

# The ‘fancyvrb’ package

## Fancy Verbatims in $\LaTeX$

Timothy Van Zandt  
Princeton University  
Princeton – USA  
tvz@Princeton.EDU

Packaging and documentation by

Denis Girou (CNRS/IDRIS – France),  
Sebastian Rahtz (Elsevier – GB)  
and  
Herbert Voß (FU Berlin – DE)

Version 2.8  
May 15, 2010

Documentation revised May 15, 2010

### Abstract

This package provides very sophisticated facilities for reading and writing verbatim  $\TeX$  code. Users can perform common tasks like changing font family and size, numbering lines, framing code examples, colouring text and conditionally processing text.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Verbatim material in footnotes</b>	<b>3</b>
<b>3</b>	<b>Improved verbatim commands</b>	<b>4</b>
<b>4</b>	<b>Verbatim environments</b>	<b>4</b>
4.1	Customization of verbatim environments . . . . .	4
4.1.1	Comments . . . . .	4
4.1.2	Initial characters to suppress . . . . .	5
4.1.3	Customization of formatting . . . . .	5
4.1.4	Changing individual line formatting . . . . .	6
4.1.5	Fonts . . . . .	6
4.1.6	Types and characteristics of frames . . . . .	7

4.1.7	Label for the environment . . . . .	9
4.1.8	Line numbering . . . . .	10
4.1.9	Selection of lines to print . . . . .	12
4.1.10	Spaces and tab characters . . . . .	13
4.1.11	Space between lines . . . . .	14
4.1.12	Escape characters for inserting commands . . . . .	14
4.1.13	Margins . . . . .	15
4.1.14	Overfull box messages . . . . .	16
4.1.15	Page breaks . . . . .	16
4.1.16	Catcode characters . . . . .	16
4.1.17	Active characters . . . . .	16
4.2	Different kinds of verbatim environments . . . . .	17
4.2.1	Verbatim environment . . . . .	17
4.2.2	BVerbatim environment . . . . .	17
4.2.3	LVerbatim environment . . . . .	17
4.2.4	Personalized environments . . . . .	18
<b>5</b>	<b>Saving and restoring verbatim text and environments</b>	<b>18</b>
<b>6</b>	<b>Writing and reading verbatim files</b>	<b>20</b>
<b>7</b>	<b>Automatic pretty printing</b>	<b>21</b>
<b>8</b>	<b>Known problems</b>	<b>21</b>
<b>9</b>	<b>Thanks</b>	<b>22</b>
<b>10</b>	<b>Conclusion</b>	<b>22</b>
<b>11</b>	<b>Driver file</b>	<b>22</b>
<b>12</b>	<b>'fancyvrb' code</b>	<b>23</b>
12.1	Preamble . . . . .	23
12.2	Errors . . . . .	23
12.3	Verbatim footnotes . . . . .	23
12.4	key=value . . . . .	24
12.5	Scanning . . . . .	27
12.6	Codes . . . . .	28
12.7	Preprocess line . . . . .	29
12.8	Scanning environments . . . . .	34
12.9	Check end . . . . .	34
12.10	Line-by-line scanning . . . . .	36
12.11	Input . . . . .	38
12.12	Formatting – Common components . . . . .	39
12.13	List environments . . . . .	44
12.14	BVerbatim . . . . .	60
12.15	Save verbatim . . . . .	62

12.16	Verbatim out . . . . .	63
12.17	Short verbatim . . . . .	64

## 1 Introduction

‘fancyvrb’ is the development of the *verbatim* macros of the ‘fancybox’ package, Section 11 of [1]. It offers six kinds of extended functionality, compared to the standard  $\TeX$  verbatim environment:

1. verbatim commands can be used in footnotes,
2. several verbatim commands are enhanced,
3. a variety of verbatim environments are provided, with many parameters to change the way the contents are printed; it is also possible to define new customized verbatim environments,
4. a way is provided to save and restore verbatim text and environments,
5. there are macros to write and read files in verbatim mode, with the usual versatility,
6. you can build *example* environments (showing both result and verbatim text), with the same versatility as normal verbatim.

The package works by scanning a line at a time from an environment or a file. This allows it to pre-process each line, rejecting it, removing spaces, numbering it, etc, before going on to execute the body of the line with the appropriate catcodes set.

## 2 Verbatim material in footnotes

After a `\VerbatimFootnotes` macro declaration (to use after the preamble), it is possible to put verbatim commands and environments (the  $\TeX$  or ‘fancyvrb’ ones) in footnotes, unlike in standard  $\TeX$ :

```

1 \VerbatimFootnotes
2 We can put verbatim\footnote{\verb+_Yes!+_} text in footnotes.

```

We can put verbatim<sup>1</sup> text in footnotes.

---

<sup>1</sup>\_Yes!\_

### 3 Improved verbatim commands

The `\DefineShortVerb` macro allows us to define a special character as an abbreviation to enclose verbatim text and the `\UndefineShortVerb` macro suppresses the special meaning of the specified character (the same functionalities are provided in the  $\TeX$  'shortvrb' package):

We can simply write <code>_verbatim_</code> material using a single <code>_delimiter_</code> . And we can <code>_change_</code> 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>
--	--

To make matters more versatile, we can nominate *escape* characters in verbatim text (using the `\Verb` macro or with a 'shortvrb' character defined), to perform formatting or similar tasks, using the `commandchars` parameter as shown for environments in paragraph 4.1.12.

### 4 Verbatim environments

Several verbatim environments are available, each with a lot of parameters to customize them. In the following examples we use the `Verbatim` environment, which is the equivalent of the standard `verbatim`. The parameters can be set globally using the `\fvset` macro or in an optional argument after the start of the environment<sup>2,3</sup>.

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

#### 4.1 Customization of verbatim environments

The appearance of verbatim environments can be changed in many and varied ways; here we list the keys that can be set.

##### 4.1.1 Comments

`commentchar` (character) : character to define comments in the verbatim code, so that lines starting with this character will not be printed (*Default: empty*).

---

<sup>2</sup>For clarification in this paper, note that we generally indent each verbatim line with two spaces.  
<sup>3</sup>This mechanism uses the 'keyval' package from the standard  $\TeX$  graphics distribution, written by David CARLISLE.

```
% A comment
Verbatim line.
```

```
1 \begin{Verbatim}[commentchar=!]
2   % A comment
3   Verbatim line.
4   ! A comment that you will not see
5 \end{Verbatim}
```

Take care to a special effect if the comment character is not the first non blank one: it is because this character is in fact managed as the  $\TeX$  comment one, that is to say that it gobble the newline character too. So, in this case, the current line will be joined with the next one and, more, the last one will be lost if it contain a comment, as ‘fancyvrb’ print a line only after finding it end character, which will never occurred in this case...

```
First line.   Second.
```

```
1 \begin{Verbatim}[commentchar=\%]
2   First line. % First line
3   Second.
4   Third line. % Third line lost...
5 \end{Verbatim}
```

#### 4.1.2 Initial characters to suppress

`gobble` (integer) : number of characters to suppress at the beginning of each line (up to a maximum of 9), mainly useful when environments are indented (*Default: empty* — no character suppressed).

```
Verbatim line.
```

```
Verbatim line.
```

```
im line.
```

```
1 \begin{Verbatim}
2   Verbatim line.
3 \end{Verbatim}
4
5 \begin{Verbatim}[gobble=2]
6   Verbatim line.
7 \end{Verbatim}
8
9 \begin{Verbatim}[gobble=8]
10  Verbatim line.
11 \end{Verbatim}
```

#### 4.1.3 Customization of formatting

`formatcom` (command) : command to execute before printing verbatim text (*Default: empty*).

```

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

```

First verbatim line.  
Second verbatim line.

