用大括号将单元格中需要多行排版的内容包围起来，然后在在大括号内中使用\\手动断行即可，如：

```

\begin{tblr}{|l|c|r|}
\hline
Left & {Center \\ Cent \\ C} & {Right \\ R} \\
\hline
{L \\ Left} & {C \\ Cent \\ Center} & R \\
\hline
\end{tblr}

```

Left	Center Cent C	Right R
L Left	C Cent Center	R

## 1.3 单元格对齐方式

Tabularray宏包提供了Q列格式用于同时指定单元格水平和垂直对齐方式。实质上，Q格式是该宏包提供的唯一列格式元格式，其它的列格式都是通过为Q元格式设置不同参数实现定义的。

```
\begin{tblr}{|Q[l,t]|Q[c,m]|Q[r,b]|}
\hline
{Top Baseline\\Left Left} & Middle Center & {Right Right\\Bottom Baseline} \\
\hline
\end{tblr}
```

		Right Right
Top Baseline	Middle Center	Bottom Baseline
Left Left		

注意，对于顶端基线对齐，可使用含义更为明确的t格式替代p格式。对于熟悉字处理工具的用户而言，类似t和b这些格式是违背直觉的，其中，t是顶端与基线对齐，不是顶端对齐，而b是底端与基线对齐，不是底端对齐。在Tabularray宏包中，定义了h和f格式，以分别实现常规意义上的顶端对齐和底端对齐：

```
\begin{tblr}{Q[h,4em]Q[t,4em]Q[m,4em]Q[b,4em]Q[f,4em]}
\hline
{row\\head} & {top\\line} & {middle} & {line\\bottom} & {row\\foot} \\
\hline
{row\\head} & {top\\line} & {11\\22\\mid\\44\\55}
& {line\\bottom} & {row\\foot} \\
\hline
\end{tblr}
```

row			line	
head	top	middle	bottom	row
	line			foot
<hr/>				
row		11		
head		22	line	
	top	mid	bottom	
	line	44		row
		55		foot

## 1.4 单元格行合并

在Tabularray中，可以使用\SetCell命令实现单元格行合并，此时，常用h和f列对齐格式。

```
\begin{tabular}{|l|l|l|l|}
\hline
\multirow[t]{4}{1.5cm}{Multirow Cell One}
& Alpha & & Alpha \\
\multirow[b]{4}{1.5cm}{Multirow Cell Two}
& Beta & & Beta \\
& Alpha & & \\
& Beta & & Beta \\
& Gamma & & Gamma \\
& Delta & & Delta \\
\hline
\end{tabular}
```

Multirow Cell One	Alpha		Alpha
	Beta		Beta
	Gamma	Multirow	Gamma
	Delta	Cell Two	Delta

```

\begin{tblr}{|l|l|l|l|}
\hline
\SetCell[r=4]{h,1.5cm} Multirow Cell One
& Alpha &
\SetCell[r=4]{f,1.5cm} Multirow Cell Two
& Alpha \\
& Beta & & Beta \\
& Gamma & & Gamma \\
& Delta & & Delta \\
\hline
\end{tblr}

```

Multirow Cell One	Alpha	Multirow Cell Two	Alpha
	Beta		Beta
	Gamma		Gamma
	Delta		Delta

注意，由于Tabularray宏包并不依赖multirow宏包，因此，在行合并前并不需要载入multirow宏包。此外，Tabularray宏包将自动处理单元格行合并后的垂直对齐，这使得即便是在某些行的行高较大，也能够正确地实现行合并后的垂直居中对齐。

```

\begin{tabular}{|l|m{4em}|}
\hline
\multirow[c]{4}{1.5cm}{Multirow} & Alpha \\
& Beta \\
& Gamma \\
& Delta Delta Delta \\
\hline
\end{tabular}

```

Multirow	Alpha
	Beta
	Gamma
	Delta

```

\begin{tblr}{|l|m{4em}|}
\hline
\SetCell[r=4]{m,1.5cm} Multirow & Alpha \\
& Beta \\
& Gamma \\
& Delta Delta Delta \\
\hline
\end{tblr}

```

Multirow	Alpha
	Beta
	Gamma
	Delta

同时，如果行合并后的文字累计行高大于表格行高，则Tabularray会自动扩展合并后单元格行高，从而确保不会发生文字垂直溢出现象。

```

\begin{tabular}{|l|m{4em}|}
\hline
\multirow[c]{2}{1cm}{Line\\Line\\Line\\Line} & Alpha \\
\cline{2-2}
& Beta \\
\hline
\end{tabular}

```

Line	Alpha
Line	Beta

Line  
Line



```

\begin{tblr}{|l|m{4em}|}
\hline
\SetCell[r=2]{m,1cm} {Line\\Line\\Line\\Line} & Alpha \\
\cline{2}
& Beta \\
\hline
\end{tblr}

```

Line	Alpha
Line	Beta
Line	
Line	

如果想把额外的垂直空间均匀地分配给两行。可以使用第三章说明的 `vspan` 选项。

## 1.5 单元格行合并和列合并

在传统 `tabular` 表格中，同时实现单元格行合并和列合并是比较困难的，例如：

```

\begin{tabular}{|c|c|c|c|c|}
\hline
\multirow{2}{*}{2 Rows}
& \multicolumn{2}{c|}{2 Columns}
& \multicolumn{2}{c|}
& \multicolumn{2}{*}{2 Rows 2 Columns} \\
\cline{2-3}
& 2-2 & 2-3 & \multicolumn{2}{c|}{} \\
\hline
3-1 & 3-2 & 3-3 & 3-4 & 3-5 \\
\hline
\end{tabular}

```

2 Rows	2 Columns		2 Rows 2 Columns	
	2-2	2-3		
3-1	3-2	3-3	3-4	3-5

但在 `Tabularray` 宏包中，可以简单地使用 `\SetCell` 命令实现单元格合并。在 `\SetCell` 命令的可选参数中，`r` 选项用于指定需要合并的行数，`c` 选项用于指定需要合并的列数。在 `\SetCell` 命令的必选参数中，可以指定合并后的水平和垂直对齐方式。因此，在 `Tabularray` 中，单元格合并将更为简捷：

```

\begin{tblr}{|c|c|c|c|c|}
\hline
\SetCell[r=2]{c} 2 Rows & \SetCell[c=2]{c} 2 Columns
& & \SetCell[r=2,c=2]{c} 2 Rows 2 Columns & \\
\hline
& 2-2 & 2-3 & & \\
\hline
3-1 & 3-2 & 3-3 & 3-4 & 3-5 \\
\hline
\end{tblr}

```

2 Rows	2 Columns		2 Rows 2 Columns	
	2-2	2-3		
3-1	3-2	3-3	3-4	3-5

在使用`\multicolumn`命令时, **必须删除**需要合并的其它单元格。相反, 在使用`\multirow`命令时, 则**必须保留**需要合并的其它单元格。`\SetCell`命令则与`\multirow`命令的行为相同。

在使用`tblr`环境中, 由于会自动忽略合并单元格的`\hline`线段, 因此, 在上述示例中, 可以直接使用`\hline`命令绘制表格横线。同时, 无论单元格是否留空, 任何合并时省略的单元格在排版时都会被忽略。基于此, 在排版表格时, 可以将行号、列号或其它标识性内容写入需要省略的单元格, 这有助于在排版复杂表格时的单元格定位, 如:

```
\begin{tblr}{|l|c|rr|}
\hline
\SetCell[r=3,c=2]{h} r=3 c=2 & 1-2 & \SetCell[r=2,c=3]{r} r=2 c=3 & 1-4 & 1-5 \\
2-1 & 2-2 & 2-3 & 2-4 & 2-5 \\
\hline
3-1 & 3-2 & MIDDLE & \SetCell[r=3,c=2]{f} r=3 c=2 & 3-5 \\
\hline
\SetCell[r=2,c=3]{l} r=2 c=3 & 4-2 & 4-3 & 4-4 & 4-5 \\
5-1 & 5-2 & 5-3 & 5-4 & 5-5 \\
\hline
\end{tblr}
```

r=3 c=2	r=2 c=3	
	MIDDLE	
r=2 c=3		r=3 c=2

## 1.6 列格式

`Tabularray`宏包支持所有常规的列格式, 包括第一次出现在`tabularx`宏包中, 并被`tabu`宏包优化后的可扩展X列格式。

```
\begin{tblr}{|X[2,1]|X[3,1]|X[1,r]|X[r]|}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta
-------	------	-------	-------

此外, 还可以在X列格式的比例系数可以取负值:

```
\begin{tblr}{|X[2,1]|X[3,1]|X[-1,r]|X[r]|}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta
-------	------	-------	-------

在使用X列格式排版表格时, 需要指定表格宽度。如未设置, 则默认为`\linewidth`。如需更改表格宽度, 则必须将所有列格式置入`colspec={...}`选项的参数中:

```
\begin{tblr}{width=0.8\linewidth,colspec={|X[2,1]|X[3,1]|X[-1,r]|X[r]|}}
\hline
Alpha & Beta & Gamma & Delta \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta
-------	------	-------	-------

可以使用`\NewColumnType`命令定义新的列格式。例如，在`Tabularray`宏包中，`b`和`X`列格式是用`Q`元格式通过指定必要参数实现的：

```
\NewColumnType{b}[1]{Q[b,wd=#1]}
\NewColumnType{X}[1]{}{Q[co=1,#1]}
```

## 1.7 行格式

除了可以使用`colspec`选项指定列格式外，也可以通过`rowspec`选项指定行格式，如：

```
\begin{tblr}{colspec={Q[1]Q[c]Q[r]},rowspec={|Q[t]|Q[m]|Q[b]|}}
{Alpha \\ Alpha} & Beta & & Gamma \\
Delta & & Epsilon & {Zeta \\ Zeta} \\
Eta & & {Theta \\ Theta} & Iota \\
\end{tblr}
```

Alpha	Beta	Gamma
Alpha		
Delta	Epsilon	Zeta
		Zeta
	Theta	
Eta	Theta	Iota

与列格式类似，`Q`是唯一的行格式的元格式，其它行格式都是通过为`Q`元格式指定不同参数实现的。强烈建议在`colspec`中指定水平对齐方式，在`rowspec`中指定垂直对齐方式。

在`rowspec`中，`l`用于指定表格横线格式。因此，无需再在表格内容中使用`\hline`命令，这会使表格代码更为清晰、可维护性更强。

## 1.8 表格横线与表格竖线

在`Tabularray`宏包中，重新设计了表格横线和竖线命令，可以通过命令的 `keyval` 选项指定其的宽度、线型、颜色等样式：

```

\begin{tblr}{|l|[dotted]|2ptc|r|[solid]|[dashed]|}
\hline
One & Two & Three \\
\hline\hline[dotted]\hline
Four & Five & Six \\
\hline[dashed]\hline[1pt]
Seven & Eight & Nine \\
\hline
\end{tblr}

```

One	Two	Three
Four	Five	Six
Seven	Eight	Nine

## 1.9 彩色表格

如需排版彩色表格，则需加载 `xcolor`宏包，`Tabularray`宏包一旦发现用户加载了`xcolor`宏包，则会自动加载`ninecolors`宏包，以使用前景与背景对比度更为合理的预定义颜色名称。例如，可以在`rowspec/colspec`选项中通过Q元格式的背景参数为行/列指定背景色：

