

The ‘fancyvrb’ package

Fancy Verbatims in L^AT_EX

Timothy Van Zandt
Princeton University
Princeton – USA

Packaging and documentation by

Denis Girou (CNRS/IDRIS – France),
Sebastian Rahtz† (Elsevier – GB)
and
Herbert Voß (FU Berlin – DE)
翻译: virhuiai@qq.com

2022 年 6 月 22 日

摘要

`fancyvrb` 宏包提供了高度可配置的抄录环境。用户可以对抄录环境做的有：修改字体、修改字号、为抄录内容加上行号、使用带边框的代码示例环境、高亮抄录的文本、以及有条件的处理抄录文本等等。

Report bugs to hvoss@tug.org

Thanks to Claudio Beccari, Jean-François Burnol, Philippe Esperet, Michael Friendly, Mario Hassler, Ohad Kammar, Mikhail Kolodin, Andreas Matthias, Frank Mittelbach, Rolf Niepraschk, Will Robertson, Ulrich M. Schwarz, Thomas Siegel, Clemens Steinke, and Vladimir Volovich.

目录

第一部分	Package fancyvrb	4
1	介绍	4
2	脚注中的 Verbatim material	4
3	改进的抄录命令	5
4	Verbatim 环境	5
4.1	抄录环境的设置	5
4.1.1	注释	6
4.1.2	忽略起始的字符数量	6
4.1.3	指定修饰格式	7
4.1.4	修改每行的格式化方式	7
4.1.5	字体	8
4.1.6	边框的类型和 characteristics	9
4.1.7	Label for the 环境	11
4.1.8	行号	13
4.1.9	选择要排版的起始行和终止的行	15
4.1.10	空格和 tab characters	17
4.1.11	行间距	17
4.1.12	用于插入命令的转义字符	17
4.1.13	边距	18
4.1.14	溢出盒子信息	19
4.1.15	跨页	19
4.1.16	Catcode characters	19
4.1.17	Active characters	20
4.1.18	引用标签	20

4.2	不同类型的抄录环境	20
4.2.1	Verbatim 环境	20
4.2.2	BVerbatim 环境	20
4.2.3	LVerbatim 环境	21
4.2.4	自定义抄录环境	21
5	保存和取用抄录文本和环境	22
6	从文件读写抄录	26
7	自动漂亮排版	27
8	已知的问题	28
9	小结	28
第二部分 fancyvrb-ex 包		29
10	Example 环境	29
11	CenterExample 环境	30
12	SideBySideExample 环境	30

第一部分 Package fancyvrb

1 介绍

‘fancyvrb’ is the development of the *verbatim* macros of the ‘fancybox’ package, Section 11 of [1]. 同标准的 L^AT_EX verbatim 环境相比, ‘fancyvrb’ 提供了 6 种拓展功能,:

1. 抄录命令可以在脚注中使用,
2. 一些抄录命令得到了加强,
3. 宏包自带了一些抄录环境, 配有许多参数来控制内容的输出格式; 它也让你能够更方便的定义新的符合自身需求的抄录环境,
4. 提供了保存和取用抄录文本和环境方法,
5. 在抄录模式提供了命令以便读写文件, 这是很有用的。
6. 你可以 build *example* 环境 (同时展示出排版的结果和所使用的排版的代码), with the same versatility as normal verbatim.

该宏包对抄录环境或从文件中读取文本是, 每次只扫描一行。这样的方式允许它 to pre-process each line, rejecting it, 移除空格, numbering it, etc, before going on to execute the body of the line with the appropriate catcodes set.

2 脚注中的 Verbatim material

调用 \VerbatimFootnotes 宏声明之后 (在 the preamble 之后使用), 就可以将 verbatim 命令和环境 (the L^AT_EX or ‘fancyvrb’ ones) 放到脚注中了, 而标准的 L^AT_EX 中是不允许这样做的:

1	\VerbatimFootnotes
2	我们可以将抄录\footnote{\verb+_Yes!_+} 文本放到脚注中了.

我们可以将抄录¹ 文本放到脚注中了.

¹_Yes!_

3 改进的抄录命令

`\DefineShortVerb` 命令让我们可以指定一个符号作为抄录命令的简化方式, 这个符号包围住要抄录的文本。命令 `\UndefineShortVerb` 用来取消这个效果 (在 \LaTeX 的 ‘shortverb’ 宏包中也提供了这两个命令):

We can simply write _verbatim_ material using a single _delimiter_. And we can _change_ the character.	<pre> 1 \DefineShortVerb{\ } 2 We can simply write \Verb+_verbatim_+ 3 material using a single _delimiter_ 4 \UndefineShortVerb{\ } 5 \DefineShortVerb{+} 6 And we can +_change_+ the character. </pre>
---	---

更强大的 (使排版效果多多样化) 是, 我们可以指定在抄录文本中的 *escape* (转义) 字符 (using the `\Verb` macro or with a ‘shortverb’ character defined), 以执行类似格式化这样内容, 请使用在 4.1.12 中介绍的, 抄录环境的 `commandchars` 参数。

4 Verbatim 环境

提供了不少的抄录环境, 都有一堆的参数可以用来配置。下面的例子我使用了 `Verbatim` 环境, 它相当于标准 \LaTeX 的 `verbatim` 环境。配置参数可以使用 `\fvset` 命令全局设置, 也可以作为可选参数在环境的开始处指定^{2,3}。

<pre> 1 First verbatim line. 2 Second verbatim line. </pre>	<pre> 1 \begin{Verbatim} 2 First verbatim line. 3 Second verbatim line. 4 \end{Verbatim} </pre>
---	---

4.1 抄录环境的设置

抄录环境的输出效果, 有不少是可以定制的; 下面列出我们可以设置的参数。

²For clarification in this paper, note that we generally indent each verbatim line with two spaces.