#### 4.1.4 Changing individual line formatting

The macro `\FancyVerbFormatLine` defines the way each line is formatted. Its default value is `\def\FancyVerbFormatLine#1{#1}`, but we can redefine it at our convenience (`FancyVerbLine` is the name of the line counter):

```

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

```

⇒ First verbatim line.  
⇒ Second verbatim line.  
⇒ 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}

```

FIRST VERBATIM LINE.  
Second verbatim line.  
THIRD VERBATIM LINE.

#### 4.1.5 Fonts

`fontfamily (family name)` : font family to use. `tt`, `courier` and `helvetica` are pre-defined (*Default: tt*).

```

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

```

Verbatim line.

`fontsize (font size)` : size of the font to use (*Default: auto* — the same as the current font). If you use the `resize` package too, you can require a change of the size proportional to the current one (for instance: `fontsize=\resize{-2}`).

Verbatim line.

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 (font shape) : font shape to use (*Default: auto* — the same as the current font).

*Verbatim line.*

```
1 \begin{Verbatim}[fontfamily=courier,
2                   fontshape=it]
3   Verbatim line.
4 \end{Verbatim}
```

fontseries (series name) :  $\TeX$  font ‘series’ to use (*Default: auto* — the same as the current font).

**Verbatim line.**

```
1 \begin{Verbatim}[fontfamily=courier,
2                   fontseries=b]
3   Verbatim line.
4 \end{Verbatim}
```

#### 4.1.6 Types and characteristics of frames

frame (none|leftline|topline|bottomline|lines|single) : type of frame around the verbatim environment (*Default: none* — no frame). With leftline and single modes, a space of a length given by the  $\TeX$  `\fboxsep` macro is added between the left vertical line and the text.

| Verbatim line.

---

Verbatim line.

---

Verbatim line.

---

---

Verbatim line.

Verbatim line.

```
1 \begin{Verbatim}[frame=leftline]
2   Verbatim line.
3 \end{Verbatim}
4
5 \begin{Verbatim}[frame=topline]
6   Verbatim line.
7 \end{Verbatim}
8
9 \begin{Verbatim}[frame=bottomline]
10  Verbatim line.
11 \end{Verbatim}
12
13 \begin{Verbatim}[frame=lines]
14   Verbatim line.
15 \end{Verbatim}
16
17 \begin{Verbatim}[frame=single]
18   Verbatim line.
19 \end{Verbatim}
```

framerule (dimension) : width of the rule of the frame (*Default: 0.4pt if framing specified*).

Verbatim line.

```
1 \begin{Verbatim}[frame=single,
2                   framerule=1mm]
3   Verbatim line.
4 \end{Verbatim}
```

framesep (dimension) : width of the gap between the frame and the text (*Default: \fboxsep*).

Verbatim line.

```
1 \begin{Verbatim}[frame=single,
2                   framesep=5mm]
3   Verbatim line.
4 \end{Verbatim}
```

rulecolor (color command) : color of the frame rule, expressed in the standard  $\TeX$  way (*Default: black*).

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

`fillcolor` (color command) : color used to fill the space between the frame and the text (its thickness is given by `framesep`) (*Default: none* — no color).

Verbatim line.	<pre> 1 \begin{Verbatim}[frame=single, 2   framerule=1mm,framesep=3mm, 3   rulecolor=\color{red}, 4   fillcolor=\color{yellow}] 5   Verbatim line. 6 \end{Verbatim} </pre>
----------------	--

#### 4.1.7 Label for the environment

`label` (`{[string]string}`) : label(s) to print on top, bottom or both frame lines of the environment to describe its content (*Default: empty* — no label). If the label(s) contains special characters, as a comma or an equal sign, it must be put inside a group. If only one string is given, it will be used for both top and bottom lines (if the two are printed), but if an optional first label is given too, this one will be used for the top line and the second one for the bottom line. Note also that, if another value than `topline`, `bottomline`, `lines` or `single` is used for the `frame` parameter, the label(s) will not be printed.

My text First verbatim line. Second verbatim line.	<pre> 1 \fvset{gobble=2} 2 \begin{Verbatim}[frame=single, 3   label=My text] 4   First verbatim line. 5   Second verbatim line. 6 \end{Verbatim} </pre>
The code First verbatim line. Second verbatim line.	<pre> 7 8 \begin{Verbatim}[frame=topline, 9   framesep=4mm, 10  label=\fbox{\Large\emph{The code}}] 11  First verbatim line. 12  Second verbatim line. 13 \end{Verbatim} </pre>

labelposition (none|topline|bottomline|all) : position where to print the label if one is defined, which must be coherent with the kind of frame chosen (*Default: none if the label is empty, topline if one label is defined and all if two are defined*). Of course, some incompatible options (like frame=topline,labelposition=bottomline) will not print the label.

```

_____ Text _____
| First verbatim line. |
| Second verbatim line. |
|_____ Text _____|

```

```

_____ Text _____
| First verbatim line. |
| Second verbatim line. |
|_____ Text _____|

```

```

1 \fvset{gobble=2}
2 \begin{Verbatim}[frame=single,
3     framesep=2mm,
4     label=Text,labelposition=all]
5     First verbatim line.
6     Second verbatim line.
7 \end{Verbatim}
8
9 \begin{Verbatim}[frame=lines,
10     label=Text,labelposition=topline]
11     First verbatim line.
12     Second verbatim line.
13 \end{Verbatim}

```

```

First verbatim line.
Second verbatim line.
— Code included —
— Beginning of code —
First verbatim line.
Second verbatim line.
— End of code —

```

```

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}
8
9 \begin{Verbatim}[frame=lines,
10     framesep=3mm,
11     label={[[Beginning of code]End of code}]
12     First verbatim line.
13     Second verbatim line.
14 \end{Verbatim}

```

#### 4.1.8 Line numbering

numbers (none|left|right) : numbering of the verbatim lines (*Default: none* — no numbering). If requested, this numbering is done *outside* the verbatim environment.

<pre> 1 First verbatim line. 2 Second verbatim line.  First verbatim line. Second verbatim line. </pre>	<pre> 1 \begin{Verbatim}[gobble=2,numbers=left] 2   First verbatim line. 3   Second verbatim line. 4 \end{Verbatim}  5 6 \begin{Verbatim}[gobble=2, 7   numbers=right,numbersep=0pt] 8   First verbatim line. 9   Second verbatim line. 10 \end{Verbatim} </pre>
---	--

numbersep (dimension) : gap between numbers and verbatim lines (*Default: 12pt*).

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

firstnumber (auto|last|integer) : number of the first line (*Default: auto*— numbering starts from 1). last means that the numbering is continued from the previous verbatim environment. If an integer is given, its value will be used to start the numbering.

<pre> 1 Verbatim line.  2 Verbatim line.  100 Verbatim line. </pre>	<pre> 1 \fvset{gobble=2, 2   numbers=left,numbersep=3pt} 3 \begin{Verbatim} 4   Verbatim line. 5 \end{Verbatim}  6 7 \begin{Verbatim}[firstnumber=last] 8   Verbatim line. 9 \end{Verbatim}  10 11 \begin{Verbatim}[firstnumber=100] 12   Verbatim line. 13 \end{Verbatim} </pre>
---	---

stepnumber (integer) : interval at which line numbers are printed (*Default: 1*— all lines are numbered).

First verbatim line. 2 Second verbatim line. Third verbatim line.	<pre> 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} </pre>
---	---

The macro `\theFancyVerbLine` defines the typesetting style of the numbering, and the counter used is `FancyVerbLine`:

8.a First verbatim line. 8.b Second verbatim line. 8.c Third verbatim line.	<pre> 1 \renewcommand{\theFancyVerbLine}{% 2     \textcolor{red}{\small 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} </pre>
---	---

`numberblanklines` (boolean) : to number or not the empty lines (really empty or containing blank characters only) (*Default: true*—all lines are numbered).

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

#### 4.1.9 Selection of lines to print

`firstline` (integer) : first line to print (*Default: empty*—all lines from the first are printed).