```

\begin{tblr}{colspec={lcr},
rowspec={|Q[cyan7]|Q[azure7]|Q[blue7]|}}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

```

\begin{tblr}{%
colspec={|Q[l,brown7]Q[c,yellow7]Q[r,olive7]|},
rowspec={|Q|Q|Q|}}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

当然，也可以使用`\SetRow`或`\SetColumn`命令统一为指定的行或列设置颜色：

```

\begin{tblr}{colspec={lcr},rowspec={|Q|Q|Q|}}
\SetRow{cyan7} Alpha & Beta & Gamma \\
\SetRow{azure7} Epsilon & Zeta & Eta \\
\SetRow{blue7} Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

```

\begin{tblr}{colspec={lcr},rowspec={|Q|Q|Q|}}
\SetColumn{brown7}
Alpha & \SetColumn{yellow7}
Beta & \SetColumn{olive7}
Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

此外，还可以为表格横线和竖线指定颜色：

```

\begin{tblr}{colspec={lcr},
rowspec={|[2pt,green7]Q|[teal7]Q|[green7]Q|[3pt,teal7]}}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

```

\begin{tblr}{colspec={|[2pt,violet5]l|[2pt,magenta5]c|
[2pt,purple5]r|[2pt,red5]}}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

## 第二章 用户接口

### 2.1 新旧用户接口

在 `Tabularray` 宏包中，通过用户接口可以实现表格样式控制。

旧的用户接口由嵌入表格内容中的一系列命令构成。此时，与 `tabular` 和 `array` 环境类似，所有的命令**必须**置于单元格文本内容之前。另外，如果需要新表格命令，则**必须**使用 `\NewTableCommand` 命令进行定义。

新的用户接口由 `tblr` 环境的必选参数中的 `keyval` 实现，因此，使用新用户接口可以实现表格内容与格式的完全分离。

新旧接口的对应关系如2.1所示。

表 2.1: 新旧用户接口

旧接口	新接口
<code>\SetHlines</code>	<code>hlines</code>
<code>\SetHline</code> , <code>\hline</code> , <code>\hborder</code> , <code>\cline</code>	<code>hline</code> , <code>hborder</code> , <code>rowspec</code>
<code>\SetVlines</code>	<code>vlines</code>
<code>\SetVline</code> , <code>\vline</code> , <code>\vborder</code> , <code>\rline</code>	<code>vline</code> , <code>vborder</code> , <code>colspec</code>
<code>\SetCells</code>	<code>cells</code>
<code>\SetCell</code>	<code>cell</code>
<code>\SetRows</code>	<code>rows</code>
<code>\SetRow</code>	<code>row</code> , <code>rowspec</code>
<code>\SetColumns</code>	<code>columns</code>
<code>\SetColumn</code>	<code>column</code> , <code>colspec</code>

### 2.2 表格横线和竖线

表格横线 (`hlines`) 和竖线 (`vlines`) 选项所有有效键及其键值见表2.2和表2.3。

表 2.2: Hlines 的键值

键	说明与可选键值	初始值
<code>dash</code>	线型: <code>solid</code> 、 <code>dashed</code> 或 <code>dotted</code>	<code>solid</code>
<code>text</code>	用文本替换 <code>hline</code> (与在 <code>rowspec</code> 中将横线指定为 <code>!</code> 一样)	<code>×</code>
<code>wd</code>	线宽	<code>0.4pt</code>
<code>fg</code>	表线颜色	<code>×</code>
<code>leftpos</code>	左侧的相交或截断位置	<code>1</code>
<code>rightpos</code>	右侧的相交或截断位置	<code>1</code>
<code>endpos</code>	是否修正最左列左边/最右列右边位置	<code>false</code>

注意: 多数情况下, 对于带有下划线的键, 可以省略键名而只给出键值。

表 2.3: Vlines 的键值

键	说明与可选键值	初始值
<code>dash</code>	线型: <code>solid</code> 、 <code>dashed</code> 或 <code>dotted</code>	<code>solid</code>
<code>text</code>	用文本替换 <code>vline</code> (与在 <code>colspec</code> 中将竖线指定为 <code>!</code> 一样)	<code>×</code>
<code>wd</code>	线宽	<code>0.4pt</code>
<code>fg</code>	颜色	<code>×</code>
<code>abovepos</code>	上侧的相交或截断位置	<code>0</code>
<code>belowpos</code>	下侧的相交或截断位置	<code>0</code>

注意: 多数情况下, 对于带有下划线的键, 可以省略键名而只给出键值。

### 2.2.1 新接口中的表格横线与竖线

`hlines` 和 `vlines` 分别用于设置表格所有横线与竖线样式。如果其键值留空, 则会把表格所有横线/竖线都设置为实线。

```
\begin{tblr}{hlines,vlines}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

可以在一对大括号中通过指定横线/竖线样式。

```
\begin{tblr}{
hlines = {1pt,solid}, vlines = {red3,dashed},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

也可以在大括号前用另一对大括号指定需要设置单元格横线/竖线线段编号。

```

\begin{tblr}{
  vlines = {1,3,5}{dashed},
  vlines = {2,4}{solid},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
Nu & Xi & Omicron & Pi & \\
Rho & Sigma & Tau & Upsilon & \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi
Rho	Sigma	Tau	Upsilon

以上示例也可以简单地通过指定`odd`和`even`选择算子进行选择（可使用`\NewChildSelector`命令定义更多选择算子，高级用户请参阅 `Tabularray`源代码，通过模仿定义需要的选择算子）。

```

\begin{tblr}{
  vlines = {odd}{dashed},
  vlines = {even}{solid},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
Nu & Xi & Omicron & Pi & \\
Rho & Sigma & Tau & Upsilon & \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi
Rho	Sigma	Tau	Upsilon

可以再增加一对大括号用于设置多重横线/竖线（其中，`-`表示选择所有单元格线段）。

```

\begin{tblr}{
  hlines = {1}{-}{dashed},
  hlines = {2}{-}{solid},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

注意：必须先使用 1，然后是 2 等这样的顺序设置多重横线/竖线。

`hline{i}`和`vline{j}`键分别用于设置指定的横线/竖线，`i`或`j`的值与 `hlines`和 `vlines` 的参数含义相同：

```

\begin{tblr}{
  hline{1,7} = {1pt,solid},
  hline{3-5} = {blue3,dashed},
  vline{1,5} = {3-4}{dotted},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
Nu & Xi & Omicron & Pi & \\
Rho & Sigma & Tau & Upsilon & \\
Phi & Chi & Psi & Omega & \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi
Rho	Sigma	Tau	Upsilon
Phi	Chi	Psi	Omega



也可以使用X、Y和Z分别指定最后三条横线/竖线的样式，这在排版多行/多列表格时特别有用。

```

\begin{tblr}{
  hline{1,Z} = {2pt},
  hline{2,Y} = {1pt},
  hline{3-X} = {dashed},
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
Nu & Xi & Omicron & Pi \\
Rho & Sigma & Tau & Upsilon \\
Phi & Chi & Psi & Omega \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi
Rho	Sigma	Tau	Upsilon
Phi	Chi	Psi	Omega

下面的示例演示了 `text` 的使用方式<sup>1</sup>：

```

\begin{tblr}{
  vlines, hlines,
  colspec = {1X[c]X[c]X[c]X[c]},
  vline{2} = {1}{text=\clap{:}},
  vline{3} = {1}{text=\clap{\ch{+}}},
  vline{4} = {1}{text=\clap{\ch{->}}},
  vline{5} = {1}{text=\clap{\ch{+}}},
}
Equation & \ch{CH4} & \ch{2 O2} & \ch{CO2} & \ch{2 H2O} \\
Initial & $n_1$ & $n_2$ & 0 & 0 \\
Final & $n_1-x$ & $n_2-2x$ & $x$ & $2x$ \\
\end{tblr}

```

Equation :	CH <sub>4</sub>	+	2 O <sub>2</sub>	→	CO <sub>2</sub>	+	2 H <sub>2</sub> O
Initial	$n_1$		$n_2$		0		0
Final	$n_1 - x$		$n_2 - 2x$		$x$		$2x$

注意，为使用 `\ch` 命令，必须载入 `chemmacros` 宏包。

`leftpos` 和 `rightpos` 用于指定 `hlines` 的相交或截断位置，其取值为 -1 和 1 之间的十进制数。它们的初始值是 1。

---

-1 使用 `colsep` 对 `hline` 进行截断

---

0 `hline` 仅与第 1 条 `vline` 相交

---

1 `hline` 与所有 `vlines` 相交

`vlines` 的 `abovepos` 和 `belowpos` 键具备类似的含义，但其初始值为 0。

---

-1 使用 `rowsep` 对 `vline` 进行截断

---

0 `vline` 仅与第 1 条 `hline` 相交

---

1 `vline` 与所有 `hlines` 相交

<sup>1</sup>代码来自 <https://tex.stackexchange.com/questions/603023/tabularray-and-tabularx-column-separator>。

以下是这四个键的一个应用实例:

```
\begin{tblr}{
  hline{1,4} = {1}{-}{},
  hline{1,4} = {2}{-}{},
  hline{2,3} = {1}{-}{leftpos = -1, rightpos = -1},
  hline{2,3} = {2}{-}{leftpos = -1, rightpos = -1},
  vline{1,4} = {abovepos = 1, belowpos = 1},
}
Alpha & Beta & Gamma & \\
Epsilon & Zeta & Eta & \\
Iota & Kappa & Lambda & \\
\end{tblr}
```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

`endpos`选项用于设置是否调整最左列的 `leftpos` 或最右列的 `rightpos`:

```
\begin{tblr}{
  hline{1,4} = {1}{-}{},
  hline{1,4} = {2}{-}{},
  hline{2,3} = {leftpos = -1, rightpos = -1, endpos},
  vline{1,4} = {abovepos = 1, belowpos = 1},
}
Alpha & Beta & Gamma & \\
Epsilon & Zeta & Eta & \\
Iota & Kappa & Lambda & \\
\end{tblr}
```

Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

## 2.2.2 旧接口中的表格横线与竖线

可以在`\hline`命令中通过选项指定横线样式，其有效的键值见表2.2.

```
\begin{tblr}{l1l1l1}
\hline
Alpha & Beta & Gamma & Delta & \\
\hline[dashed]
Epsilon & Zeta & Eta & Theta & \\
\hline[dotted]
Iota & Kappa & Lambda & Mu & \\
\hline[2pt,blue5]
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

`\cline`命令的选项与`\hline`相同。

```
\begin{tblr}{l1l1l1}
\cline{1-4}
Alpha & Beta & Gamma & Delta & \\
\cline[dashed]{1,3}
Epsilon & Zeta & Eta & Theta & \\
\cline[dashed]{2,4}
Iota & Kappa & Lambda & Mu & \\
\cline[2pt,blue5]{-}
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

也可以在 `\cline` 命令的必选参数中通过选择算子实现行/列选择。

```

\begin{tblr}{l1111}
\cline{1-4}
Alpha & Beta & Gamma & Delta \\
\cline[dashed]{odd}
Epsilon & Zeta & Eta & Theta \\
\cline[dashed]{even}
Iota & Kappa & Lambda & Mu \\
\cline[2pt,blue5]{-}
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

\SetHline命令是一个组合了 \hline 和 \cline功能的命令:

```

\begin{tblr}{l1111}
\SetHline{1-3}{blue5,1pt}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\SetHline{2-4}{teal5,1pt}
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

```

\begin{tblr}{l1111}
\SetHline[1]{1-3}{blue5,1pt}
\SetHline[2]{1-3}{azure5,1pt}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\SetHline[1]{2-4}{teal5,1pt}
\SetHline[2]{2-4}{green5,1pt}
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

本质上, 在第*i*行前使用 \SetHline[<index>]{<columns>}{<styles>}命令与 `hline{i}={<index>}{<columns>}{<styles>}`键的作用完全相同。

同样, 在某些行前使用 \SetHlines[<index>]{<columns>}{<styles>}命令与 `hlines={<index>}{<columns>}{<styles>}`键的作用完全相同。

\vline、\rline、\SetVline和 \SetVlines命令的使用方法分别与\hline、\cline、\SetHline、\SetHlines的使用方法相同。但通常情况下, 一般不建议直接使用这些命令。

## 2.3 水平和垂直边框

选项 `hborder{i}` 和 `vborder{j}` 分别与 `hline{i}` 和 `vline{j}`类似, 但它们指定的边框线与 `hline` 和 `vline` 无关。所有有效的 `hborder{i}` 和 `vborder{j}`键见表 2.4 和 2.5。

表 2.4: 水平边框键

键	含义	初始值
<code>pagebreak</code>	在该位置分页: yes, no or auto (见第 四章)	auto

续下页

表 2.4: 水平边框键 (接前页)

键	含义	初始值
<code>abovespace</code>	设置前一行的 <code>belowsep</code> (见表 2.8)	2pt
<code>belowspace</code>	设置当前行的 <code>abovesep</code> (见表 2.8)	2pt
<code>abovespace+</code>	增加前一行的 <code>belowsep</code>	×
<code>belowspace+</code>	增加当前行的 <code>abovesep</code>	×

表 2.5: 垂直边框键

键	定义和值	初始值
<code>leftspace</code>	设置前一列的 <code>rightsep</code> (见表 2.9)	6pt
<code>rightspace</code>	设置当前列的 <code>leftsep</code> (见表 2.9)	6pt
<code>leftspace+</code>	增加前一列的 <code>rightsep</code>	×
<code>rightspace+</code>	增加当前列的 <code>leftsep</code>	×

进一步来讲, 在第*i*行开始处的 `\hborder{<specs>}` 命令与表格的 `hborder{i}={<specs>}` 选项作用相同, 在第*j*列开始处的 `\vborder{<specs>}` 命令与表格的 `vborder{j}={<specs>}` 选项作用相同,

## 2.4 单元格与单元格合并选项

单元格所有有效键及键值见表 2.6 和表 2.7.

表 2.6: `cells` 键与键值

键	说明与可选键值	初始值
<code>halign</code>	水平对齐方式: l (left), c (center), r (right) 或 j (justify)	j
<code>valign</code>	垂直对齐方式: t (top), m (middle), b (bottom), h (head) 或 f (foot)	t
<code>wd</code>	宽度	×
<code>bg</code>	背景颜色	×
<code>fg</code>	前景颜色	×
<code>font</code>	字体命令	×
<code>mode</code>	单元格模式: <code>math</code> , <code>imath</code> , <code>dmath</code> 或 <code>text</code>	×
<code>\$</code>	与 <code>mode=math</code> 相同	×
<code>\$\$</code>	与 <code>mode=dmath</code> 相同	×
<code>cmd</code>	对单元格文本要执行的命令	×
<code>preto</code>	单元格前导文本	×

续下页

表 2.6: cells 键与键值 (接前页)

键	说明与可选键值	初始值
appto	单元格附加文本	×

注意: 多数情况下, 对于带下划线的键, 可以省略键名而只给出键值。

表 2.7: 单元格合并键与键值

键	说明与可选键值	初始值
r	合并行数	1
c	合并列数	1

### 2.4.1 单元格与单元格合并新接口

cells键用于设置所有单元格的样式。

```
\begin{tblr}{hlines={white},cells={c,blue7}}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
Nu & Xi & Omicron & Pi \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi

cell{i}{j}键用于设置指定行列位置单元格的样式, 其中, i表示行索引, j表示列索引。

```
\begin{tblr}{
cell{1,6}{2-4} = {cmd=\fbox}
}
Alpha & Beta & Gamma & Delta \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
-------	------	-------	-------

```
\begin{tblr}{
hlines = {white},
vlines = {white},
cell{1,6}{odd} = {teal7},
cell{1,6}{even} = {green7},
cell{2,4}{1,4} = {red7},
cell{3,5}{1,4} = {purple7},
cell{2}{2} = {r=4,c=2}{c,azure7},
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
Nu & Xi & Omicron & Pi \\
Rho & Sigma & Tau & Upsilon \\
Phi & Chi & Psi & Omega \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta		Theta
Iota	Zeta		Mu
Nu	Zeta		Pi
Rho	Zeta		Upsilon
Phi	Chi	Psi	Omega

## 2.4.2 单元格与单元格合并旧接口

`\SetCell`命令的必选参数用于设置当前单元格的样式。其有效值见表 2.6.

```
\begin{tblr}{|l|l|l|}
\hline[1pt]
Alpha & \SetCell{bg=teal2,fg=white} Beta & Gamma \\
\hline
Epsilon & Zeta & \SetCell{r,font=\scshape} Eta \\
\hline
Iota & Kappa & Lambda \\
\hline[1pt]
\end{tblr}
```

Alpha	Beta	Gamma
Epsilon	Zeta	ETA
Iota	Kappa	Lambda

`\SetCell`命令也可以使用可选参数设置当前单元格需要合并的列数和行数。其有效值见表 2.7.

```
\begin{tblr}{|X|X|X|X|X|X|}
\hline
Alpha & Beta & Gamma & Delta & Epsilon & Zeta \\
\hline
\SetCell[c=2]{c} Eta & 2-2 & & & & \\
& & \SetCell[c=2]{c} Iota & 2-4 & & \\
& & & & \SetCell[c=2]{c} Lambda & 2-6 \\
\hline
\SetCell[c=3]{c} Nu & 3-2 & 3-3 & & & \\
& & & \SetCell[c=3]{c} Pi & 3-5 & 3-6 \\
\hline
\SetCell[c=6]{c} Tau & 4-2 & 4-3 & 4-4 & 4-5 & 4-6 \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta	Epsilon	Zeta
Eta		Iota		Lambda	
Nu			Pi		
Tau					

```
\begin{tblr}{|X|X|X|X|X|X|}
\hline
Alpha & Beta & Gamma & Delta & Epsilon & Zeta \\
\hline
\SetCell[r=2]{m} Eta & & & & & \\
& Theta & Iota & Kappa & Lambda & \SetCell[r=2]{m} Mu \\
\hline
Nu & Xi & Omicron & Pi & Rho & Sigma \\
\hline
\end{tblr}
```

Alpha	Beta	Gamma	Delta	Epsilon	Zeta
Eta	Theta	Iota	Kappa	Lambda	Mu
	Xi	Omicron	Pi	Rho	

本质上，在第*i*行、第*j*列的单元格前使用`\SetCell[<span>]{<styles>}`命令与`cell{i}{j}={<span>}{<styles>}`的作用完全相同。

同样，表格命令 `\SetCells[<span>]{<styles>}` 与 `cells={<span>}{<styles>}`的作用完全相同。

## 2.5 行 (rows) 和列 (columns) 选项

行 (rows) 和列 (columns) 选项的键和有效键值见表 2.8和表 2.9。

表 2.8: rows 选项的键和键值

键	说明与可选键值	初始值
<u>halign</u>	水平对齐方式: l (left), c (center), r (right) 或 j(justify)	j
<u>valign</u>	垂直对齐方式: t (top), m (middle), b (bottom), h (head) 或 f (foot)	t
<u>ht</u>	行高	×
<u>bg</u>	背景颜色	×
<u>fg</u>	前景颜色	×
<u>font</u>	字体命令	×
<u>mode</u>	行模式: math, imath, dmath or text	×
\$	与 mode=math 相同	×
\$\$	与 mode=dmath 相同	×
<u>cmd</u>	对每个单元格文本要执行的命令	×
<u>abovesep</u>	行前垂直间距	2pt
<u>abovesep+</u>	行前垂直间距增量	×
<u>belowsep</u>	行后垂直间距	2pt
<u>belowsep+</u>	行后垂直间距增量	×
<u>rowsep</u>	行前行后垂直间距	2pt
<u>rowsep+</u>	行前行后垂直间距增量	×
<u>preto</u>	单元格前导文本 (如 rowspec 选项中的 >)	×
<u>appto</u>	单元格附加文本 (如 rowspec 选项中的 <)	×

注意: 多数情况下, 对于带有下划线的键, 可以省略键名而只给出键值。

表 2.9: columns 选项的键和键值

键	说明与可选键值	初始值
<u>halign</u>	水平对齐方式: l (left)、c (center)、r (right) 或 j(justify)	j
<u>valign</u>	垂直对齐方式: t (top)、m (middle)、b (bottom)、h (head) 或 f (foot)	t

续下页

表 2.9: columns 选项的键和键值 (接前页)

键	说明与可选键值	初始值
<u>wd</u>	列宽	×
<u>co</u>	可扩展列的扩展系数 (X 列)	×
<u>bg</u>	背景颜色	×
<u>fg</u>	前景颜色	×
<u>font</u>	字体命令	×
<u>mode</u>	列模式: math, imath, dmath or text	×
<u>\$</u>	与 mode=math 相同	×
<u>\$\$</u>	与 mode=dmath 相同	×
<u>cmd</u>	对每个单元格文本要执行的命令	×
<u>leftsep</u>	列左侧水平间距	6pt
<u>leftsep+</u>	列左侧水平间距增量	×
<u>rightsep</u>	列右侧水平间距	6pt
<u>rightsep+</u>	列右侧水平间距增量	×
<u>colsep</u>	列左右水平间距	6pt
<u>colsep+</u>	列左右水平间距增量	×
<u>preto</u>	单元格前导文本 (如 colspec 选项中的 >)	×
<u>appto</u>	单元格附加文本 (如 colspec 选项中的 <)	×

注意: 多数情况下, 对于带有下划线的键, 可以省略键名而只给出键值。

### 2.5.1 行和列设置新接口

rows 和 columns 分别用于设置表格的所有行/列格式。

```
\begin{tblr}{
  hlines, vlines,
  rows = {7mm}, columns = {15mm,c},
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

row{i} 和 column{j} 键分别用于设置指定行/列格式。



```

\begin{tblr}{
  hlines = {1pt,white},
  row{odd} = {blue7},
  row{even} = {azure7},
  column{1} = {purple7,c},
}
Alpha & Beta & Gamma & Delta & \\
Epsilon & Zeta & Eta & Theta & \\
Iota & Kappa & Lambda & Mu & \\
Nu & Xi & Omicron & Pi & \\
Rho & Sigma & Tau & Upsilon & \\
Phi & Chi & Psi & Omega & \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi
Rho	Sigma	Tau	Upsilon
Phi	Chi	Psi	Omega

以下示例演示了 `bg`、`fg` 和 `font` 键的基本用法：

```

\begin{tblr}{
  row{odd} = {bg=azure8},
  row{1} = {bg=azure3, fg=white, font=\sffamily},
}
Alpha & Beta & Gamma & \\
Delta & Epsilon & Zeta & \\
Eta & Theta & Iota & \\
Kappa & Lambda & Mu & \\
Nu Xi Omicron & Pi Rho Sigma & Tau Upsilon Phi & \\
\end{tblr}

```

Alpha	Beta	Gamma
Delta	Epsilon	Zeta
Eta	Theta	Iota
Kappa	Lambda	Mu
Nu Xi Omicron	Pi Rho Sigma	Tau Upsilon Phi

以下示例演示了 `mode` 键的用法。

```

\begin{tblr}{
  column{1} = {mode=text},
  column{3} = {mode=dmath},
}
\hline
Alpha & \frac{1}{2} & \frac{1}{2} & \\
Epsilon & \frac{3}{4} & \frac{3}{4} & \\
Iota & \frac{5}{6} & \frac{5}{6} & \\
\hline
\end{tblr}

```

Alpha	$\frac{1}{2}$	$\frac{1}{2}$
Epsilon	$\frac{3}{4}$	$\frac{3}{4}$
Iota	$\frac{5}{6}$	$\frac{5}{6}$

以下示例演示了 `abovesep`、`belowsep`、`leftsep`、`rightsep` 键的用法：

```

\begin{tblr}{
  hlines, vlines,
  rows = {abovesep=1pt,belowsep=5pt},
  columns = {leftsep=1pt,rightsep=5pt},
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

以下示例演示了用`belowsep+` 替代 `\\[dimen]` 命令的方法。

```

\begin{tblr}{
  hlines, row{2} = {belowsep+=5pt},
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

## 2.5.2 行和列设置旧接口

可以通过`\SetRow` 命令的必选参数设置当前行的格式。其有效值见表 2.8.