³This mechanism uses the ‘keyval’ package from the standard \LaTeX graphics distribution, written by David CARLISLE.

4.1.1 注释

commentchar (字符) : 定义抄录环境中的注释字符, 若某行以此字符开头, 在输出中你整行都不会看到 (默认: *empty*)。

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 A comment 2 Verbatim line. ! 一 </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim}[commentchar=!] 2 A comment 3 Verbatim line. ! 一 4 ! 二、这行注释在输出中你是看不到的 5 \end{Verbatim} </div>
---	--

要注意一种特殊的情况, 当注释符不是第一个非空白字符时: 这是因为我们定义的这个注释符是被等同于 \TeX 的注释符了, 这意味着换行符也会被忽略. 因此, 这种情况下, 当前行和下一行会合为一行了, 另外, 如果最后一行包含注释符, 最后一行的内容也不会输出, 因为 ‘fancyvrb’ 宏包的每一行的输出是在遇到结束符才执行的, 而最后一行包含注释符这个情况下永远不会遇到结束符...

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 第一行. 第二行. </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim}[commentchar=\%] 2 第一行. % First line 3 第二行. 4 第三行. % Third line lost... 5 \end{Verbatim} </div>
<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 第一行. 2 第二行. </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim}[commentchar=\%] 2 第一行. 3 第二行. 4 最后一行包含注释符. % 不输出 5 \end{Verbatim} </div>

4.1.2 忽略起始的字符数量

gobble (integer) : 每行开始跳过的字符数 (最大值为 9), 这在包含有空格缩进的内容时很有用 (默认: *empty*—没有字符被忽略)。

1	Verbatim line.	1	<code>\begin{Verbatim}</code>
		2	Verbatim line.
		3	<code>\end{Verbatim}</code>
1	Verbatim line.	4	
		5	<code>\begin{Verbatim}[gobble=2]</code>
		6	Verbatim line.
		7	<code>\end{Verbatim}</code>
1	atim line.	8	
		9	<code>\begin{Verbatim}[gobble=8]</code>
		10	Verbatim line.
		11	<code>\end{Verbatim}</code>

4.1.3 指定修饰格式

`formatcom` (命令) : 指定输出抄录 文本之前运行的命令。(默认: *empty*).

1	First verbatim line.	1	<code>\begin{Verbatim}[formatcom=\color{red}]</code>
2	Second verbatim line.	2	First verbatim line.
		3	Second verbatim line.
		4	<code>\end{Verbatim}</code>

4.1.4 修改每行的格式化方式

`\FancyVerbFormatLine` 命令定义了每行的格式化方式。默认值是 `\def\FancyVerbFormatLine#1{#1}`, 我们也可以按我们的需要, 重新定义它 (抄录环境中, 行号的计数器名是 `FancyVerbLine`):

译注: 原样输出

1	\Rightarrow First verbatim line.	1	<code>\renewcommand{\FancyVerbFormatLine}[1]{%</code>
2	\Rightarrow Second verbatim line.	2	<code>\makebox[0cm][l]{\${\rightarrow}\$#1}</code>
3	\Rightarrow Third verbatim line.	3	<code>\begin{Verbatim}</code>
		4	First verbatim line.
		5	Second verbatim line.
		6	Third verbatim line.
		7	<code>\end{Verbatim}</code>

4.1 抄录环境的设置

```
1 FIRST VERBATIM LINE.  
2 Second verbatim line.  
3 THIRD VERBATIM LINE.
```

```
1 \renewcommand{\FancyVerbFormatLine}[1]{%  
2 \ifodd\value{FancyVerbLine}%  
3 \MakeUppercase{#1}\else#1\fi}  
4 \begin{Verbatim}  
5 First verbatim line.  
6 Second verbatim line.  
7 Third verbatim line.  
8 \end{Verbatim}
```

4.1.5 字体

`fontfamily (family name)` : 使用的字体。预定义的有: `tt`、`courier` 和 `helvetica` (默认: `tt`).

`\ifPostScriptFonts`

我们可以猜测, `PostScript` 字体是可用的

```
1 Verbatim line.
```

```
1 \begin{Verbatim}[fontfamily=helvetica]  
2 Verbatim line.  
3 \end{Verbatim}
```

`fontsize (font size)` : 要使用的字体的大小 (默认: `auto`—同当前字号一致). 如果你也使用了 '`relsize`' 宏包, 您可以将之改成与当前字体成比例的大小 (例如: `fontsize=\relsize{-2}`).

我们可以猜测, `PostScript` 字体是可用的

```
1 Verbatim line.
```

```
1 Verbatim line.
```

```
1 \begin{Verbatim}[fontsize=\small]  
2 Verbatim line.  
3 \end{Verbatim}  
4  
5 \begin{Verbatim}[fontfamily=courier,  
6 fontsize=\large]  
7 Verbatim line.  
8 \end{Verbatim}
```

`fontshape (字形)` : 使用的字形 (默认: `auto`—同当前字形一致).