```

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

```

lastline (integer) : last line to print (*Default: empty* — all lines until the last one are printed).

```

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

```

Instead of specifying a firstline at which to start printing a range of lines, you can define a start string; the start of the range is the first line that exactly equals the string. (The comparison is made before any characters are gobbled off the front of the line.) Similarly for a stop string. You can mix line-numbers and strings, e.g. start at firstline, and end at a stop string. Specifying the strings is a bit klunky. Initially you must define the strings with `\newcommand*` as in:      Second verbatim line.

```

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

```

To redefine the strings, you must use `\renewcommand*`.

#### 4.1.10 Spaces and tab characters

showspaces (boolean) : print a special character representing each space (*Default: false* — spaces not shown).

```

1 \begin{Verbatim}[showspaces=true]
2 Verbatim line.
3 \end{Verbatim}

```

In practice, all verbatim environments have a `*` variant, which sets `showspaces=true`:

```

\begin{Verbatim*}
  Verbatim line.
\end{Verbatim*}

```

There are also some parameters to determine the way tab characters are interpreted (using `tabs` is in fact a rather old-fashioned style of coding):

`showtabs` (boolean) : explicitly show tab characters (*Default: false* — tab characters not shown).

`obeytabs` (boolean) : position characters according to the tabs (*Default: false* — tab characters are added to the current position).

`tabsize` (integer) : number of spaces given by a tab character (*Default: 8*).

#### 4.1.11 Space between lines

`baselinestretch` (auto|dimension) : value to give to the usual ‘`baselinestretch`’  $\TeX$  parameter (*Default: auto*—its current value just before the verbatim command).

```

\begin{Verbatim}[baselinestretch=2]
  First verbatim line.
  Second verbatim line.
\end{Verbatim}

```

#### 4.1.12 Escape characters for inserting commands

`commandchars` (three characters) : characters which define the character which starts a macro and marks the beginning and end of a group; thus lets us introduce *escape* sequences in verbatim code. Of course, it is better to choose special characters which are not used in the verbatim text! (*Default: empty*).

```

\begin{Verbatim}[commandchars=\\\{\}]
  \textit{% This is a comment}
  First verbatim line.
  \fbox{Second} verbatim line.
  \textcolor{red}{Third} verbatim line.
\end{Verbatim}

\textbf{Verbatim} line.
\begin{Verbatim}[commandchars=+\[\]]
  +textit[\textbf{Verbatim} line].
\end{Verbatim}

```

Using this way, it is also possible to put labels to be able, later, to make reference to some lines of the verbatim environments:

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

#### 4.1.13 Margins

`xleftmargin` (dimension) : indentation to add at the start of each line (*Default: Opt* — no left margin).

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

`xrightmargin` (dimension) : right margin to add after each line (*Default: Opt* — no right margin).

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

`resetmargins` (boolean) : reset the left margin, which is useful if we are inside other indented environments (*Default: false* — no reset of the margin).

- First item

Verbatim line.

- Second item

Verbatim line.

```

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

```

**4.1.14 Overfull box messages**

hfuzz (dimension) : value to give to the  $\TeX$  `\hfuzz` dimension for text to format. This can be used to avoid seeing some unimportant *Overfull box* messages (Default: 2pt).

**4.1.15 Page breaks**

samepage (boolean) : in very special circumstances, we may want to make sure that a verbatim environment is not broken, even if it does not fit on the current page. To avoid a page break, we can set the samepage parameter to *true* (Default: *false*).

**4.1.16 Catcode characters**

codes (macro) : to specify *catcode* changes (Default: empty).

For instance, this allows us to include formatted mathematics in verbatim text:

$x=1/\sqrt{z^2}$  !  $\frac{1}{\sqrt{z^2}}$

```

1 \begin{Verbatim}[commandchars=\\\{\},
2   codes={\catcode'\$=3\catcode'\^=7}]
3   x=1/\sqrt{z**2} ! $\frac{1}{\sqrt{z^2}}$
4 \end{Verbatim}

```

**4.1.17 Active characters**

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

This allows us to do some devious tricks: see the example in Section 6 on page 20.

## 4.2 Different kinds of verbatim environments

### 4.2.1 Verbatim environment

This is the ‘normal’ verbatim environment which we have been using up to now.

### 4.2.2 BVerbatim environment

This environment puts the verbatim material in a  $\TeX$  box. Some parameters do not work inside this environment (notably the framing ones), but two new ones are available:

`boxwidth (auto|dimension)` : size of the box used (*Default: auto* — the width of the longest line is used).

`baseline (b|c|t)` : position of the baseline (on the baseline, the center or the top of the box) (*Default: 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 environment

This environment puts verbatim material into  $\TeX$  ‘LR’ mode (the so-called *left-to-right* mode, which in fact is the same thing that  $\TeX$  itself calls *restricted horizontal mode*).

#### 4.2.4 Personalized environments

It is easy to define personal customized environments. You can redefine the existing ones using the `\RecustomVerbatimEnvironment` macro or create your own ones, using the `\DefineVerbatimEnvironment` macro<sup>4</sup>. In each case, you specify the name of the new environment, the type of environment on which it is based, and a set of initial option values. The options can be overridden with an optional argument in the normal way:

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

## 5 Saving and restoring verbatim text and environments

The `\SaveVerb` and `\UseVerb` macros allow us to save and restore verbatim material.

<sup>4</sup>For verbatim commands, the `\CustomVerbatimCommand` and `\RecustomVerbatimCommand` macros also exist; for instance:  
`\RecustomVerbatimCommand{\VerbatimInput}{VerbatimInput}{frame=lines}`

I have saved `_verbatim_` and reuse it later as many times as I want `_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 \UseVerb{Verb}.

```

This also provides a solution to putting verbatim text inside  $\TeX$  commands which do not normally permit it:

```

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

```

`_OK^`

There is a useful ability to use verbatim text as the item text in a description list (something not normally permitted in  $\TeX$ ), using the `aftersave` parameter:

`aftersave` (macro) : macro to dynamically save some verbatim material (*Default: empty*).

```

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

```

In the same way, we can use and restore (in normal, boxed and LR mode, using `\UseVerbatim`, `\BUseVerbatim` and `\LUseVerbatim` respectively) entire verbatim environments:

```

1 \begin{SaveVerbatim}{VerbEnv}
2   Verbatim line.
3 \end{SaveVerbatim}
4 \UseVerbatim{VerbEnv}
5 and \UseVerbatim{VerbEnv}

```

Verbatim line.  
and  
Verbatim line.

```

st      st
ond and ond.

      st
      ond

and

      st
      ond

```

```

1 \begin{SaveVerbatim}[gobble=5]{VerbEnv}
2   First
3   Second
4 \end{SaveVerbatim}
5
6 \fbox{\BUseVerbatim{VerbEnv}}
7 and \BUseVerbatim{VerbEnv}.
8
9 \LUseVerbatim{VerbEnv} and
10 \LUseVerbatim{VerbEnv}