```

\begin{tblr}{llll}
\hline[1pt]
\SetRow{azure8} Alpha & Beta & Gamma & Delta \\
\hline
\SetRow{blue8,c} Epsilon & Zeta & Eta & Theta \\
\hline
\SetRow{violet8} Iota & Kappa & Lambda & Mu \\
\hline[1pt]
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

本质上，在第*i*行开始的命令`\SetRow{<styles>}`与 `keyval row{i}={<styles>}`的作用相同。

另外，在一些行开始的命令`\SetRows{<styles>}`与 `keyval rows={<styles>}`的作用相同。

`\SetColumn` 和 `\SetColumns`表格命令的用法分别与 `\SetRow` 和 `\SetRows`命令类似。但一般不直接使用这两个命令。

## 2.6 colspec 和 rowspec 键

`colspec/rowspec` 用于使用 `column/row` 格式参数设置 `column/row` 的格式。

### 2.6.1 colspec 键和 width 键

width 用于设置具备可扩展列的表格总宽度。下面的示例演示了 width 的用法:

```
\begin{tblr}{width=0.8\textwidth, colspec={|l|X[2]|X[3]|X[-1]|}}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

如果在必选参数使用唯一的键, 则可以省略 colspec 名称。下面的示例演示了 \$ 的用法:

```
\begin{tblr}{Q[1]Q[r,$]Q[r,$$]}
\hline
Alpha & \frac{1}{2} & \frac{1}{2} \\
Epsilon & \frac{3}{4} & \frac{3}{4} \\
Iota & \frac{5}{6} & \frac{5}{6} \\
\hline
\end{tblr}
```

Alpha	$\frac{1}{2}$	$\frac{1}{2}$
Epsilon	$\frac{3}{4}$	$\frac{3}{4}$
Iota	$\frac{5}{6}$	$\frac{5}{6}$

### 2.6.2 列格式

Tabularray 宏包仅设计了一个列格式的 Q 元格式。其它的列格式都是通过为 Q 元格式指定不同的参数实现定义。

```
\NewColumnType{l}{Q[l]}
\NewColumnType{c}{Q[c]}
\NewColumnType{r}{Q[r]}
\NewColumnType{t}[1]{Q[t,wd=#1]}
\NewColumnType{m}[1]{Q[m,wd=#1]}
\NewColumnType{b}[1]{Q[b,wd=#1]}
\NewColumnType{h}[1]{Q[h,wd=#1]}
\NewColumnType{f}[1]{Q[f,wd=#1]}
\NewColumnType{X}[1][]{Q[co=1,#1]}
```

```
\begin{tblr}{|t{15mm}|m{15mm}|b{20mm}|}
Alpha & Beta & {Gamma\\Gamma} \\
Epsilon & Zeta & {Eta\\Eta} \\
Iota & Kappa & {Lambda\\Lambda} \\
\end{tblr}
```

Alpha	Beta	Gamma Gamma
Epsilon	Zeta	Eta Eta
Iota	Kappa	Lambda Lambda

任何新的列格式都需要使用 \NewColumnType 命令定义。在定义时, 可以使用可选参数。

### 2.6.3 行格式

同样，`Tabularray` 宏包仅设计了一个行格式的Q 元格式。其它的行格式都是通过为Q 元格式指定不同的参数实现定义。

```
\NewRowType{l}{Q[l]}
\NewRowType{c}{Q[c]}
\NewRowType{r}{Q[r]}
\NewRowType{t}[1]{Q[t,ht=#1]}
\NewRowType{m}[1]{Q[m,ht=#1]}
\NewRowType{b}[1]{Q[b,ht=#1]}
\NewRowType{h}[1]{Q[h,ht=#1]}
\NewRowType{f}[1]{Q[f,ht=#1]}
```

```
\begin{tblr}{rowspec={|t{12mm}|m{10mm}|b{10mm}|}}
Alpha & Beta & {Gamma\\Gamma} \\
Epsilon & Zeta & {Eta\\Eta} \\
Iota & Kappa & {Lambda\\Lambda} \\
\end{tblr}
```

Alpha	Beta	Gamma Gamma
Epsilon	Zeta	Eta Eta
Iota	Kappa	Lambda Lambda

任何新的行格式都需要使用 `\NewRowType` 命令定义。在定义时，可以使用可选参数。

## 第三章 附加接口

通常，tblr 环境能够接收内部和外部参数：

```
\begin{tblr} [<outer specs> ] { <inner specs> }  
  <table body>  
\end{tblr}
```

**内部参数**是 tblr 的 必选 项中的参数，它可以包含第二章中描述的新接口。

**外部参数**是 tblr 的 可选 项中的参数，多用于跨页长表格 (参见第四章)。

可以通过 \SetTblrInner 和 \SetTblrOuter 命令分别设置内部参数和外部参数。

### 3.1 内部参数

除了第二章中的新接口外，还有表 3.1 中所描述的内部参数。

表 3.1: 内部参数键与键值

键	说明与可选键值	初始值
rulesep	两条表格横线或竖线间的间距	2pt
stretch	单元格文本行距拉伸系数	1
abovesep	每行前的垂直间距	2pt
belowsep	每行后的垂直间距	2pt
rowsep	每行前后的垂直间距	2pt
leftsep	每列左边的水平间距	6pt
rightsep	每列右边的水平间距	6pt
colsep	每列左右的水平间距	6pt
hspan	单元格水平合并算法: default、even 或 minimal	default
vspan	单元格垂直合并算法: default 或 even	default
verb	是否使用 verb 命令	×
baseline	表格的基线	m

### 3.1.1 双线间距

以下示例演示了使用 `rulesep` 键替代 `\doublerulesep` 命令的方法。

```
\begin{tblr}{
  colspec={|l|l|l|l|},rowspec={|QQQ|},
  rulesep=4pt,
}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

### 3.1.2 单元格文本的最小支架

以下示例演示了使用 `stretch` 键替代 `\arraystretch` 命令的方法。

```
\begin{tblr}{hlines,stretch=1.5}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

### 3.1.3 设置所有行列间距

以下示例演示了使用 `rowsep` 键和 `colsep` 键设置行/列间距的方法。

```
\SetTblrInner{rowsep=2pt,colsep=2pt}
\begin{tblr}{hlines,vlines}
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\end{tblr}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

### 3.1.4 行/列合并算法

使用 `hspan=default` 或 `hspan=even` 时, `Tabularray` 宏包会使用合并宽度计算列宽度。但是, 使用 `hspan=minimal` 时, 则使用列宽度计算合并宽度。以下示例演示了使用 `hspan` 键不同取值的结果。

```
\SetTblrInner{hlines, vlines, hspan=default}
\begin{tblr}{cell{2}{1}={c=2}{1},cell{3}{1}={c=3}{1},cell{4}{2}={c=2}{1}}
111 111 & 222 222 & 333 333 \\
12 Multi Columns Multi Columns 12 & & 333 \\
13 Multi Columns Multi Columns Multi Columns 13 & & \\
111 & 23 Multi Columns Multi Columns 23 & \\
\end{tblr}
```

111 111	222 222	333 333
12 Multi Columns Multi Columns 12	333	
13 Multi Columns Multi Columns Multi Columns 13		
111	23 Multi Columns Multi Columns 23	

```
\SetTblrInner{hlines, vlines, hspan=even}
\begin{tblr}{cell{2}{1}={c=2}{1},cell{3}{1}={c=3}{1},cell{4}{2}={c=2}{1}}
111 111 & 222 222 & 333 333 \\
12 Multi Columns Multi Columns 12 & & 333 \\
13 Multi Columns Multi Columns Multi Columns 13 & & \\
111 & 23 Multi Columns Multi Columns 23 & \\
\end{tblr}
```

111 111	222 222	333 333
12 Multi Columns Multi Columns 12	333	
13 Multi Columns Multi Columns Multi Columns 13		
111	23 Multi Columns Multi Columns 23	

```
\SetTblrInner{hlines, vlines, hspan=minimal}
\begin{tblr}{cell{2}{1}={c=2}{1},cell{3}{1}={c=3}{1},cell{4}{2}={c=2}{1}}
111 111 & 222 222 & 333 333 \\
12 Multi Columns Multi Columns 12 & & 333 \\
13 Multi Columns Multi Columns Multi Columns 13 & & \\
111 & 23 Multi Columns Multi Columns 23 & \\
\end{tblr}
```

111 111	222 222	333 333
12 Multi Columns Multi Columns 12	333	
13 Multi Columns Multi Columns Multi Columns 13		
111	23 Multi Columns Multi Columns 23	

以下示例演示了使用 `vspan` 键不同取值的结果。

```
\SetTblrInner{hlines, vlines, vspan=default}
\begin{tblr}{column{2}={3.25cm}, cell{2}{2}={r=3}{1}}
  Column1 & Column2 \\
  Row1 & Long text that needs multiple lines.
         Long text that needs multiple lines.
         Long text that needs multiple lines. \\
  Row2 & \\
  Row3 & \\
  Row4 & Short text \\
\end{tblr}
```

Column1	Column2
Row1	Long text that
Row2	needs multiple
Row3	lines. Long text that needs multiple lines. Long text that needs multiple lines.
Row4	Short text

```
\SetTblrInner{hlines, vlines, vspan=even}
\begin{tblr}{column{2}={3.25cm}, cell{2}{2}={r=3}{1}}
  Column1 & Column2 \\
  Row1 & Long text that needs multiple lines.
         Long text that needs multiple lines.
         Long text that needs multiple lines. \\
  Row2 & \\
  Row3 & \\
  Row4 & Short text \\
\end{tblr}
```

Column1	Column2
Row1	Long text that
Row2	needs multiple
Row3	lines. Long text that needs multiple lines. Long text that needs multiple lines.
Row4	Short text



### 3.1.5 使用代码原样输出命令

使用 `verb` 键，则在单元格文本中可以使用 `\verb` 命令：

```
\begin{tblr}{hlines,verb}
 20 & 30 & \verb!\hello{world}!40 \\
 50 & \verb!\hello!60 & 70 \\
\end{tblr}
```

20	30	\hello{world}40
50	\hello60	70

### 3.1.6 设置表格基线

使用 `baseline` 键，可以设置表格的基线。`baseline` 的取值有：

t	顶端对齐
T	首行对齐
m	垂直居中对齐，初始值
b	底端对齐
B	最后一行对齐
<n>	按第 <n> 行对齐 (n 是正整数)