我们可以猜测，PostScript 字体是可用的

1	Verbatim line.	1	<pre>\begin{Verbatim}[fontfamily=courier, 2 fontshape=it] 3 Verbatim line. 4 \end{Verbatim}</pre>
---	----------------	---	---

fontseries (series name) : 使用的 \LaTeX 字体 ‘series’ (默认: *auto*—同当前字体一致).

我们可以猜测，PostScript 字体是可用的

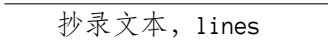
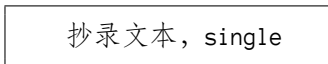
1	Verbatim line.	1	<pre>\begin{Verbatim}[fontfamily=courier, 2 fontseries=b] 3 Verbatim line. 4 \end{Verbatim}</pre>
---	----------------	---	---

4.1.6 边框的类型和 characteristics

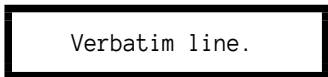
frame (none|leftline|topline|bottomline|lines|single) : 抄录文本的四周的边框类型 (默认: *none*—无边框). 使用 *leftline* 和 *single* 模式时, \LaTeX 的 `\fboxsep` 命令指定的宽度的空白会被添加到左边的垂直线和文本之间。

1	抄录文本, leftline	1	<pre>\begin{Verbatim}[frame=leftline] 2 抄录文本, leftline 3 \end{Verbatim}</pre>
1	抄录文本, topline	1	<pre>\begin{Verbatim}[frame=topline] 2 抄录文本, topline 3 \end{Verbatim}</pre>
1	抄录文本, bottomline	1	<pre>\begin{Verbatim}[frame=bottomline] 2 抄录文本, bottomline 3 \end{Verbatim}</pre>

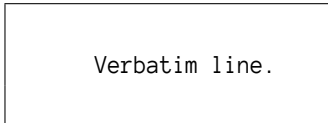
4.1 抄录环境的设置

1		1 <code>\begin{Verbatim}[frame=lines]</code> 2 抄录文本, lines 3 <code>\end{Verbatim}</code>
1		1 <code>\begin{Verbatim}[frame=single]</code> 2 抄录文本, single 3 <code>\end{Verbatim}</code>

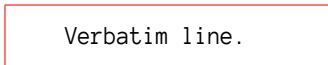
`framerule (dimension)` : 设置边框线的宽度 (默认: $0.4pt$, 如果有指定边框).

1		1 <code>\begin{Verbatim}[frame=single,</code> 2 <code>framerule=1mm]</code> 3 Verbatim line. 4 <code>\end{Verbatim}</code>
---	---	---

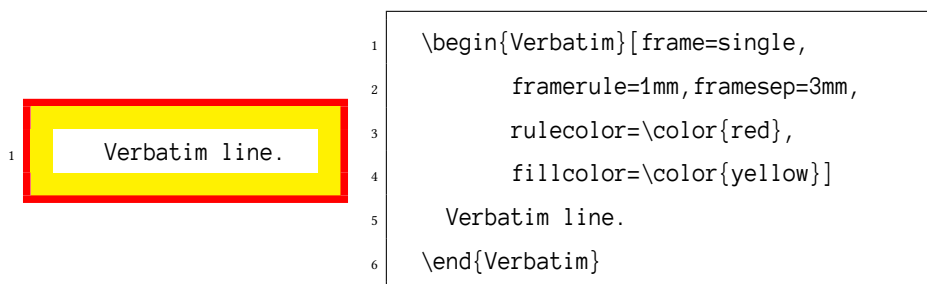
`framesep (dimension)` : 设置抄录文本与边框的距离 (默认: `\fboxsep`).

1		1 <code>\begin{Verbatim}[frame=single,</code> 2 <code>framesep=5mm]</code> 3 Verbatim line. 4 <code>\end{Verbatim}</code>
---	---	--

`rulecolor (color command)` : 设置边框的颜色, 标准的 \LaTeX 描述方式 (默认: *black*).

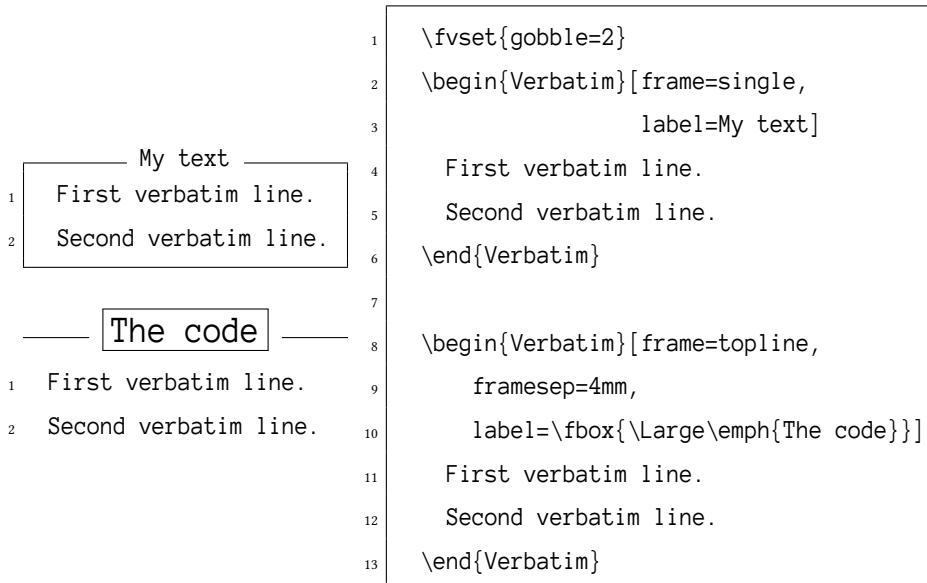
1		1 <code>\begin{Verbatim}[frame=single,</code> 2 <code>rulecolor=\color{red}]</code> 3 Verbatim line. 4 <code>\end{Verbatim}</code>
---	---	---

`fillcolor (color command)` : 设置抄录文本与边线之间空白的颜色 (宽度为前面介绍的 `framesep` 的值) (默认: *none* — no color).