```

## 6 Writing and reading verbatim files

The command `\VerbatimInput` (the variants `\BVerbatimInput` and `\LVerbatimInput` also exist) allows inclusion of the contents of a file with verbatim formatting. Of course, the various parameters which we have described for customizing can still be used:

```
! A "hello" program
```

```

program hello
  print *,"Hello world"
end program hello

```

```

1 ! A "hello" program
2
3 program hello
4   print *,"Hello world"
5 end program hello

```

```

3 program hello
4   print *,"Hello world"
5 end program hello

```

---

```

1 ! A "hello" program
2
3 program hello
4   print *,"Hello world"
5 end program hello

```

---

```

1 \fvset{fontsize=\small}
2 \VerbatimInput{hello.f90}
3
4 \fvset{frame=single,numbers=left,
5       numbersep=3pt}
6 \VerbatimInput{hello.f90}
7
8 \VerbatimInput[firstline=3,
9       rulecolor=\color{green}]
10 {hello.f90}
11
12 \VerbatimInput[frame=lines,
13       fontshape=sl,fontsize=\footnotesize]
14 {hello.f90}

```

We can make use of the ‘defineinactive’ parameter to set the comment lines in the program text in a different style:

```
! A "hello" program

program hello
  print *, "Hello world"
end program hello
```

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

It is important to note that if the contents of the file does not fit on the page, it will be automatically broken across pages as needed (unless the `samepage` parameter has been set to true).

There is also a `VerbatimOut` environment to write verbatim text to an output file, 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 Automatic pretty printing

Obviously, automatic *pretty printing* is outside the scope of this package. Nevertheless, this is specially interesting for verbatim inclusion of programming code files or fragments. In the  $\text{\TeX}$  world (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  $\text{\TeX}$  based solution): the ‘LGrind’ [3] system, currently maintained by Michael PIEFEL, and the ‘listings’ [4] package from Carsten HEINZ.

Future versions of ‘fancyvrb’ and ‘listings’ packages are planned to cooperate, which will offer great advantages to both users of the two actual packages, and will allow ‘fancyvrb’ users to have automatic pretty printing of programming codes.

## 8 Known problems

- Vladimir VOLOVICH <vvv@vvv.vsu.ru> reported that the special character `\th`, available with T1 encoding, can’t be included as verbatim with ‘fancyvrb’. It can be true for other special characters too.

## 9 Thanks

For interesting comments and suggestions, we would like to thank specially (alphabetic order): Philippe ESPERET `esperet@marie.polytechnique.fr`, Michael FRIENDLY `friendly@hotspur.psych.yorku.ca`, Rolf NIEPRASCHK `niepraschk@gmx.de` and for bug reports Mario HASSLER `HASSLER@ippnv2.ipp.kfa-juelich.de`, Mikhail KOLODIN `myke@morrigan.spb.su`, Andreas Matthias, Ulrich M. Schwarz, and Vladimir VOLOVICH `<vvv@vvv.vsu.ru>`.

## 10 Conclusion

There are a few other possibilities that we have not described here. Note specially that it is possible to define a customization file (`fancyvrb.cfg`) loaded at the end of the package, to store definitions of your customized commands and environments and to redefine the attributes of existing ones.

## References

- [1] Timothy VAN ZANDT, *Documentation for ‘fancybox’: Box tips and tricks for  $\text{\LaTeX}$* . Available from CTAN: `macros/latex/contrib/supported/fancybox`, 1993.
- [2] Timothy VAN ZANDT, *‘fancyvrb’: Fancy Verbatims in  $\text{\LaTeX}$* . 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.

## 11 Driver file

The next bit of code contains the documentation driver file for  $\text{\TeX}$ , i.e., the file that will produce the documentation you are currently reading. It will be extracted from this file by the `docstrip` program.

```
1 (*driver)
2 \documentclass{ltxdoc}
3 \GetFileInfo{fancyvrb.dtx}
4 \usepackage{color}
5 \usepackage{fancyvrb,shortvrb}
6 \usepackage[T1]{fontenc}
7 \usepackage[latin1]{inputenc}
8 \usepackage[charter]{mathdesign}
```

```

9 \usepackage{url}
10 \newif\ifChangeBar
11 \IfFileExists{changebar.sty}%
12     {\ChangeBartrue\usepackage[dvips,rightbars]{changebar}}{}
13 \EnableCrossrefs
14 \CodelineIndex
15 \RecordChanges
16 \OnlyDescription           % Comment it for implementation details
17 %\Oldmakeindex            % Uncomment if your MakeIndex is pre-0.9
18 \hbadness=7000           % Over and under full box warnings
19 \hfuzz=3pt
20 \begin{document}
21   \DocInput{fancyvrb.dtx}
22 \end{document}
23 /driver)

```

## 12 ‘fancyvrb’ code

<\*fancyvrb>

***Disclaimer (D.G./S.R.):** This is the original comments of the code by Timothy VAN ZANDT. We have not change them.*

### 12.1 Preamble

What we need.

```

24 \NeedsTeXFormat{LaTeX2e}
    Who we are.
25 \def\fileversion{2.7a, with DG/SPQR fixes, and firstline=lastline fix}
26 \def\filedate{2008/02/07}
27 \ProvidesPackage{fancyvrb}[\filedate]
28 \message{Style option: ‘fancyvrb’ v\fileversion \space <\filedate> (tvz)}
29 \csname fancyvrb@loaded\endcsname
30 \let\fancyvrb@loaded\endinput

```

### 12.2 Errors

```

31 \def\FV@Error#1#2{%
32   \edef\@tempc{#2}\expandafter\errhelp\expandafter{\@tempc}%
33   \errmessage{FancyVerb Error:^^J\space\space #1^^J}}
34
35 \def\FV@eha{Your command was ignored. Type <return> to continue.}

```

### 12.3 Verbatim footnotes

Color has to be protected.

Won’t work for some definitions of \@makefnstext. If it weren’t for \@makefnstext, I would do this properly by defining a footnote environment.

`\VerbatimFootnotes`

```
36 %% DG/SR modification begin - Jan. 21, 1998
37 %% Suggested by Bernard Gaulle to solve a compatibility problem with 'french'
38 %% (it introduce the restriction to put \VerbatimFootnotes AFTER the preamble)
39 %%\def\VerbatimFootnotes{\let\@footnotetext\V@footnotetext}
40 \let\V@footnote\footnote
41 \def\VerbatimFootnotes{%
42 \let\@footnotetext\V@footnotetext%
43 \let\footnote\V@footnote}
44 %% DG/SR modification end
```

`\V@footnotetext`

```
45 \long\def\V@footnotetext{%
46 \afterassignment\V@@footnotetext
47 \let\@tempa}
```

`\V@@footnotetext`

```
48 \def\V@@footnotetext{%
49 \insert\footins\bgroup
50 \csname reset@font\endcsname
51 \footnotesize
52 \interlinepenalty\interfootnotelinepenalty
53 \splittopskip\footnotesep
54 \splitmaxdepth\dp\strutbox
55 \floatingpenalty \@MM
56 \hsize\columnwidth
57 \@parboxrestore
58 \edef\@currentlabel{\csname p@footnote\endcsname\@thefnmark}%
59 \@makefnint{%
60 \rule{\z@}{\footnotesep}%
61 \bgroup
62 \aftergroup\V@@@footnotetext
63 \ignorespaces}
```

`\V@@@footnotetext`

```
64 \def\V@@@footnotetext{\strut\egroup}
```

## 12.4 key=value

```
65 \RequirePackage{keyval}
```

Extensions to keyval.sty:

```
\define@booleankey{<family>}{<key>}{<iftrue>}{<iffalse>}
```

Defines a <key> for <family> that executes:

<iftrue> clause when value begins with 't' or 'T', or is omitted.