如果在第一行前没有 `hline`，则 `t` 或 `T` 的作用相同。但如果第一行前有 1 个或多个 `hlines`，则结果不同。

```
Baseline\begin{tblr}{hlines,baseline=t}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}Baseline
```

Baseline				Baseline
	Alpha	Beta	Gamma	
	Epsilon	Zeta	Eta	
	Iota	Kappa	Lambda	

```
Baseline\begin{tblr}{hlines,baseline=T}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}Baseline
```

Baseline	Alpha	Beta	Gamma	Baseline
	Epsilon	Zeta	Eta	
	Iota	Kappa	Lambda	

`b`和`B`的区别与`t`和`T`的区别类似。实质上，`T`和`B`是目前废弃的`\firsthline`和`\lasthline`命令的替代操作。

## 3.2 外部参数

除了第 4 章中说明的参数外，表 3.2 给出了其它外部参数。

表 3.2: 外部参数键和键值

键	说明与可选键值	初始值
baseline	设置表格基线	m
long	更改表格为跨页长表格 (longtblr)	×
tall	更改表格为浮动长表格 (talltblr)	×
expand	表格内容提前展开宏	×

### 3.2.1 设置表格基线的另一种方式

注意，可以在内部参数或外部参数中使用 `baseline` 键，这两种方式做的是同一件事，但是稍有区别，如果使用外部参数，则 `baseline=t/T/m/b/B` 中可以省略键名，仅使用键值即可。

```
Baseline\begin{tblr}[m]{hlines}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}Baseline
```

	Alpha	Beta	Gamma	
Baseline	Epsilon	Zeta	Eta	Baseline
	Iota	Kappa	Lambda	

### 3.2.2 跨页长表格 (longtblr) 和浮动长表格 (talltblr)

可以通过在外部参数中设置 `long` 将表格转换为跨页长表格 (longtblr)，或设置 `tall` 将其转换为浮动长表格 (talltblr) (参见第 四章)。因此，如下两个表格的结果是一样的：

```
\begin{longtblr}{lcr}
Alpha & Beta & Gamma
\end{longtblr}
\begin{tblr}[long]{lcr}
Alpha & Beta & Gamma
\end{tblr}
```

## 3.3 表格内容宏的提前展开

`Tabularray` 宏包使用 `l3regex` 解析表格内容，因此，需要知道表格内容中的每一个 `&` 和 `\\` 符号。所以，不能将表格内容放在任何由 `\NewTableCommand` 命令定义的表格命令中。但是，可以使用 `expand` 键为 `Tabularray` 宏包指定在解析表格内容之前 一次 性展开包含表格内容的宏。注意，该宏不可以使用 `\NewDocumentCommand` 或 `\NewExpandableDocumentCommand` 命令定义，并且 不能 有可选参数。

```

\def\tblrbody{
  \hline
  20 & 30 & 40 \\
  50 & 60 & 70 \\
  \hline
}
\begin{tblr}[expand=\tblrbody]{ccc}
  \hline
  AA & BB & CC \\
  \tblrbody
  DD & EE & FF \\
  \tblrbody
  GG & HH & II \\
  \hline
\end{tblr}

```

AA	BB	CC
20	30	40
50	60	70
DD	EE	FF
20	30	40
50	60	70
GG	HH	II

### 3.4 参数默认值

Tabularray 宏包提供了 `\SetTblrInner` 和 `\SetTblrOuter` 两个命令，这两个命令用于设置表格内部参数和外部参数默认值。

以下示例中，第一行代码用于设置此后所有表格都绘制表格横线和竖线，第二行代码用于设置此后所有表格的垂直对齐方式为底端基线对齐。

```

\SetTblrInner{hlines,vlines}
\SetTblrOuter{baseline=B}

```

### 3.5 定义新 tabularray 环境

可以使用 `\NewTblrEnviron` 命令定义新 tabularray 环境：

```

\NewTblrEnviron{mytblr}
\SetTblrInner[mytblr]{hlines,vlines}
\SetTblrOuter[mytblr]{baseline=B}
Text \begin{mytblr}{cccc}
  Alpha & Beta & Gamma & Delta \\
  Epsilon & Zeta & Eta & Theta \\
  Iota & Kappa & Lambda & Mu \\
\end{mytblr} Text

```

	Alpha	Beta	Gamma	Delta	
	Epsilon	Zeta	Eta	Theta	
Text	Iota	Kappa	Lambda	Mu	Text

如果在 `\SetTblrInner` 或 `\SetTblrOuter` 命令中未指定可选参数，则使用 `tblr` 环境的默认参数。因此，不同 Tabularray 环境可以具有不同的默认参数。

### 3.6 定义新通用环境

使用 `\NewDocumentEnvironment` 命令的 `+b` 类型参数，可以定义基于 `tblr` 环境的新的通用环境（注意在最后有一对额外的大括号）：

```
\NewDocumentEnvironment{fancytblr}{+b}{
  Before Text
  \begin{tblr}{hlines}
    #1
  \end{tblr}
  After Text
}{}

```

```
\begin{fancytblr}
  One & Two & Three \\
  Four & Five & Six \\
  Seven & Eight & Nine \\
\end{fancytblr}

```

	One	Two	Three	
Before Text	Four	Five	Six	After Text
	Seven	Eight	Nine	

### 3.7 定义新表格命令

必须使用 `\NewTableCommand` 命令定义所有用于改变表格内容样式的命令。下面的示例演示了如何定义一个新的表格命令：

```
\NewTableCommand\myhline{\hline[0.1em,red5]}
\begin{tblr}{llll}
\myhline
Alpha & Beta & Gamma & Delta \\
Epsilon & Zeta & Eta & Theta \\
Iota & Kappa & Lambda & Mu \\
\myhline
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

### 3.8 寄 (odd)/偶 (even) 选择器

从 2022A 版开始，`odd` 和 `even` 子区域选择器可以接受一个可选参数，用于指定子区域的起始和终止索引值。

```
\begin{tblr}{
  cell{odd}{1} = {red9},
  cell{odd[4]}{2} = {green9},
  cell{odd[3-X]}{3} = {blue9},
}
Head & Head & Head \\
Talk A & Place A & Date A \\
Talk B & Place B & Date B \\
Talk C & Place C & Date C \\
Talk D & Place D & Date D \\
Talk E & Place E & Date E \\
Talk F & Place F & Date F \\
Talk G & Place G & Date G \\
Talk H & Place H & Date H \\
\end{tblr}

```

Head	Head	Head
Talk A	Place A	Date A
Talk B	Place B	Date B
Talk C	Place C	Date C
Talk D	Place D	Date D
Talk E	Place E	Date E
Talk F	Place F	Date F
Talk G	Place G	Date G
Talk H	Place H	Date H

```

\begin{tblr}{
  cell{even}{1} = {yellow9},
  cell{even[4]}{2} = {cyan9},
  cell{even[3-X]}{3} = {purple9},
}
  Head & Head & Head & \\
  Talk A & Place A & Date A & \\
  Talk B & Place B & Date B & \\
  Talk C & Place C & Date C & \\
  Talk D & Place D & Date D & \\
  Talk E & Place E & Date E & \\
  Talk F & Place F & Date F & \\
  Talk G & Place G & Date G & \\
  Talk H & Place H & Date H & \\
\end{tblr}

```

Head	Head	Head
Talk A	Place A	Date A
Talk B	Place B	Date B
Talk C	Place C	Date C
Talk D	Place D	Date D
Talk E	Place E	Date E
Talk F	Place F	Date F
Talk G	Place G	Date G
Talk H	Place H	Date H

### 3.9 计数器和长度

可以在单元格中使用`rownum`、`colnum`、`rowcount`和`colcount`计数器：

```

\begin{tblr}{hlines}
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] \\
  Row=\arabic{rowcount}, Col=\arabic{colcount} &
  Row=\arabic{rowcount}, Col=\arabic{colcount} &
  Row=\arabic{rowcount}, Col=\arabic{colcount} &
  Row=\arabic{rowcount}, Col=\arabic{colcount} \\
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] &
  Cell[\arabic{rownum}][\arabic{colnum}] \\
\end{tblr}

```

Cell[1][1]	Cell[1][2]	Cell[1][3]	Cell[1][4]
Row=3, Col=4	Row=3, Col=4	Row=3, Col=4	Row=3, Col=4
Cell[3][1]	Cell[3][2]	Cell[3][3]	Cell[3][4]

也可以在单元格文本中也可以使用`\leftsep`、`\rightsep`、`\abovesep`和`\belowsep`长度。

### 3.10 跟踪 Tabularray

可使用`\SetTblrTracing`命令跟踪`tblr`的内部数据流。例如，`\SetTblrTracing{all}`用于打开所有跟踪，`\SetTblrTracing{none}`用于关闭所有跟踪。而`\SetTblrTracing{+row,+column}`仅跟踪指定的行和列的数据流。所有的跟踪结果都将写入`log`文件。

## 第四章 长表格

### 4.1 简单示例

在排版有表头和表尾的长表格时，最好将表头/表尾分开设计为题注/尾注(包括标题、表注、说明、续表文本) 和标题行/尾行(每一页都重复出现的行)。例如，通过这种办法，交替对各行使用有不同颜色，就可以实现“斑马色”表格。

表 4.1: 一个长长长长长长长长的表格

Head	Head	Head
Head	Head	Head
Alpha	Beta	Gamma
Epsilon	Zeta <sup>a</sup>	Eta
Iota	Kappa <sup>†</sup>	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Foot	Foot	Foot

续下页

表 4.1: 一个长长长长长长长长的表格 (接前页)

Head	Head	Head
Head	Head	Head
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Foot	Foot	Foot

续下页

表 4.1: 一个长长长长长长长长的表格 (接前页)

Head	Head	Head
Head	Head	Head
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Foot	Foot	Foot

续下页



表 4.1: 一个长长长长长长长长的表格 (接前页)

Head	Head	Head
Head	Head	Head
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Alpha	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda
Nu	Xi	Omicron
Rho	Sigma	Tau
Phi	Chi	Psi
Foot	Foot	Foot

<sup>a</sup> 第一个表注。

<sup>†</sup> 第二个长长长长长长的表注。

注意: 一些常规说明, 一些常规说明, 一些常规说明。

来源: 自力更生, 自力更生, 自力更生。

显然, `Tabularray`宏包中的长表格与 `threeparttablex`宏包排版的表格类似, 并且`Tabularray`支持表格尾注。注意, 表格尾注并不是页面脚注。

上述表格的 L<sup>A</sup>T<sub>E</sub>X 源代码如下，该源码具备足够的自明性。