4.1.7 Label for the 环境

`label` (`{[string]string}`) : 指定顶部的标签、底部的标签, 或同时指定顶部和底部的标签, 用于描述抄录的内容 (默认: *empty* – no label)。如果标签文字包含特殊字符, 比如逗号、等号, 那就必须放到组里。如果只指定了一个标签字符串, 各种情况下使用的标签内容是一致的。但如果可选的标签也给出了 (2 个标签字符串), 可选的标签文字会用到顶部, 第二个用到底部。要注意的是, 如果 `frame` 参数使用了 `topline`、`bottomline`、`lines` 或 `single` 以外的其他值, 则不会打印标签。



`labelposition` (`none|topline|bottomline|all`) : 设置标签的位置, 如果设置了标签文字的话, *which must be coherent with the kind of frame chosen* (默认: *none if the label is empty, topline if one label is defined and all if two are de-*

*fin*ed). 当然, 如果配置成冲突的了 (比如 `frame=topline,labelposition=bottomline`) 将不显示标签.

```

1  frame=single,
2  label=Text,
3  labelposition=all
Text

```

```

1  \fvset{gobble=2}
2  \begin{Verbatim}[frame=single,
3      framesep=2mm,
4      label=Text,labelposition=all]
5      frame=single,
6      label=Text,
7      labelposition=all
8  \end{Verbatim}

```

```

Text
1  First verbatim line.
2  Second verbatim line.

```

```

1  \fvset{gobble=2}
2  \begin{Verbatim}[frame=lines,
3      label=Text,labelposition=topline]
4      First verbatim line.
5      Second verbatim line.
6  \end{Verbatim}

```

```

1  First verbatim line.
2  Second verbatim line.
Code included

```

```

1  \begin{Verbatim}[frame=bottomline,
2      framesep=3mm,
3      label=\textit{Code included},
4      labelposition=bottomline]
5      First verbatim line.
6      Second verbatim line.
7  \end{Verbatim}

```

```

Beginning of code
1  First verbatim line.
2  Second verbatim line.
End of code

```

```

1  \begin{Verbatim}[frame=lines,
2      framesep=3mm,
3      label={[Beginning of code]End of code}]
4      First verbatim line.
5      Second verbatim line.
6  \end{Verbatim}

```

4.1.8 行号

`numbers (none|left|right)` : 抄录的文本行的行号 (默认: *none* — no numbering).

如果启用, 行号放在抄录环境的边框之外。

1 左:numbers=left	1 \begin{Verbatim}[gobble=2,numbers=left] 2 左:numbers=left 3 \end{Verbatim}
右:numbers=right	1 \begin{Verbatim}[gobble=2, 2 numbers=right,numbersep=0pt] 3 右:numbers=right 4 \end{Verbatim}

`numbersep (dimension)` : 行号和抄录文本行间的距离 (默认: *12pt*).

1 First verbatim line.	1 \begin{Verbatim}[gobble=2, 2 numbers=left,numbersep=2pt] 3 First verbatim line. 4 Second verbatim line. 5 \end{Verbatim}
2 Second verbatim line.	

`firstnumber (auto|last| 数字)` : 指定行号的起始数字 (默认: *auto* — 从 1 开始).

`last` 指定行号续着上一个抄录环境的行号。如果指定的是一个数字, 行号就从这个数字开始增加。

1 Verbatim line.	1 \fvset{gobble=2, 2 numbers=left,numbersep=3pt} 3 \begin{Verbatim} 4 Verbatim line. 5 \end{Verbatim}
2 Verbatim line.	6 \begin{Verbatim}[firstnumber=last] 7 Verbatim line. 8 \end{Verbatim}

4.1 抄录环境的设置

100	Verbatim line.	1 \fvset{gobble=2, 2 numbers=left,numbersep=3pt} 3 \begin{Verbatim}[firstnumber=100] 4 Verbatim line. 5 \end{Verbatim}
-----	----------------	---

stepnumber (数字) : 每多少行显示行号 (默认: 1— 在设置了行号的情况下, 所有行的行号都会显示出来)。

2	First verbatim line. Second verbatim line. Third verbatim line.	1 \begin{Verbatim}[gobble=2,numbers=left, 2 numbersep=3pt,stepnumber=2] 3 First verbatim line. 4 Second verbatim line. 5 Third verbatim line. 6 \end{Verbatim}
---	---	---

`\theFancyVerbLine` 命令定义了行号的排版格式, 行号使用的计数器的名称是 `FancyVerbLine`:

8.a	First verbatim line.	1 \renewcommand{\theFancyVerbLine}{%
8.b	Second verbatim line.	2 \textcolor{red}{\small
8.c	Third verbatim line.	3 8.\alph{FancyVerbLine}}}
		4 \begin{Verbatim}[gobble=2,
		5 numbers=left,numbersep=2pt]
		6 First verbatim line.
		7 Second verbatim line.
		8 Third verbatim line.
		9 \end{Verbatim}

numberblanklines (boolean) : 是否对空行编号 (全空的行, 或只包含空白字符的行) (默认: *true*— 所有行都会被编号)。

1	First verbatim line.	1	<code>\begin{Verbatim}[gobble=2,numbers=left,</code>
		2	<code>numbersep=3pt,</code>
		3	<code>numberblanklines=false]</code>
		4	First verbatim line.
2	Second verbatim line.	5	
		6	
		7	Second verbatim line.
		8	<code>\end{Verbatim}</code>

4.1.9 选择要排版的起始行和终止的行

`firstline (integer)` : 要排版的起始行号 (默认: *empty* — 从第一行开始).