<iffalse> clause otherwise.

`\define@booleankey`

```
66 \def\define@booleankey#1#2#3#4{%
67 \@namedef{KV@#1@#2@default}{#3}%
```

```

68 \@namedef{KV@#1@#2@false}{#4}%
69 \@namedef{KV@#1@#2}##1{\KV@booleankey{##1}{#1}{#2}}

\KV@booleankey
70 \def\KV@booleankey#1#2#3{%
71 \edef\@tempa{#1}\expandafter\KV@booleankey\@tempa\relax\@nil{#2}{#3}}

\KV@@booleankey
72 \def\KV@@booleankey#1#2\@nil#3#4{%
73 \@nameuse{KV@#3@#4}\if t#1default\else\if T#1default\else false\fi\fi}}

\FV@None
74 \def\FV@None{none}

\FV@Auto
75 \def\FV@Auto{auto}

\fvset
76 \def\fvset#1{\setkeys{FV}{#1}}

\FV@Command is for verbatim commands. Example:
\def\VerbatimInput{\FV@Command}{\VerbatimInput}
\FV@Command{<key=value>}{<name>}:

1. Defines \FV@KeyValues to be <key=value>.
2. Looks for *, and adds 'showspaces' to \FV@KeyValues if found.
3. Looks for [<key=value>] argument, and adds it to \FV@KeyValues if found.
4. Executes \FVC@<name>.

\FV@Command
77 \def\FV@Command#1#2{%
78 \@ifstar
79 {\def\FV@KeyValues{#1,showspaces}\FV@@Command{#2}}%
80 {\def\FV@KeyValues{#1}\FV@@Command{#2}}}

\FV@@Command
81 \def\FV@@Command#1{%
82 \@ifnextchar[%
83 {\FV@GetKeyValues{\@nameuse{FVC@#1}}}%
84 {\@nameuse{FVC@#1}}}

\FV@GetKeyValues
85 \def\FV@GetKeyValues#1[#2]{%
86 \expandafter\def\expandafter\FV@KeyValues\expandafter{\FV@KeyValues,#2}#1}

```

`\@CustomVerbatimCommand`

```
87 \def\@CustomVerbatimCommand#1#2#3#4{%
88 \begingroup\fvset{#4}\endgroup % If there are errors, it easier to locate.
89 %% DG/SR modification begin - Jan. 13, 1998
90 %% \def\@tempa##1##2\@nil{\def\@tempa{##2}}%
91 %% \expandafter\@tempa\string#3\@empty\@nil
92 %% \@ifundefined{FVC@\@tempa}%
93 \@ifundefined{FVC@#3}%
94 %% DG/SR modification end
95 {\FV@Error{Command '\string#3' is not a FancyVerb command.}\@eha}%
96 {#1{#2}{\FV@Command{#4}{#3}}}}
```

`\CustomVerbatimCommand`

```
97 \def\CustomVerbatimCommand{\@CustomVerbatimCommand\newcommand}
```

`\RecustomVerbatimCommand`

```
98 \def\RecustomVerbatimCommand{\@CustomVerbatimCommand\renewcommand}
```

`\FV@Environment` is for verbatim environments. Example:

```
\def\Verbatim{\FV@Environment}{\Verbatim}
\@namedef{Verbatim*}{\FV@Environment{showsaces}{Verbatim}}
\FV@Environment{<key=value>}{<name>}:

```

1. Defines `\FV@KeyValues` to be `<key=value>`.
2. Sets `\catcode'\^^M=13`, checks for `'[`, and resets `\catcode'\^^M=5`.
3. Reads `[<key=value>]` and adds it to `\FV@KeyValues` if `'[` was found.
4. Executes `\FVB@<name>`.

`\FV@Environment` is like `\FV@Command`, except:

- It omits step 2 (looking for `*`), and
- It sets `\catcode'\^^M=13` when checking for the `'[` of the optional argument, so that it does not skip over `^^M` if there is no `'[`.

`\FV@Environment`

```
99 \def\FV@Environment#1#2{%
100 \def\FV@KeyValues{#1}%
101 \catcode'\^^M=\active
102 \@ifnextchar[%
103 {\catcode'\^^M=5 \FV@GetKeyValues{\@nameuse{FVB@#2}}}%
104 {\catcode'\^^M=5 \@nameuse{FVB@#2}}}
```

`\CustomVerbatimEnvironment`

```
105 \def\CustomVerbatimEnvironment{\@CustomVerbatimEnvironment\newenvironment}
```

```

\RecustomVerbatimEnvironment
106 \def\RecustomVerbatimEnvironment{\@CustomVerbatimEnvironment\renewenvironment}

\@CustomVerbatimEnvironment
107 \def\@CustomVerbatimEnvironment#1#2#3#4{%
108   \begingroup\fvset{#4}\endgroup % If there are errors, it easier to locate.
109   \@ifundefined{FVB@#3}%
110     {\FV@Error{‘#3’ is not a FancyVerb environment.}\@eha}%
111     {#1{#2}{\FV@Environment{#4}{#3}}{\@nameuse{FVE@#3}}}%
112     #1{#2*}{\FV@Environment{#4,showspaces}{#3}}{\@nameuse{FVE@#3}}}}

\DefineVerbatimEnvironment
113 \def\DefineVerbatimEnvironment#1#2#3{%
114   \@namedef{#1}{\FV@Environment{#3}{#2}}%
115   \@namedef{end#1}{\@nameuse{FVE@#2}}%
116   \@namedef{#1*}{\FV@Environment{#3,showspaces}{#2}}%
117   \@namedef{end#1*}{\@nameuse{FVE@#2}}}}

Then commands or environments set key values with \FV@UseKeyValues,
within a group.

\FV@UseKeyValues
118 \def\FV@UseKeyValues{%
119   \ifx\FV@KeyValues\empty\else
120     \def\KV@prefix{KV@FV@}%
121     \expandafter\KV@do\FV@KeyValues,\relax,%
122     \def\FV@KeyValues{%
123     \fi}

```

## 12.5 Scanning

Scanning macros:

- read a line at a time from an environment or a file,
- save the line in \FV@Line, and
- execute \FV@PreProcessLine.

\FV@PreProcessLine:

- Determines whether line is in a user-specified ranges, and if so,
- Strips the line of a user-specified no. of tokens, and
- Executes \FV@ProcessLine{<line>}.

Two things are common to scanning an environment or reading a file:

- \FV@CatCodes  $\implies$  Change \catcode’s.
- \FV@PreProcessLine  $\implies$  See above.