```

\NewTblrTheme{fancy}{
  \SetTblrStyle{firsthead}{font=\bfseries}
  \SetTblrStyle{firstfoot}{fg=blue2}
  \SetTblrStyle{middlefoot}{\itshape}
  \SetTblrStyle{caption-tag}{red2}
}
\begin{longtblr}[
  theme = fancy,
  caption = {一个长长长长长长长长的表格},
  entry = {短标题},
  label = {tblr:test},
  note{a} = {第一个表注。},
  note{${\dag}$} = {第二个长长长长长长的表注。},
  remark{注意} = {一些常规说明，一些常规说明，一些常规说明。},
  remark{来源} = {自力更生，自力更生，自力更生。},
]{
  colspec = {XXX}, width = 0.85\linewidth,
  rowhead = 2, rowfoot = 1,
  row{odd} = {gray9}, row{even} = {brown9},
  row{1-2} = {purple7}, row{Z} = {blue7},
}
\hline
Head & Head & Head & \\\
\hline
Head & Head & Head & \\\
\hline
Alpha & Beta & Gamma & \\\
\hline
Epsilon & Zeta\TblrNote{a} & & Eta & \\\
\hline
Iota & Kappa\TblrNote{${\dag}$} & & Lambda & \\\
\hline
Nu & Xi & Omicron & \\\
\hline
Rho & Sigma & Tau & \\\
\hline
.....
\hline
Nu & Xi & Omicron & \\\
\hline
Rho & Sigma & Tau & \\\
\hline
Phi & Chi & Psi & \\\
\hline
Foot & Foot & Foot & \\\
\hline
\end{longtblr}

```

在Tabularray宏包中，longtblr 环境用于排版跨页长表格，并完全实现了表格样式与内容分离。

标题行和尾行是出现在每页的表格行，它们由表格中指定的行构成。需要通过 longtblr 环境的必选参数中的内部参数指定标题行和尾行。例如，上述代码中，分别通过 rowhead=2 和 rowfoot=1 指定了标题行和尾行。

表 4.2: 标题行和尾行内部参数

键	含义	初始值
rowhead	每页要显示的标题行行数，从表格的起始行开始向后计数	0
rowfoot	每页要显示的尾行行数，从表格最后一行开始向前计数	0

表格题注和尾注由标题、表注、说明和续表文本构成。需要通过 `longtblr` 环境可选参数中的外部参数指定题注和尾注。

表 4.3: 表格题注和尾注外部参数

Key Name	Key Description	Initial Value
headsep	表格题注与表格之间的垂直间距	6pt
footsep	表格尾注与表格之间的垂直间距	6pt
presep	表格题注与其之前文本之间的垂直间距	1.5\bigskipamount
postsep	表格尾注与其之后文本之间的垂直间距	1.5\bigskipamount
theme	表格主题 (包括模板及样式设置)	×
caption	表格标题	×
entry	用于目录的表格短标题	×
label	表格标签	×
note{<name>}	表格表注，其中 <name> 是表注标签	×
remark{<name>}	表格说明，其中 <name> 是说明标签	×

如果使用了 `entry=none`，则不会表格目录中添加任何条目。因此，`caption=text,entry=none` 与 `longtable` 中的 `\caption[] {text}` 的功能类似。

如使用了 `label=none`，`Tabularray` 宏包的 `table` 计数器将不自增，并将 `caption-tag` 和 `caption-sep` 模板元素置空 (见后续示例)。因此，`caption=text,entry=none,label=none` 除了计数器处理外，与 `longtable` 宏包的 `\caption* {text}` 功能类似。

## 4.2 个性化模板

### 4.2.1 模板概述

`Tabularray` 的题注和尾注模板系统的设计主要受 `beamer`、`caption` 和 `longtable` 宏包的启发。可以使用 `\DefTblrTemplate` 命令<sup>1</sup> 定义或修改一个模板，用 `\SetTblrTemplate` 命令选择默认模板。在定义模板时，可以用 `\UseTblrTemplate` 和 `\ExpTblrTemplate` 命令引入其它模板。

<sup>1</sup>从 2022A 版开始，`\DefTblrTemplate` 的另一个名称是 `\DeclareTblrTemplate`。

表 4.4: 题注和尾注模板元素

元素名称	元素说明和默认模板
contfoot-text	表格在每页尾部的续表文本, 一般是“Continued on next page”
contfoot	表格在每页尾部的续表段落, 一般包括 contfoot-text 模板
conthead-text	表格在每页标题中的续表文本, 一般是“(Continued)”
conthead	表格在每页标题中的续表段落, 一般包括 conthead-text 模板
caption-tag	标题标签, 一般类似“表 4.2”
caption-sep	标题分隔符, 一般类似“: ”
caption-text	标题文本, 一般由用户提供内容
caption	包括 caption-tag + caption-sep + caption-text 的组合
note-tag	尾注标签, 一般由用户提供
note-sep	尾注分隔符, 一般类似“ ”
note-text	尾注内容, 一般由用户提供
note	包括 note-tag + note-sep + note-text 的组合
remark-tag	注释标签, 一般由用户提供
remark-sep	注释分隔符, 一般类似“: ”
remark-text	注释文本, 一般由用户提供
remark	包括 remark-tag + remark-sep + remark-text 的组合
firsthead	第一页的表头, 一般包括 caption 模板
middlehead	中间页的表头, 一般包括 caption 和 conthead 模板
lasthead	最后一页的表头, 一般包括 caption 和 conthead 模板
head	firsthead、middlehead 和 lasthead 的所有设置
firstfoot	第一页的表尾, 一般包括 contfoot 模板
middlefoot	中间页的表尾, 一般包括 contfoot 模板
lastfoot	最后一页的表尾, 一般包括 note 和 remark 模板
foot	firstfoot、middlefoot 和 lastfoot 的所有设置

仅包含短文本的元素称为子元素。一般在子元素的名称中有一个-符号。包含一个或多个段落的元素称为主元素。通常, 在主元素的名称中不包含-符号。

除了上述模板, Tabularray 预定义了 normal 和 empty 两个模板。可以使用 \SetTblrTemplate 命令进行选择。

#### 4.2.2 续表模板

首先是续表文本的模板定义:

```

\DefTblrTemplate{contfoot-text}{normal}{Continued on next page}
\SetTblrTemplate{contfoot-text}{normal}
\DefTblrTemplate{conthead-text}{normal}{(Continued)}
\SetTblrTemplate{conthead-text}{normal}

```

在以上代码中，`\DefTblrTemplate` 命令定义了名为 `normal` 的模板，然后用 `\SetTblrTemplate` 命令将名为 `normal` 的模板设置为默认模板。`normal` 模板总是被定义的，并且会被 `Tabularray` 设置为任何一个元素的默认模板。因此，在定义一个新模板时，需要使用其它名称。

如果在 `\DefTblrTemplate` 命令中使用 `default` 作为模板名称，则会在定义时同时将其设置为默认模板。因此，上述代码也可以修改为：

```

\DefTblrTemplate{contfoot-text}{default}{Continued on next page}
\DefTblrTemplate{conthead-text}{default}{(Continued)}

```

可以通过修改这些代码以适应自己的需求。

`contfoot` 和 `conthead` 模板一般使用 `\UseTblrTemplate` 命令包含它们子元素的模板。但是，可以使用诸如水平对齐等设置参数。

```

\DefTblrTemplate{contfoot}{default}{\UseTblrTemplate{contfoot-text}{default}}
\DefTblrTemplate{conthead}{default}{\UseTblrTemplate{conthead-text}{default}}

```

### 4.2.3 标题模板

通常，标题由三部分构成，可以用如下代码定义其模板：

```

\DefTblrTemplate{caption-tag}{default}{Table\hspace{0.25em}\thetable}
\DefTblrTemplate{caption-sep}{default}{:\enskip}
\DefTblrTemplate{caption-text}{default}{\InsertTblrText{caption}}

```

`\InsertTblrText{caption}` 命令使用 `caption` 键值作为标题内容，在 `longtblr` 环境的可选参数中，可以通过 `caption` 键设置标题内容。

`caption` 模板通常用 `\UseTblrTemplate` 插入三个子模板：`caption` 模板将被用于 `firsthead` 模板。

```

\DefTblrTemplate{caption}{default}{
  \UseTblrTemplate{caption-tag}{default}
  \UseTblrTemplate{caption-sep}{default}
  \UseTblrTemplate{caption-text}{default}
}

```

当然，`capcont` 模板也包含 `conthead` 模板。`capcont` 模板也被用于 `middlehead` 和 `lasthead` 模板。

```

\DefTblrTemplate{capcont}{default}{
  \UseTblrTemplate{caption-tag}{default}
  \UseTblrTemplate{caption-sep}{default}
  \UseTblrTemplate{caption-text}{default}
  \UseTblrTemplate{conthead-text}{default}
}

```

### 4.2.4 表注和说明模板

表注模板可以按如下方式定义：

```
\DefTblrTemplate{note-tag}{default}{\textsuperscript{\InsertTblrNoteTag}}
\DefTblrTemplate{note-sep}{default}{\space}
\DefTblrTemplate{note-text}{default}{\InsertTblrNoteText}
```

```
\DefTblrTemplate{note}{default}{
  \MapTblrNotes{
    \noindent
    \UseTblrTemplate{note-tag}{default}
    \UseTblrTemplate{note-sep}{default}
    \UseTblrTemplate{note-text}{default}
  }
}
```

`\MapTblrNotes` 命令用于遍历所有表格的表注，这些表注在 `longtblr` 环境的可选参数中设置。在遍历中，可以分别通过 `\InsertTblrNoteTag` 命令 `\InsertTblrNoteText` 插入当前标签和表注文

本。

注释模板的定义与表注模板的定义类似。

```
\DefTblrTemplate{remark-tag}{default}{\InsertTblrRemarkTag}
\DefTblrTemplate{remark-sep}{default}{:\space}
\DefTblrTemplate{remark-text}{default}{\InsertTblrRemarkText}
```

```
\DefTblrTemplate{remark}{default}{
  \MapTblrRemarks{
    \noindent
    \UseTblrTemplate{remark-tag}{default}
    \UseTblrTemplate{remark-sep}{default}
    \UseTblrTemplate{remark-text}{default}
  }
}
```

### 4.2.5 表头和表尾模板

表格的表头和表尾模板被定义为包含其它模板：

```

\DefTblrTemplate{firsthead}{default}{
  \UseTblrTemplate{caption}{default}
}
\DefTblrTemplate{middlehead,lasthead}{default}{
  \UseTblrTemplate{capcont}{default}
}
\DefTblrTemplate{firstfoot,middlefoot}{default}{
  \UseTblrTemplate{contfoot}{default}
}
\DefTblrTemplate{lastfoot}{default}{
  \UseTblrTemplate{note}{default}
  \UseTblrTemplate{remark}{default}
}

```

注意，可以在 `\DefTblrTemplate` 命令中为多个元素定义同一个模板。如果仅仅需要在第一页显示 caption，则可以通过修改 `middlehead` 和 `lasthead` 的定义实现：

```

\DefTblrTemplate{middlehead,lasthead}{default}{
  \UseTblrTemplate{conthead}{default}
}

```

### 4.3 改变样式

模板元素的有效设置详见表 4.5.

表 4.5: 元素的样式

键	含义	初始值
<code>fg</code>	前景颜色	×
<code>font</code>	字体命令	×
<code>halign</code>	水平对齐方式：l (left)、c (center)、r (right) 或 j (justify)	j
<code>indent</code>	段落缩进值	Opt
<code>hang</code>	悬挂缩进值	Opt or 0.7em

注意：多数情况下，可以省略带下划线的键名而仅给出键值。`halign`、`indent` 和 `hang` 仅对主模板有效。

可以使用 `\SetTblrStyle` 命令改变元素的样式：

```

\SetTblrStyle{firsthead}{font=\bfseries}
\SetTblrStyle{firstfoot}{fg=blue2}
\SetTblrStyle{middlefoot}{\itshape}
\SetTblrStyle{caption-tag}{red2}

```

在模板定义中，当使用 `\UseTblrTemplate{element}{default}` 时，除了包含模板 `element` 代码外，会自动设置模板 `element` 的前景颜色和字体命令。相反，`\ExpTblrTemplate{element}{default}` 将仅包含模板代码。

## 4.4 定义主题

可以使用 `\NewTblrTheme` 命令定义表头与表尾的样式主题。一个主题由模板和样式设置组成，例如：