`lastline (integer)` : 要排版的终止行号 (默认: *empty* — 一直到最后一行).

1	First verbatim line.	1	<code>\begin{Verbatim}[gobble=2,</code>
		2	<code>firstline,lastline,</code>
		3	<code>numbers=left,numbersep=2pt]</code>
2	Second verbatim line.	4	First verbatim line.
3	Third verbatim line.	5	Second verbatim line.
		6	Third verbatim line.
		7	<code>\end{Verbatim}</code>

2	Second verbatim line.	1	<code>\begin{Verbatim}[gobble=2,firstline=2,</code>
3	Third verbatim line.	2	<code>numbers=left,numbersep=2pt]</code>
		3	First verbatim line.
		4	Second verbatim line.
		5	Third verbatim line.
		6	<code>\end{Verbatim}</code>

4.1 抄录环境的设置

1	First verbatim line.
2	
3	
4	
5	
6	

```
1 \begin{Verbatim}[gobble=2,lastline=1,
2     numbers=left,numbersep=2pt]
3     First verbatim line.
4     Second verbatim line.
5     Third verbatim line.
6 \end{Verbatim}
```

除了指定 `firstline` 参数来控制开始的行号范围, 你也可以指定一个开始字符串; 开始的第一行就是第一个完全匹配这个字符串的那一行. (当指定了忽略的起始字符数时, 是会有区别的) 结束字符串也是类似的. 你可以混合使用指定行号和指定字符串这两种方式, e.g. 用 `firstline` 指定开始的行号, 用结束字符串 来指定结束的位置. 指定字符串有点笨拙. 首先你需要用以下的 `\newcommand*` 命令定义相应的字符串:

3	Second verbatim line.
4	
5	
6	
7	
8	
9	

```
1 \newcommand*\FancyVerbStartString{FROM}
2 \newcommand*\FancyVerbStopString{TO}
3 \begin{Verbatim}
4     First verbatim line.
5 FROM
6     Second verbatim line.
7 TO
8     Third verbatim line.
9 \end{Verbatim}
```

在重新定义开始字符串、结束字符串时, 你要使用 `\renewcommand*` 命令重新定义。

若设置了 `lastline=` 或 `lastline=0` 所有行都不会输出。

foo	
bar	

```
1 foo
2 \begin{Verbatim}[frame=none,lastline=]
3     First verbatim line.
4     Second verbatim line.
5     Third verbatim line.
6 \end{Verbatim}
7 bar
```


4.1.10 空格和 tab characters

`showspaces` (boolean) : 是否将每个空格用可视的字符展示出来。(默认: *false*).

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \verb\Verbatim\line. </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim}[showspaces=true] 2 Verbatim line. 3 \end{Verbatim} </div>
---	--

In practice, 所有的 `verbatim` 环境有相应的带 `*` 变体, 设置了 `showspaces=true`:

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \verb\Verbatim\line. </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim*} 2 Verbatim line. 3 \end{Verbatim*} </div>
---	---

还有一些参数用来配置制表符的输出方式 (使用制表符实际上是相当过时的编程风格了):

`showtabs` (boolean) : 显式显示制表符 (默认: *false* — 不显示制表符).

`obeytabs` (boolean) : 根据制表符定位字符 (默认: *false* — tab characters are added to the current position).

`tabsize` (integer) : 一个制表符对应的空格数 (默认: 8).

4.1.11 行间距

`baselinestretch` (auto|dimension) : 设置抄录环境中, \LaTeX 的 ‘`baselinestretch`’ 参数的值。(默认: *auto* — 它在抄录命令之前的当前值).

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 First verbatim line. 2 Second verbatim line. </div>	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> 1 \begin{Verbatim}[baselinestretch=2] 2 First verbatim line. 3 Second verbatim line. 4 \end{Verbatim} </div>
---	--

4.1.12 用于插入命令的转义字符

`commandchars` (指定三个字符) : 指定开始命令的字符、命令参数的组的开始和结束符号; 我们介绍下在抄录代码中使用 转义序列. 当然, 最好是选择抄录文本中未出现的字符来使用! (默认: *empty*). 译注: 通常三个符号为`\\\{\}`

4.1 抄录环境的设置

1	This is a comment	1	<code>\begin{Verbatim}[commandchars=\\\{\}]</code>
2	First verbatim line.	2	<code>\textit{This is a comment}</code>
3	<code>Second</code> verbatim line.	3	First verbatim line.
4	Third verbatim line.	4	<code>\fbox{Second}</code> verbatim line.
		5	<code>\textcolor{red}{Third}</code> verbatim line.
		6	<code>\end{Verbatim}</code>
		7	
1	<code>\textbf{Verbatim}</code> line	8	<code>\begin{Verbatim}[commandchars=+\{\}]</code>
		9	<code>+textit[\textbf{Verbatim}] line.</code>
		10	<code>\end{Verbatim}</code>

这样，指定了 `commandchars` 后，就可以放置标签了，`\ref` 引用的时候，显示的是在抄录环境中的行号：

1	First verbatim line.	1	<code>\begin{Verbatim}[commandchars=\\\{\},</code>
2	Second line.	2	<code>numbers=left,numbersep=2pt]</code>
3	Third verbatim line.	3	First verbatim line.
		4	Second line. <code>\label{vrb:Important}</code>
		5	Third verbatim line.
		6	<code>\end{Verbatim}</code>
		7	
		8	As I previously shown
		9	line~ <code>\ref{vrb:Important}</code> , it is...