## 12.6 Codes

`\FV@CatCodes`

```
124 \def\FV@CatCodes{%
125   \let\do\@makeother\dospecials % The usual stuff.
126   \FV@ActiveWhiteSpace % See below.
127   \FV@FontScanPrep % See below.
128   \FV@CatCodesHook % A style hook.
129   \FancyVerbCodes} % A user-defined hook.
```

`\FV@ActiveWhiteSpace`

```
130 \def\FV@ActiveWhiteSpace{%
131   \catcode'\^^M=\active % End of line
132   \catcode'\ =\active % Space
133   \catcode'\^^I=\active} % Tab
```

CM Ligatures:

```
Left-quote: “ i ð
Letter f: ff fi fl ffi ffl
Hyphen: -- ---
```

`\FV@CatCodesHook`

```
134 \def\FV@CatCodesHook{}
```

`\FV@AddToHook`

```
135 \def\FV@AddToHook#1#2{%
136   \expandafter\def\expandafter#1\expandafter{#1#2\relax}}
```

`\FancyVerbCodes`

```
137 \define@key{FV}{codes}[]{\def\FancyVerbCodes{#1\relax}}
138 \define@key{FV}{codes*}{%
139   \expandafter\def\expandafter\FancyVerbCodes\expandafter{%
140     \FancyVerbCodes#1\relax}}
141 \fvset{codes}

142 \define@key{FV}{commandchars}[\\{\}]%
143   {\def\@tempa{#1}%
144     \ifx\@tempa\FV@None
145       \let\FV@CommandChars\relax
146     \else
147       \FV@DefineCommandChars#1\relax\relax\relax
148     \fi}
```

`\FV@DefineCommandChars`

```
149 \def\FV@DefineCommandChars#1#2#3{%
150   \def\FV@CommandChars{%
151     \catcode'#1=0\relax\catcode'#2=1\relax\catcode'#3=2\relax}}

152 \FV@AddToHook\FV@CatCodesHook\FV@CommandChars
```

`\FV@CommentChar`

```
153 \define@key{FV}{commentchar}[\%]{%
154   \def\@tempa{#1}%
155   \ifx\@tempa\FV@None
156     \let\FV@CommentChar\relax
157   \else
158     \def\FV@CommentChar{\catcode'#1=14}%
159   \fi}

160 \FV@AddToHook\FV@CatCodesHook\FV@CommentChar
161 \fvset{commandchars=none,commentchar=none}
```

## 12.7 Preprocess line

This is tedious and takes up macro space, but it doesn't slow things down much when the parameters are not used.

`start`, `stop` and `gobble` parameters:

if value is a number, then after the assignment the next token is `\relax`.

"Preprocessing the line" involves:

- Checking whether this line falls in the range specified by `start` or `stop` strings, or `firstline` or `lastline` line-numbers, or a mixture.
- Gobble'ing initial characters.

```
162 \define@key{FV}{firstline}{%
163   \afterassignment\FV@ParseStart\@tempcnta=0#1\relax\@nil{#1}}
```

`\FV@ParseStart`

```
164 \def\FV@ParseStart#1\relax\@nil#2{%
165   \ifx\@nil#1\@nil
166     \edef\FancyVerbStartNum{\the\@tempcnta}%
167     \let\FancyVerbStartString\relax
168   \else
169     \edef\FancyVerbStartString{#2}%
170   \fi}
```

`\KV@FV@firstline@default`

```
171 \def\KV@FV@firstline@default{%
172   \let\FancyVerbStartNum\z@
173   \let\FancyVerbStartString\relax}

174 \define@key{FV}{lastline}{%
175   \afterassignment\FV@ParseStop\@tempcnta=0#1\relax\@nil{#1}}
```

`\FV@ParseStop`

```
176 \def\FV@ParseStop#1\relax\@nil#2{%
177   \ifx\@nil#1\@nil
178     \edef\FancyVerbStopNum{\the\@tempcnta}%
```

```

179 \let\FancyVerbStopString\relax
180 \else
181 \edef\FancyVerbStopString{#2}%
182 \fi}

```

`\KV@FV@lastline@default`

```

183 \def\KV@FV@lastline@default{%
184 \let\FancyVerbStopNum\z@
185 \let\FancyVerbStopString\relax}

```

```
186 \fvset{firstline,lastline}
```

```
187 \newcount\FV@CodeLineNo
```

If `\FV@FindStartStop` determines the line should be printed, it executes `\FV@@PreProcessLine`

`\FV@PreProcessLine`

```

188 \def\FV@PreProcessLine{%
189 \global\advance\FV@CodeLineNo\@ne
190 \FV@FindStartStop}

```

`\FV@@PreProcessLine`

```

191 \def\FV@@PreProcessLine{%
192 \FV@StepLineNo
193 \FV@Gobble
194 \expandafter\FV@ProcessLine\expandafter{\FV@Line}}

```

The definition of `\FV@FindStartStop` changes, so that we don't have to check irrelevant conditions with each line.

Here's how it works; for simplicity we assume `firstline` and `lastline` have been specified, rather than start or stop strings.

The first time `\FV@FindStartStop` is called:

1. It calls `\FV@DefineFindStart`. (The name roughly means "define `\FV@FindStartStop` to detect when we've reached the start of the specified range".) This checks the range parameters that have been specified, and then redefines `\FV@FindStartStop` (yes, the function that has called this code) to contain only the necessary checks, for efficiency reasons:
  - If we haven't printed any lines yet, which is true when `\FV@FindStartStop` is first called, we only need to check the current line-number against `firstline`, so we redefine `\FV@FindStartStop` to be `\FV@FindStartNum`.
  - If `firstline` isn't set, we only have to check the current line-number against `lastline`, so we redefine `\FV@FindStartStop` to be `\FV@DefineFindStop`. (The name roughly means "define `\FV@FindStartStop` to detect when we've reached the stop (end) of the specified range", but unlike the START case, it includes code `(\FV@@PreProcessLine)` to print the current line.)

2. It then calls the newly-defined `\FV@FindStartStop` to see if the current line being processed is within the range to be printed.

`\FV@FindStartStop` is redefined once as described above, but it can be redefined again:

1. When we have reached `firstline`, i.e. have entered the range to be printed, we redefine `\FV@FindStartStop` to be `\FV@DefineFindStop`. (We know we've reached the start, so there's no point checking that again on future calls.)
2. When we reach `lastline` and have printed it, we won't want to print any more lines, so we redefine `\FV@FindStartStop` to `\relax` so we do nothing (and do it efficiently).

Note that when `firstline = lastline` (= 6, say), it is `\FV@FindStartNum` (which `\FV@FindStartStop` has initially been redefined to) that notices when we reach line 6, and prints it, and redefines `\FV@FindStartStop` to be `\FV@DefineFindStop`. Then we read line 7, so `\FV@DefineFindStop` has to check whether we've passed `lastline`, and therefore shouldn't print the current line, before redefining `\FV@FindStartStop` to `\relax` as explained above.

By contrast, when `firstline` is less than `lastline` (= 6, 8, say), when we reach line 8, `\FV@DefineFindStop` sees that we're at the end of the range, and *does* print the line, before redefining `\FV@FindStartStop` to `\relax` as above.

This slight weirdness is because several of the macros are defining themselves and one another, resulting in a slightly weird execution flow.