```
\NewTblrTheme{fancy}{
  \DefTblrTemplate{conthead}{default}{[[Continued]]}
  \SetTblrStyle{firsthead}{font=\bfseries}
  \SetTblrStyle{firstfoot}{fg=blue2}
  \SetTblrStyle{middlefoot}{\itshape}
  \SetTblrStyle{caption-tag}{red2}
}
```

定义了 `fancy` 主题后，在 `longtblr` 环境的可选参数中使用 `theme=fancy` 使用该主题。

## 4.5 分页控制

像 `longtable` 宏包一样，在 `longtblr` 环境中，可以使用 `\\*` 或 `\nopagebreak` 禁用分页，用 `\pagebreak` 实现强制分页。

## 4.6 可浮动长表格 (talltblr)

`Tabularray` 宏包提供了 `talltblr` 环境，以替代 `threeparttable` 环境。该环境不可跨页，但可以用于 `table` 环境中。

```
TEXT\begin{talltblr}[
  caption = {长长长长长长的表格},
  entry = {短标题},
  label = {tblr:tall},
  note{a} = {第一个表注。},
  note{${\dag$}} = {第二个长长长长长的表注。},
]{
  colspec = {XXX}, width = 0.5\linewidth, hlines,
}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta\TblrNote{a} \\
Iota & Kappa & Lambda\TblrNote{${\dag$}} \\
\end{talltblr}TEXT
```

表 4.6: 长长长长长长的表格

Alpha	Beta	Gamma
Epsilon	Zeta	Eta <sup>a</sup>
Iota	Kappa	Lambda <sup>†</sup>

<sup>a</sup> 第一个表注。

<sup>†</sup> 第二个长长长长长的表注。



## 4.7 移除长表格题注和尾注

在`Tabularray`宏包提供的`longtblr`和`talltblr`环境中,可以使用`\SetTblrTemplate{head,foot}{empty}`移除长表格的题注和尾注<sup>2</sup>。

例如,可以移除表4.6的题注。

```
\SetTblrTemplate{head,foot}{empty}
TEXT\begin{talltblr}[
  caption = {长长长长长长的表格},
  entry = {短标题},
  label = {tblr:removetallcaption},
  note{a} = {第一个表注。},
  note{${\dag}} = {第二个长长长长长长的表注。},
]{
  colspec = {XXX}, width = 0.5\linewidth, hlines,
}
Alpha & Beta & Gamma \\
Epsilon & Zeta & Eta\TblrNote{a} \\
Iota & Kappa & Lambda\TblrNote{${\dag}} \\
\end{talltblr}TEXT
```

	Alpha	Beta	Gamma	
TEXT	Epsilon	Zeta	Eta <sup>a</sup>	TEXT
	Iota	Kappa	Lambda <sup>†</sup>	

<sup>2</sup>原说明书无此说明,译者在此根据其 github 的 issues 和 discussions 添加。

## 第五章 使用扩展库

Tabularray 宏包模仿或修改了其它宏包的一些命令，为避免冲突，需要使用 `\UseTblrLibrary` 载入这些扩展库。

### 5.1 amsmath 库

如果在导言区使用了 `\UseTblrLibrary{amsmath}`，则 `tabularray` 会自动载入 `amsmath` 宏包，并定义 `+array`，`+matrix`，`+bmatrix`，`+Bmatrix`，`+pmatrix`，`+vmatrix`，`+Vmatrix` 和 `+cases` 环境。其中，每一个环境都类似于不带 `+` 前缀的环境，但是就像 `tblr` 环境一样，使用 `rowsep=2pt` 默认值。除了 `+array` 环境外，其余的每个环境都可以带一个可选项，通过该可选项，可能为环境设置内部参数。

```


$$\begin{pmatrix} \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \end{pmatrix}$$


```

$$\begin{pmatrix} \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \end{pmatrix}$$

```


$$\begin{pmatrix} \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \end{pmatrix}$$


```

$$\begin{pmatrix} \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} & \frac{2}{3} \end{pmatrix}$$

```


$$f(x) = \begin{cases} 0, & x=1; \\ \frac{1}{3}, & x=2; \\ \frac{2}{3}, & x=3; \\ 1, & x=4. \end{cases}$$


```

$$f(x) = \begin{cases} 0, & x = 1; \\ \frac{1}{3}, & x = 2; \\ \frac{2}{3}, & x = 3; \\ 1, & x = 4. \end{cases}$$

```

$f(x)=\begin{+cases}
0, & & & & x=1; \\
\frac{1}{3}, & & & & x=2; \\
\frac{2}{3}, & & & & x=3; \\
1, & & & & x=4. \\
\end{+cases}$

```

$$f(x) = \begin{cases} 0, & x = 1; \\ \frac{1}{3}, & x = 2; \\ \frac{2}{3}, & x = 3; \\ 1, & x = 4. \end{cases}$$

## 5.2 booktabs 库

在导言区使用了 `\UseTblrLibrary{booktabs}` 后, `Tabularray` 宏包则会自动载入 `booktabs` 宏包, 并且定义 `\toprule`、`\midrule`、`\bottomrule` 和 `\cmidrule` 命令, 这些命令可以直接用于 `tblr` 环境中。

```

\begin{tblr}{l1l1l1}
\toprule
Alpha & Beta & Gamma & Delta \\
\midrule
Epsilon & Zeta & Eta & Theta \\
\cmidrule{1-3}
Iota & Kappa & Lambda & Mu \\
\cmidrule{2-4}
Nu & Xi & Omicron & Pi \\
\bottomrule
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu
Nu	Xi	Omicron	Pi

类似于 `\hline` 和 `\cline` 命令, 可以通过这些命令的选项指定线宽与颜色。

```

\begin{tblr}{l1l1l1}
\toprule[2pt,purple3]
Alpha & Beta & Gamma & Delta \\
\midrule[blue3]
Epsilon & Zeta & Eta & Theta \\
\cmidrule[azure3]{2-3}
Iota & Kappa & Lambda & Mu \\
\bottomrule[2pt,purple3]
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

如果需要更多的 `cmidrule`, 则可以使用 `\cmidrulemore` 命令。

```

\begin{tblr}{l1l1l1}
\toprule
Alpha & Beta & Gamma & Delta \\
\cmidrule{1-3} \cmidrulemore{2-4}
Epsilon & Zeta & Eta & Theta \\
\cmidrule{1-3} \morecmidrules \cmidrule{2-4}
Iota & Kappa & Lambda & Mu \\
\bottomrule
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

从 2021N (2021-09-01) 版后, `\cmidrule` 命令也支持 `(l, r, lr)` 裁剪选项。

```

\begin{tblr}{llll}
\toprule
Alpha & Beta & Gamma & Delta \\
\cmidrule[lr]{1-2} \cmidrule[lr=-0.4]{3-4}
Epsilon & Zeta & Eta & Theta \\
\cmidrule[r]{1-2} \cmidrule[l]{3-4}
Iota & Kappa & Lambda & Mu \\
\bottomrule
\end{tblr}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

注意，需要将 `l`, `r` 或 `lr` 选项置于 方括号 内，并且其取值为 `-1` 和 `0` 之间的十进制数，其中，`-1` 表示裁剪整个 `colsep`，`0` 表示不进行裁剪。其默认值是 `-0.8`，即与 `booktabs` 宏包的结果类似。

同时，也提供了 `booktabs` 环境。在该环境中，将 `rowsep=0pt` 设置为默认值，但是，通过 `\toprule`、`\midrule`、`\bottomrule` 和 `\cmidrule` 命令添加了额外的垂直间距。这些垂直间距由 `\aboverulesep` 和 `\belowrulesep` 的尺寸决定。

```

\begin{booktabs}{
  colspec = lcccc,
  cell{1}{1} = {r=2}{}, cell{1}{2,4} = {c=2}{},
}
\toprule
Sample & I & & II & \\
\cmidrule[lr]{2-3} \cmidrule[lr]{4-5}
& A & B & C & D \\
\midrule
S1 & 5 & 6 & 7 & 8 \\
S2 & 6 & 7 & 8 & 5 \\
S3 & 7 & 8 & 5 & 6 \\
\bottomrule
\end{booktabs}

```

Sample	I		II	
	A	B	C	D
S1	5	6	7	8
S2	6	7	8	5
S3	7	8	5	6

也可以使用 `\specialrule` 命令指定表格横线。其第 2 个参数用于设置与前一行的 `belowsep`，其第 3 个参数用于设置当前行的 `abovesep`。

```

\begin{booktabs}{row{2}={olive9}}
\toprule
Alpha & Beta & Gamma & Delta \\
\specialrule{0.5pt}{4pt}{6pt}
Epsilon & Zeta & Eta & Theta \\
\specialrule{0.8pt,blue3}{3pt}{2pt}
Iota & Kappa & Lambda & Mu \\
\bottomrule
\end{booktabs}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

最后是 `\addlinespace` 命令，可以通过其可选参数指定需要添加的垂直距离，其默认值为 `0.5em`。该命令为前一行的 `belowsep` 添加了一半行距，并为当前行的 `abovesep` 添加另一半行距。

```

\begin{booktabs}{row{2}={olive9}}
\toprule
Alpha & Beta & Gamma & Delta \\
\addlinespace
Epsilon & Zeta & Eta & Theta \\
\addlinespace[1em]
Iota & Kappa & Lambda & Mu \\
\bottomrule
\end{booktabs}

```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

从 2022A (2022-03-01) 版开始, 可以使用 `longtabs` 环境排版 `booktabs` 长三线表, 使用 `talltabs` 环境 `booktabs` 可浮动长三线表。

### 5.3 counter 库

如需在 `tabularray` 表格中修改部分计数器, 则需使用 `\UseTblrLibrary{counter}` 载入 `counter` 库。

```

\newcounter{mycnta}
\newcommand{\mycnta}{\stepcounter{mycnta}\arabic{mycnta}}
\begin{tblr}{hlines}
\mycnta & \mycnta & \mycnta \\
\mycnta & \mycnta & \mycnta \\
\mycnta & \mycnta & \mycnta \\
\end{tblr}

```

1	2	3
4	5	6
7	8	9

### 5.4 diagbox 库

当在导言区使用了 `\UseTblrLibrary{diagbox}` 后, `Tabularray` 宏包会载入 `diagbox` 宏包, 然后, 就可以在 `tblr` 环境中使用 `\diagbox` 和 `\diagboxthree` 命令排版斜线表头。

```

\begin{tblr}{hlines,vlines}
\diagbox{Aa}{Pp} & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Pp \ Aa	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

```

\begin{tblr}{hlines,vlines}
\diagboxthree{Aa}{Pp}{Hh} & Beta & Gamma \\
Epsilon & Zeta & Eta \\
Iota & Kappa & Lambda \\
\end{tblr}

```

Pp \ Aa \ Hh	Beta	Gamma
Epsilon	Zeta	Eta
Iota	Kappa	Lambda

也可以在数学模式中使用 `\diagbox`和 `\diagboxthree` 命令。

```


\begin{tblr}{|c|cc|}
\hline
\diagbox{X_1}{X_2} & 0 & 1 \\
\hline
0 & 0.1 & 0.2 \\
1 & 0.3 & 0.4 \\
\hline
\end{tblr}


```

$X_1 \backslash X_2$	0	1
0	0.1	0.2
1	0.3	0.4

## 5.5 functional 库

在文档导言区使用 `\UseTblrLibrary{functional}` 后, `tabularray` 宏包将自动载入 `functional` 宏包, 并同时定义 `evaluate` 外部参数键和 `process` 内部参数键。这两个新定义的键用于在表格内部执行函数编程操作。

### 5.5.1 evaluate 外部参数键

使用 `evaluate` 键, 可以调用任何用 `\prgNewFunction` 定义的受保护函数, 并在分离表格内容之前将其替换为函数返回值。

`evaluate` 键的第一个应用是在表格内载入文件, 并用文件内容构造表格。假设有 `test1.tmp` 和 `test2.tmp` 两个文件, 它们的内容如下:

```


\begin{filecontents*}[overwrite]{test1.tmp}
Some & Some \\
\end{filecontents*}


```

```


\begin{filecontents*}[overwrite]{test2.tmp}
Other & Other \\
\end{filecontents*}


```

然后就可以使用 `functional` 宏包提供的 `evaluate=\fileInput` 函数载入这两个文件。

```


\begin{tblr}[evaluate=\fileInput]{hlines}
Row1 & 1 \\
\fileInput{test1.tmp}
Row3 & 3 \\
\fileInput{test2.tmp}
Row5 & 5 \\
\end{tblr}


```

Row1	1
Some	Some
Row3	3
Other	Other
Row5	5

通常, 可以定义返回表格内容的自定义函数并使用 `evaluate` 键调用该函数, 从而将返回值插入到表格中。

```

\IgnoreSpacesOn
\prgNewFunction \someFunc {m} {
  \prgReturn {#1 & #1 \\\}
}
\IgnoreSpacesOff
\begin{tblr}[evaluate=\someFunc]{hlines}
  Row1 & 1 \\\
  \someFunc{Text}
  Row3 & 3 \\\
  \someFunc{Text}
  Row5 & 5 \\\
\end{tblr}

```

Row1	1
Text	Text
Row3	3
Text	Text
Row5	5

```

\IgnoreSpacesOn
\prgNewFunction \otherFunc {} {
  \prgReturn {Other & Other \\\}
}
\IgnoreSpacesOff
\begin{tblr}[evaluate=\otherFunc]{hlines}
  Row1 & 1 \\\
  \otherFunc
  Row3 & 3 \\\
  \otherFunc
  Row5 & 5 \\\
\end{tblr}

```

Row1	1
Other	Other
Row3	3
Other	Other
Row5	5

甚至可以设计一个函数按指定的行列数产生整个空表格。

```

\IgnoreSpacesOn
\prgNewFunction \makeEmptyTable {mm} {
  \tlSet \lTmptl {\intReplicate {\intEval{#2-1}} {&}}
  \tlPutRight \lTmptl {\\\}
  \intReplicate {#1} {\tlUse \lTmptl}
}
\IgnoreSpacesOff
\begin{tblr}[evaluate=\makeEmptyTable]{hlines,vlines}
  \makeEmptyTable{3}{7}
\end{tblr}

```


### 5.5.2 process 内部参数键

使用内容参数键 `process`，可以在构建表格前修改单元格内容或样式。用 `\prgNewFunction` 定义的常见函数有：

- `\cellGetText{<rownum>}{<colnum>}`
- `\cellSetText{<rownum>}{<colnum>}{<text>}`
- `\cellSetStyle{<rownum>}{<colnum>}{<style>}`
- `\rowSetStyle{<rownum>}{<style>}`
- `\columnSetStyle{<colnum>}{<style>}`

第一个示例是按列求和计算:

```
\IgnoreSpacesOn
\prgNewFunction \funcSum {} {
  \intStepOneInline {1} {\arabic{colcount}} {
    \intZero \lTmpaInt
    \intStepOneInline {1} {\arabic{rowcount}-1} {
      \intAdd \lTmpaInt {\cellGetText {###1} {##1}}
    }
    \cellSetText {\expWhole{\arabic{rowcount}}} {##1} {\IntUse\lTmpaInt}
  }
}
\IgnoreSpacesOff
```

```
\begin{tblr}{colspec={rrr},process=\funcSum}
\hline
  1 & 2 & 3 \\
  4 & 5 & 6 \\
  7 & 8 & 9 \\
\hline
  & & \\
\hline
\end{tblr}
```

1	2	3
4	5	6
7	8	9
12	15	18

也可以基于内容设置不同单元格的背景色:

```
\IgnoreSpacesOn
\prgNewFunction \funcColor {} {
  \intStepOneInline {1} {\arabic{rowcount}} {
    \intStepOneInline {1} {\arabic{colcount}} {
      \intSet \lTmpaInt {\cellGetText {##1} {###1}}
      \intCompareTF {\lTmpaInt} > {0}
      {\cellSetStyle {##1} {###1} {bg=purple8}}
      {\cellSetStyle {##1} {###1} {bg=olive8}}
    }
  }
}
\IgnoreSpacesOff
```

```
\begin{tblr}{hlines,vlines,cells={r,$},process=\funcColor}
-1 & 2 & 3 \\
4 & 5 & -6 \\
7 & -8 & 9 \\
\end{tblr}
```

-1	2	3
4	5	-6
7	-8	9

也可以通过xcolor宏包自定义颜色,再根据表格行的行号,设置不同彩色行:



```

\definecolor{lightb}{RGB}{217,224,250}
\definecolorseries{tblrow}{rgb}{last}{lightb}{white}
\resetcolorseries[3]{tblrow}
\IgnoreSpacesOn
\prgNewFunction \funcSeries {} {
  \intStepOneInline {1} {\arabic{rowcount}} {
    \tlSet \lTmptl {\intMathMod {##1-1} {3}}
    \rowSetStyle {##1} {\expWhole{bg=tblrow!![\lTmptl]}}
  }
}
\IgnoreSpacesOff

```

```

\begin{tblr}{hlines,process=\funcSeries}
  Row1 & 1 & \\
  Row2 & 2 & \\
  Row3 & 3 & \\
  Row4 & 4 & \\
  Row5 & 5 & \\
  Row6 & 6 & \\
\end{tblr}

```

Row1	1
Row2	2
Row3	3
Row4	4
Row5	5
Row6	6

## 5.6 siunitx 库

当在导言区使用了`\UseTblrLibrary{siunitx}`后, `Tabularray` 宏包会载入 `siunitx` 宏包, 并定义了S列格式, 表示带有si键的Q列格式。

```

\begin{tblr}{
  hlines, vlines,
  colspec={S[table-format=3.2]S[table-format=3.2]}
}
  {{{Head}}} & {{{Head}}} & \\
  111 & & 111 & \\
  2.1 & & 2.2 & \\
  33.11 & & 33.22 & \\
\end{tblr}

```

Head	Head
111	111
2.1	2.2
33.11	33.22

```

\begin{tblr}{
  hlines, vlines,
  colspec={Q[si={table-format=3.2},c]Q[si={table-format=3.2},c]}
}
  {{{Head}}} & {{{Head}}} \\
  111 & 111 \\
  2.1 & 2.2 \\
  33.11 & 33.22 \\
\end{tblr}

```

Head	Head
111	111
2.1	2.2
33.11	33.22

注意，需要使用 三重 大括号对以确保单元格是非数字模式。但是用大括号将每个单元格括起来比较麻烦。因此从版本 2022B (2022-06-01) 开始，为单元格和行提供了一个新的 `guard` 键。使用 `guard` 关键较大幅度简化前面的例子。

```

\begin{tblr}{
  hlines, vlines,
  colspec={Q[si={table-format=3.2},c]Q[si={table-format=3.2},c]},
  row{1} = {guard}
}
  Head & Head \\
  111 & 111 \\
  2.1 & 2.2 \\
  33.11 & 33.22 \\
\end{tblr}

```

Head	Head
111	111
2.1	2.2
33.11	33.22

另外，也必须使用 `l`、`c` 或 `r` 设置非数字单元格的水平对齐方式。

```
\begin{tblr}{
  hlines, vlines, columns={6em},
  colspec={
    Q[si={table-format=3.2,table-number-alignment=left},l,blue7]
    Q[si={table-format=3.2,table-number-alignment=center},c,teal7]
    Q[si={table-format=3.2,table-number-alignment=right},r,purple7]
  },
  row{1} = {guard}
}
  Head & Head & Head & \\
  111 & & 111 & \\
  2.1 & & 2.2 & 2.3 \\
  33.11 & & 33.22 & 33.33 \\
\end{tblr}
```

Head	Head	Head	\\
111	111	111	\\
2.1	2.2	2.3	\\
33.11	33.22	33.33	\\

此时，`S` 和 `s` 列格式都可用。实质上，这两个列格式是按如下方式定义的：

```
\NewColumnType{S}[1] [] {Q[si={#1},c]}
\NewColumnType{s}[1] [] {Q[si={#1},c,cmd=\TblrUnit]}
```

## 5.7 varwidth 库

为了构建更好的表格，`tabularray` 需要度量单元格的宽度。默认情况下，它使用 `\hbox` 实现测量。但当单元格中包含有诸如列表或行间公式等垂直结构的元素时，则可能会产生错误。

通过在导言区使用 `\UseTblrLibrary{varwidth}`，`tabularray` 宏包会载入 `varwidth` 宏包，并会为表格添加 `measure` 内部参数。当设置了 `measure=vbox` 后，则会使用 `\vbox` 测量单元格宽度。

```
\begin{tblr}{hlines,measure=vbox}
  Text Text Text Text Text Text Text
  \begin{itemize}
    \item List List List List List List
    \item List List List List List List List
  \end{itemize}
  Text Text Text Text Text Text Text \\
\end{tblr}
```

Text Text Text Text Text Text Text
• List List List List List List
• List List List List List List List
Text Text Text Text Text Text Text

从 2022A (2022-03-01) 版开始，可以使用 `stretch=-1` 移除列表环境上下的间距，下面的例子需要 `enumitem` 宏包，并使用其 `nosep` 选项：

• List List List List List	oooo
• List List List List List List	
• List List List List List	
• List List List List List List	gggg

```
\begin{tblr}{
  hlines,vlines,rowspec={Q[l,t]Q[l,b]},
  measure=vbox,stretch=-1,
}
  \begin{itemize}[nosep]
    \item List List List List List
    \item List List List List List List
  \end{itemize} & oooo \\
  \begin{itemize}[nosep]
    \item List List List List List
    \item List List List List List List
  \end{itemize} & gggg \\
\end{tblr}
```

注意，`stretch=-1` 选项也会移除单元格中的支架，因此，对于使用了`rowsep=0pt`的`tabulararray`环境，其效果不会很好。如`booktabs`库中的`booktabs/longtabs/talltabs` 环境。

# 第六章 历史与未来

## 6.1 未来

从 2022 年开始，除了严重错误的热修复外，所有新版本都将仅在三月、六月、九月或十二月发布。可以通过关注里程碑页面，了解即将发布版本的预定日期和它们的变化：

<https://github.com/lvjr/tabularray/milestones>

为了使即将发布的版本更加稳定，非常欢迎测试仓库中最新的宏包文件。测试时，只需要下载下面的 `tabularray.sty` 并把它放到  $\text{T}_{\text{E}}\text{X}$  文档的工作路径中：

<https://github.com/lvjr/tabularray/raw/main/tabularray.sty>

## 6.2 历史

`tabularray` 的更新日志将发布于维基页面：

<https://github.com/lvjr/tabularray/wiki/ChangeLog>

在 2022A 中，主要的更新有：

- 移除了 `\multicolumn` 命令；使用更为方便的 `\SetCell` 命令。
- 移除了 `\multirow` 命令；使用更为方便的 `\SetCell` 命令。
- 移除了 `\firsthline` 命令；使用更为方便的 `baseline=T` 选项。
- 移除了 `\lasthline` 命令；使用更为方便的 `baseline=B` 选项。

对于旧的文档，可以使用 `\usepackage{tabularray}[=v2021]` 回滚到 2021 版。