As I previously shown line 2,
it is...

4.1.13 边距

`xleftmargin` (长度) : 每行的开头添加的缩进 (默认: `0pt`).

1	Verbatim line.	1	<code>\begin{Verbatim}[frame=single,</code>
		2	<code>xleftmargin=5mm]</code>
		3	Verbatim line.
		4	<code>\end{Verbatim}</code>

`xrightmargin` (dimension) : 在每行之后添加的右边距 (默认: `0pt -`).

1	Verbatim line.	1	<code>\begin{Verbatim}[frame=single,</code>
		2	<code> xrightmargin=1cm]</code>
		3	<code>Verbatim line.</code>
		4	<code>\end{Verbatim}</code>

`resetmargins (boolean)` : 重置 left margin, 当在其他的带有缩进的环境中时, 这很有用 (默认: *false* — no reset of the margin).

		1	<code>\begin{itemize}</code>
		2	<code>\item First item</code>
• First item		3	<code>\begin{Verbatim}[frame=single]</code>
	Verbatim line.	4	<code>Verbatim line.</code>
		5	<code>\end{Verbatim}</code>
• Second item		6	<code>\item Second item</code>
		7	<code>\begin{Verbatim}[frame=single,</code>
	Verbatim line.	8	<code> resetmargins=true]</code>
		9	<code>Verbatim line.</code>
		10	<code>\end{Verbatim}</code>
		11	<code>\end{itemize}</code>

4.1.14 溢出盒子信息

`hfuzz (dimension)` : value to give to the \TeX `\hfuzz` dimension for text to format. 这可以避免显示一些不重要的 *Overfull box* 信息 (默认: *2pt*).

4.1.15 跨页

`samepage (boolean)` : 在非常特殊的情况下, 我们希望, 抄录环境中的内容不要分到不同的页面, *even if it does not fit on the current page*. 要避免跨页, 可以将 `samepage` 设置为 *true* (默认: *false*).

4.1.16 Catcode characters

`codes (macro)` : 指定 *catcode* 变更 (默认: *empty*).

4.2 不同类型的抄录环境

例如，下面这样设置后，我们就可以在抄录文本中包含数学公式了：

1	$x=1/\sqrt{z^2} ! \frac{1}{\sqrt{z^2}}$	1	<code>\begin{Verbatim}[commandchars=\\\{\},</code>
		2	<code>codes={\catcode`\$=3\catcode`^=7}}</code>
		3	<code>x=1/sqrt(z**2) ! \$\frac{1}{\sqrt{z^2}}\$</code>
		4	<code>\end{Verbatim}</code>

4.1.17 Active characters

`defineactive (macro)` : to define the effect of *active* characters (默认: *empty*).

This allows us to do some devious tricks: see the example in Section 6 on page 26.

4.1.18 引用标签

`relabel (< 标签 >)` : 设置标签，以便 `\pageref` 引用命令显示页数。

1	First verbatim line.	1	<code>\begin{Verbatim}[relabel=verb0]</code>
2	Second verbatim line.	2	First verbatim line.
		3	Second verbatim line.
		4	<code>\end{Verbatim}</code>
	See the verbatim on page 20.	5	See the verbatim on
		6	page~\pageref{verb0}.

4.2 不同类型的抄录环境

4.2.1 Verbatim 环境

这是我们上面已经用过的 ‘normal’ 抄录环境.

4.2.2 BVerbatim 环境

这个环境将抄录内容放到一个 $\text{T}_{\text{E}}\text{X}$ 盒子中。有些前面介绍过的参数在这个环境是不存在的 (尤其是和边框相关的参数), 不过，也有两个新的参数可用于配置:

`boxwidth (auto|dimension)` : 使用的盒子的宽度 (默认: *auto* — the width of the longest line is used).

`baseline (b|c|t)` : 对齐用的基线位置 (on the baseline, the center or the top of the box) (默认: *b*).

First
Second First
 Second

```

1 \fvset{gobble=2}
2 \begin{BVerbatim}
3   First
4   Second
5 \end{BVerbatim}
6 \begin{BVerbatim}[baseline=c]
7   First
8   Second
9 \end{BVerbatim}

```

First
Second First
 Second

```

1 \begin{BVerbatim}[boxwidth=2cm]
2   First
3   Second
4 \end{BVerbatim}
5 \begin{BVerbatim}[boxwidth=2cm,
6                   baseline=t]
7   First
8   Second
9 \end{BVerbatim}

```

4.2.3 LVerbatim 环境

这个环境将抄录内容放到 \LaTeX 左右模式 (实际上这在 \TeX 里也叫 受限水平模式).