`\FV@FindStartStop`

```
195 \def\FV@FindStartStop{\FV@DefineFindStart\FV@FindStartStop}
196 %% \def\FV@DefinePreProcessLine{%
197 %%   \setcounter{FancyVerbLine}{0}%
198 %%   \FV@DefineFindStart}
```

`\FV@DefineFindStart`

```
199 \def\FV@DefineFindStart{%
200   \ifx\FancyVerbStartString\relax
201     \ifnum\FancyVerbStartNum<\tw@
202       \FV@DefineFindStop
203     \else
204       \let\FV@FindStartStop\FV@FindStartNum
205     \fi
206   \else
207     \let\FV@FindStartStop\FV@FindStartString
208   \fi}
```

```

\FV@FindStartNum
209 \def\FV@FindStartNum{%
210 \ifnum\FancyVerbStartNum>\FV@CodeLineNo\else
211 \FV@DefineFindStop
212 \expandafter\FV@@PreProcessLine
213 \fi}

\FV@FindStartString
214%% SR modification begin - 1996
215 \def\FV@FindStartString{%
216 \expandafter\FV@@FindStartString
217 {\meaning\FV@Line}%
218 {\meaning\FancyVerbStartString}%
219 }

\FV@@FindStartString
220 \def\FV@@FindStartString#1#2{%
221 \edef\@fooA{#1}\edef\@fooB{#2}%
222 \ifx\@fooA\@fooB
223 \FV@DefineFindStop
224 \fi
225 }
226%% SR modification end

\FV@DefineFindStop
227 \def\FV@DefineFindStop{%
228 \ifx\FancyVerbStopString\relax
229 \ifnum\FancyVerbStopNum<\@ne
230 \let\FV@FindStartStop\FV@@PreProcessLine
231 \else
232 \let\FV@FindStartStop\FV@FindStopNum
233 \fi
234 \else
235 \let\FV@FindStartStop\FV@FindStopString
236 \fi}

\FV@FindStopNum
237 \def\FV@FindStopNum{%
238 \ifnum\FancyVerbStopNum>\FV@CodeLineNo
239 \else
240 \let\FV@FindStartStop\relax
241 \ifeof\FV@InFile\else
242 \immediate\closein\FV@InFile
243 \fi
244 \fi
245 \ifnum\FancyVerbStopNum<\FV@CodeLineNo
246 \else
247 \FV@@PreProcessLine
248 \fi}

```

```

\FV@FindStopString
249 %% SR modification begin - 1996
250 \def\FV@FindStopString{%
251 \expandafter\FV@@FindStopString
252 {\meaning\FV@Line}%
253 {\meaning\FancyVerbStopString}%
254 }

```

```

\FV@@FindStopString
255 \def\FV@@FindStopString#1#2{%
256 \edef\fooA{#1}\edef\fooB{#2}%
257 \ifx\fooA\fooB
258 \let\FV@FindStartStop\relax
259 \ifeof\FV@InFile\else
260 \immediate\closein\FV@InFile
261 \fi
262 \else
263 \expandafter\FV@@PreProcessLine
264 \fi}
265 %% SR modification end

```

Gobblings. \FV@Gobble does nothing, or strips some tokens from the line and stores the result in \FV@Line again. We use  $\TeX$ 's \renewcommand to define a command for gobbling up to 9 arguments. This is not the same as removing 9 tokens, but is easier.

```

\FV@@@Gobble
266 \def\FV@@@Gobble{%
267 \expandafter\expandafter\expandafter\FV@@@Gobble
268 \expandafter\FV@@@Gobble\FV@Line
269 \@nil\@nil\@nil\@nil\@nil\@nil\@nil\@nil\@nil\@nil\@nil}

```

```

\FV@@@Gobble
270 \def\FV@@@Gobble#1\@nil#2\@nil{\def\FV@Line{#1}}

```

```

\FV@Gobble
271 \define@key{FV}{gobble}{%
272 \@tempcnta=#1\relax
273 \ifnum\@tempcnta<\@ne
274 \let\FV@Gobble\relax
275 \else
276 \ifnum\@tempcnta>9
277 \FV@Error{gobble parameter must be less than 10}\FV@eha
278 \else
279 \renewcommand{\FV@@@Gobble}{\@tempcnta}{%
280 \let\FV@Gobble\FV@@@Gobble
281 \fi
282 \fi}

```

`\FV@@@Gobble`

```
283 \def\FV@@@Gobble{}
```

`\KV@FV@gobble@default`

```
284 \def\KV@FV@gobble@default{\let\FV@Gobble\relax}
```

```
285 \fvset{gobble}
```

## 12.8 Scanning environments

`\FV@Scan`

```
286 \def\FV@Scan{%
287   \FV@CatCodes
288   \VerbatimEnvironment
289   \FV@DefineCheckEnd
290   \FV@BeginScanning}
```

`\VerbatimEnvironment:`

This saves the name of the current environment as `\FV@EnvironName`, if the latter is not already defined. Then `\FV@CheckEnd` knows how to find the end as long as either:

- `\begin` and `\end` are not used within the definition of the environment, OR
- `\VerbatimEnvironment` is used in the definition before the first `\begin`.

`\VerbatimEnvironment`

```
291 \def\VerbatimEnvironment{%
292   \ifx\FV@EnvironName\relax\xdef\FV@EnvironName{\@currenenv}\fi}
293 \let\FV@EnvironName\relax
```

## 12.9 Check end

We have to check the argument of the first `\end{}` in each line, compare it with `\FV@EnvironName`, and return `\iftrue` if it matches and `\iffalse` otherwise.

There are four cases (R=regular):

```
case          : i ii iii iv
catcode of \ : R 12 12 R
catcode of {} : R R 12 12
```

For uniformity, we use `![]` instead of `\{}` in all the definitions.

We first set the catcodes of `\{}` to those in effect in the verbatim environment.

Then we define:

```
!def!FV@CheckEnd#1[!FV@CheckEnd#1\end{!}@nil]
```

If have their usual catcodes, we define:

```
!def!FV@CheckEnd#1\end#2#3!@nil[!def!@tempa[#2]]
```

If have catcode 12, we define:

```
!def!FV@@CheckEnd#1\end{#2}#3!@nil[!def!@tempa[#2]]

294 \begingroup
295 \catcode'\!=0
296 \catcode'\ [=1
297 \catcode'\ ]=2

    Case i:

298 !gdef!FV@CheckEnd@i#1[!FV@@CheckEnd#1\end{!}@nil]
299 !gdef!FV@@CheckEnd@i#1\end#2#3!@nil[!def!@tempa[#2]!def!@tempb[#3]]
300 !gdef!FV@@@CheckEnd@i[\end{!}]

    Case ii:

301 \catcode'\ =12
302
303 !gdef!FV@CheckEnd@ii#1[!FV@@CheckEnd#1\end{!}@nil]
304 !gdef!FV@@CheckEnd@ii#1\end#2#3!@nil[!def!@tempa[#2]!def!@tempb[#3]]
305 !gdef!FV@@@CheckEnd@ii[\end{!}]

    Case iii:

306 !catcode'\ {=12
307 !catcode'\ }=12
308
309 !gdef!FV@CheckEnd@iii#1[!FV@@CheckEnd#1\end{!}@nil]
310 !gdef!FV@@CheckEnd@iii#1\end{#2}#3!@nil[!def!@tempa[#2]!def!@tempb[#3]]
311 !gdef!FV@@@CheckEnd@iii[\end{!}]

    Case iv:

312 !catcode'\ =0
313
314 !gdef!FV@CheckEnd@iv#1[!FV@@CheckEnd#1\end{!}@nil]
315 !gdef!FV@@CheckEnd@iv#1\end{#2}#3!@nil[!def!@tempa[#2]!def!@tempb[#3]]
316 !gdef!FV@@@CheckEnd@iv[\end{!}]

317 \endgroup
```

\FV@BadCodes

```
318 \def\FV@BadCodes#1{%
319   \FV@Error
320   {\string\catcode\space of \expandafter\@gobble\string#1 is wrong:
321   \the\catcode'#1}%
322   {Only the following catcode values are allowed:
323   ^^J\@spaces \expandafter\@gobble\string\ \space\space --> 0 or 12.
324   ^^J\@spaces \string{ \string} --> 1 and 2, resp., or both 12.
325   ^^JTo get this error, either you are a hacker or you got bad advice.}%
326   \def\FV@CheckEnd##1{\iftrue}}
```