4.2.4 自定义抄录环境

要定义自己的抄录环境是很容易的。

你可以用 `\RecustomVerbatimEnvironment` 命令重新定义已经存在的环境, 或使用 `\DefineVerbatimEnvironment` 命令⁴ 创建新的环境。这两种方式, 你需要指定新环境的名称、它所基于的环境的名字, 以及一组选项的初始值。这些选项的值可以在使用时传入的可选参数覆盖:

1	First verbatim line.	1	<code>\RecustomVerbatimEnvironment</code>
2	Second verbatim line.	2	<code>{Verbatim}{Verbatim}</code>
		3	<code>{gobble=2, frame=single}</code>
		4	<code>\begin{Verbatim}</code>
		5	First verbatim line.
		6	Second verbatim line.
		7	<code>\end{Verbatim}</code>
1	First verbatim line.	1	<code>\DefineVerbatimEnvironment%</code>
2	Second verbatim line.	2	<code>{MyVerbatim}{Verbatim}</code>
		3	<code>{gobble=2, numbers=left, numbersep=2mm,</code>
		4	<code>frame=lines, framerule=0.8mm}</code>
		5	<code>\begin{MyVerbatim}</code>
		6	First verbatim line.
		7	Second verbatim line.
		8	<code>\end{MyVerbatim}</code>
		9	
		10	<code>\begin{MyVerbatim}[numbers=none,</code>
		11	<code>framerule=1pt]</code>
		12	First verbatim line.
		13	Second verbatim line.
		14	<code>\end{MyVerbatim}</code>

5 保存和取用抄录文本和环境

`\SaveVerb` 和 `\UseVerb` 命令让我们可以保存和取用抄录内容。`\UseVerb` 是 robust 的:

⁴也存在抄录命令 `\CustomVerbatimCommand` 和 `\RecustomVerbatimCommand` 命令; 例如:
`\RecustomVerbatimCommand{\VerbatimInput}{VerbatimInput}{frame=lines}`

I have saved `_verbatim_` and
reuse it later as many times
as I want

Using `_verbatim_`
`_verbatim_`.

```
1 \DefineShortVerb{\|}
2 \SaveVerb{Verb}|_verbatim_|
3 I have saved \UseVerb{Verb} and reuse
4 it later as many times as I want
5 \subsection*{Using \UseVerb{Verb}}
6 \UseVerb{Verb}.
```

这也为，向那些通常不允许放置抄录命令的 \LaTeX 命令中放置抄录文本提供了一种方案：

```
1 \DefineShortVerb{\|}\SaveVerb{Verb}|_OK^| \marginpar{\UseVerb{Verb}}
```

`_OK^`

有个很有用的情况是：在描述列表中，使用抄录文本作为条目文本 (\LaTeX 里，这通常是不允许放置抄录命令的)，使用 `aftersave` 参数：

`aftersave (macro)` : 用来动态保存一些抄录文本的命令 (默认: *empty*).

`\MyCommand : my command`

```
1 \newcommand{\Vitem}{%
2   \SaveVerb[aftersave={%
3     \item[\UseVerb{Vitem}]]{Vitem}}
4 \DefineShortVerb{\|}
5 \begin{description}
6   \Vitem|\MyCommand|: my command
7 \end{description}
```

译者注：

- 1 保存 `\MyCommand` 到 `Vitem`
- 2 继续执行 `aftersave` 指定的 `\item[\UseVerb{Vitem}]`，以及后续的：`my command`

同样的，我们可以保存和取用 (盒模式使用 `\UseVerbatim`, `\BUseVerbatim`, `\LUseVerbatim` 使用左右模式) 整个的抄录环境：

Verbatim line.	1 \fvset{gobble=0,numbers=none}
	2 \begin{SaveVerbatim}{抄录保存名}
and	3 Verbatim line.
Verbatim line.	4 \end{SaveVerbatim}
	5 \UseVerbatim{抄录保存名}
	6 and \UseVerbatim{抄录保存名}

first first	1 \fvset{gobble=0,numbers=none}
econd and econd.	2 \begin{SaveVerbatim}[gobble=5]%
	3 {VerbEnv}
first	4 1234First
econd	5 Second
	6 \end{SaveVerbatim}
and	7
first	8 \fbox{\BUseVerbatim{VerbEnv}}
econd	9 and \BUseVerbatim{VerbEnv}.
	10
	11 \LUseVerbatim{VerbEnv} and
	12 \LUseVerbatim{VerbEnv}

1 This works.
2 我直接使用
3 Verbatim 环境.

1 然而, 如果我使用
2 SaveVerbatim
3 定义了一个可复用的
4 Verbatim,
5 开启了行号。

```
1 \begin{SaveVerbatim}{F00}
2 然而, 如果我使用
3 SaveVerbatim
4 定义了一个可复用的
5 Verbatim,
6 开启了行号。
7 \end{SaveVerbatim}
8
9 \begin{Verbatim}[numbers=left]
10 This works.
11 我直接使用
12 Verbatim 环境.
13 \end{Verbatim}
14
15 \UseVerbatim[numbers=left]{F00}
```

6 从文件读写抄录

`\VerbatimInput` 命令 (变种 `\BVerbatimInput`、`\LVerbatimInput` 也) 让我们可以从文件中读取内容并以抄录处理输出。当然, 我们前面介绍过的那些配置参数也还可以使用:

<pre>1 A "hello" program 2 3 ogram hello 4 print *, "Hello world" 5 d program hello</pre>	<pre>1 A "hello" program 2 3 ogram hello 4 print *, "Hello world" 5 d program hello</pre>	<pre>3 ogram hello 4 print *, "Hello world" 5 d program hello</pre>	<pre>1 \RecustomVerbatimCommand{\VerbatimInput} 2 {VerbatimInput}{gobble=4} 3 \fvset{fontsize=\small} 4 \VerbatimInput{hello.f90} 5 6 \fvset{frame=single,numbers=left, 7 numbersep=3pt} 8 \VerbatimInput{hello.f90} 9 10 \VerbatimInput[firstline=3, 11 rulecolor=\color{green}] 12 {hello.f90} 13 14 \VerbatimInput[frame=lines, 15 fontshape=sl,fontsize=\footnotesize] 16 {hello.f90}</pre>
---	---	---	---

我们可以用 `'defineactive'` 参数将程序文本中的注释行设置为不同的样式:

```

1 A "hello" program
2
3 ogram hello
4 print *, "Hello world"
5 d program hello

```

```

1 \RecustomVerbatimCommand{\VerbatimInput}
2 {VerbatimInput}{gobble=4}
3 \def\ExclamationPoint{\char33}
4 \catcode`\!=\active
5 \VerbatimInput%
6 [defineactive=%
7 \def!\{\color{cyan}\itshape
8 \ExclamationPoint}]
9 {hello.f90}

```

要注意的是，如果文件内容在当前面放不下了，它将根据需要，自动分解跨页 (除非 `samepage` 参数被设置为 `true` 了)。

类似的，`VerbatimOut` 环境则用来将抄录内容写入到外部的文件中, in the same way:

```

1 I write that.
2 And that too.

```

```

1 \begin{VerbatimOut}{file.txt}
2 I write that.
3 And that too.
4 \end{VerbatimOut}
5
6 \VerbatimInput[frame=single,
7 numbers=left,numbersep=6pt]{file.txt}

```

7 自动高亮排版

显然, 自动 高亮排版不在此包的范围内。尽管如此, this is specially interesting for verbatim inclusion of programming 代码文件或片段. 在 \LaTeX 世界 (not speaking of the *literate programming* way), there are software for some special languages, as the ‘C++2LaTeX’ package from Norbert KIESEL, but mainly two generic ones, which use completely different modes (an external preprocessor written in C and a \TeX based solution): the ‘LGrind’ [3] system, currently maintained by Michael PIEFEL, and the ‘listings’ [4] package from Carsten HEINZ.

‘fancyvrb’ 和 ‘listings’ 包未来版本有计划进行合作, which will offer great advantages to both users of the two actual packages, and will 允许 ‘fancyvrb’ 的

用户拥有自动漂亮排版编程语言代码。

8 已知的问题

- Vladimir VOLOVICH <vvv@vvv.vsu.ru> 有上报提到 the special character `\th`, 在 T1 编码是可用的, 但在 ‘fancyvrb’ 宏包的作为抄录内容时却不可用。It can be true for other special characters too.

9 小结

There are a few other possibilities that 我们在这没介绍到. 特别注意, 我们可以定义一些自定义的内容到 (fancyvrb.cfg) 文件, 宏包会在最后时加载这个文件 (如果有), 保存你自定义的命令和环境以及重定义一些已有的属性。

第二部分 fancyvrb-ex 包

这个宏包定义了一些 `example` 环境，可以将源码和排版效果并排或上下输出。它们在这个 `fancyvrb` 文档中也被运用到。

10 Example 环境

```

1 \begin{Example}
2   First verbatim line.
3   Second verbatim line.
4   Third verbatim line.
5 \end{Example}

```

```

1   First verbatim line.
2   Second verbatim line.
3   Third verbatim line.

```

First verbatim line. Second verbatim line. Third verbatim line.

```

1 \begin{Example}[frame=lines,framerule=1mm,
2   numbers=left]
3   First verbatim line.
4   Second verbatim line.
5   Third verbatim line.
6 \end{Example}

```

```

1 First verbatim line.
2 Second verbatim line.
3 Third verbatim line.

```

First verbatim line. Second verbatim line. Third verbatim line.

11 CenterExample 环境

```
1 \begin{CenterExample}[frame=single,  
2     numbers=right]  
3     First verbatim line.  
4     Second verbatim line.  
5     居中示例  
6 \end{CenterExample}
```

First verbatim line.	1
Second verbatim line.	2
居中示例	3

First verbatim line. Second verbatim line. 居中示例

12 SideBySideExample 环境

```
1 \begin{SideBySideExample}[  
2     xrightmargin=5cm,  
3     frame=lines, numbers=left]  
4     First verbatim line.  
5     Second verbatim line.  
6     Third verbatim line.  
7     并排示例  
8 \end{SideBySideExample}
```

First verbatim line. Second
verbatim line. 并排示例

First verbatim line.	1
Second verbatim line.	2
并排示例	3

参考文献

- [1] Timothy VAN ZANDT, *Documentation for ‘fancybox’: Box tips and tricks for \LaTeX* . Available from [CTAN:macros/latex/contrib/supported/fancybox](#), 1993.
- [2] Timothy VAN ZANDT, *‘fancyvrb’: Fancy Verbatims in \LaTeX* . Available from [CTAN:macros/latex/contrib/supported/fancyvrb](#), 1998.
- [3] Various authors (current maintainer: Michael PIEFEL), *The ‘LGrind’ package*. Available from [CTAN:support/lgrind](#), 1998.
- [4] Carsten HEINZ, *The ‘Listings’ package*. Available from [CTAN:macros/latex/contrib/supported/listings](#), 1996-1997.