\FV@DefineCheckEnd

```
327 \def\FV@DefineCheckEnd{%
328   \ifnum\catcode'\ =\z@
```

```

329 \ifnum\catcode'\{=\@ne
330 \let\FV@CheckEnd\FV@CheckEnd@i
331 \let\FV@@CheckEnd\FV@@CheckEnd@i
332 \let\FV@@@CheckEnd\FV@@@CheckEnd@i
333 \else
334 \ifnum\catcode'\{=12
335 \let\FV@CheckEnd\FV@CheckEnd@iv
336 \let\FV@@CheckEnd\FV@@CheckEnd@iv
337 \let\FV@@@CheckEnd\FV@@@CheckEnd@iv
338 \else
339 \FV@BadCodes\{%
340 \fi
341 \fi
342 \else
343 \ifnum\catcode'\}=12
344 \ifnum\catcode'\{=\@ne
345 \let\FV@CheckEnd\FV@CheckEnd@ii
346 \let\FV@@CheckEnd\FV@@CheckEnd@ii
347 \let\FV@@@CheckEnd\FV@@@CheckEnd@ii
348 \else
349 \ifnum\catcode'\{=12
350 \let\FV@CheckEnd\FV@CheckEnd@iii
351 \let\FV@@CheckEnd\FV@@CheckEnd@iii
352 \let\FV@@@CheckEnd\FV@@@CheckEnd@iii
353 \else
354 \FV@BadCodes\{%
355 \fi
356 \fi
357 \else
358 \FV@BadCodes\}%
359 \fi
360 \fi}

```

## 12.10 Line-by-line scanning

We first skip everything after the beginning of the environment.

`\FV@BeginScanning`

```

361 \begingroup
362 \catcode'\^^M=\active
363 \gdef\FV@BeginScanning#1^^M{%
364 \def\@tempa{#1}\ifx\@tempa\@empty\else\FV@BadBeginError\fi%
365 \FV@GetLine}%
366 \endgroup

```

`\FV@BadBeginError`

```

367 \def\FV@BadBeginError#1{%
368 \expandafter\@temptokena\expandafter{\@tempa}%
369 \FV@Error

```

```

370 {Extraneous input ‘\the\@temptokena’ between
371 \string\begin{\FV@EnvironName}[<key=value>] and line end}%
372 {This input will be discarded. Hit <return> to continue.}}

```

If \FancyVerbGetLine does not find a  $\sim$ M, then we are at the end of the file, and \FV@EOF attempts to terminate the document. Otherwise, \FV@EOF is gobbled by \FancyVerbGetLine.

\FV@GetLine

```

373 %% DG/SR modification begin - May. 18, 1998 (added code to turn off ligatures)
374 %% \def\FV@GetLine{\expandafter\FV@CheckScan\FancyVerbGetLine}
375 \def\FV@GetLine{\@noligs\expandafter\FV@CheckScan\FancyVerbGetLine}
376 %% DG/SR modification end

```

\FancyVerbGetLine

```

377 \begingroup
378 \catcode‘\simM=\active
379 \gdef\FancyVerbGetLine#1\simM{%
380 \@nil%
381 \FV@CheckEnd{#1}%
382 \ifx\@tempa\FV@EnvironName% % True if end is found
383 \ifx\@tempb\FV@@@CheckEnd\else\FV@BadEndError\fi%
384 \let\next\FV@EndScanning%
385 \else%
386 \def\FV@Line{#1}%
387 \def\next{\FV@PreProcessLine\FV@GetLine}%
388 \fi%
389 \next}%
390 \endgroup

```

\FV@BadEndError

```

391 \def\FV@BadEndError{%
392 \expandafter\@temptokena\expandafter{\@tempb}%
393 \FV@Error
394 {Extraneous input ‘\the\@temptokena’ between
395 \string\end{\FV@EnvironName} and line end}%
396 {This input will be discarded. Type <return> to continue.}}

```

\FV@EndScanning

```

397 \def\FV@EndScanning{%
398 \edef\next{\noexpand\end{\FV@EnvironName}}%
399 \global\let\FV@EnvironName\relax
400 \next}

401 \@ifundefined{@currentvline}{\let\@currentvline\@empty}{

```

\FV@CheckScan

```

402 \def\FV@CheckScan#1{\@ifnextchar\@nil{\@gobble}{\FV@EOF}}
403 \def\FV@CheckScan#1{\ifx\@nil#1\@empty\else\expandafter\FV@EOF\fi}

```

`\FV@EOF`

```
404 \def\FV@EOF{%
405   \FV@Error{Couldn't find '\string\end{\FV@EnvironName}' to end
406     a verbatim environment\@currentline.}%
407   {Probably you mistyped the environment name or included an extraneous
408     ^^Jspace, or are using an improperly defined verbatim environment.
409     ^^JHit return and I will try to terminate this job.}%
410   \FV@EndScanning
411   \end{document}}
```

## 12.11 Input

```
412 \newread\FV@InFile
```

`\FV@Input`

```
413 \def\FV@Input#1{%
414   \immediate\openin\FV@InFile #1\relax
415   \ifeof\FV@InFile
416     \FV@Error{No verbatim file #1}\FV@eha
417     \immediate\closein\FV@InFile
418   \else
419     \FV@CatCodes
420     \expandafter\FV@@Input
421   \fi}
```

$\TeX$  reports EOF when reading after the last newline character.

Thus, we read to `InLine`, and if  $\TeX$  reports EOF:

- If `InLine=Empty`, previous line was last line in file (file ends `nl`).
- Otherwise, `InLine` holds last line in file (file doesn't end in `nl`).

`\FV@@Input` handles both cases correctly.

`\FV@@Input`

```
422 \def\FV@@Input{%
423   \def\FV@Line{}%
424   \FV@ReadLine
425   \ifeof\FV@InFile
426     \ifx\FV@Line\@empty\else
427       \FV@PreProcessLine
428     \fi
429   \immediate\closein\FV@InFile
430   \else
431     \FV@PreProcessLine
432   \expandafter\FV@@Input
433   \fi}
```

We also want to handle true comment characters correctly. This means that we keep accumulating text in `InLine` until we find a `^^M` (indicating that the line did not contain a comment character).

`\FV@ReadLine`

```
434 \begingroup
435 \catcode'\^^M=\active
436 \gdef\FV@ReadLine{%
437   \ifeof\FV@InFile\else
438     \immediate\read\FV@InFile to\@tempa%
439     \expandafter\FV@@ReadLine\@tempa^^M\relax^^M\@nil%
440   \fi}
```

#2 is empty if line ends in ^^M, #2=\relax otherwise

`\FV@@ReadLine`

```
441 \gdef\FV@@ReadLine#1^^M#2^^M#3\@nil{%
442   \expandafter\def\expandafter\FV@Line\expandafter{%
443     \FV@Line#1}%
444   \ifx\relax#2\@empty\expandafter\FV@ReadLine\fi}%
445 \endgroup
```

## 12.12 Formatting – Common components

Some things that are common to all verbatim formatting:

`\FV@FormattingPrep`

```
446 \def\FV@FormattingPrep{%
447   \global\FV@CodeLineNo\z@
448   \frenchspacing           % Cancels special punctuation spacing.
449   \FV@SetupFont           % See below.
450   \FV@DefineWhiteSpace    % See below.
451   \FancyVerbDefineActive
452   \FancyVerbFormatCom}    % A user-defined hook (formatcom parameter).
```

Fonts

```
453 \expandafter\ifx\csname selectfont\endcsname\relax
```

`\FV@SetupFont`

```
454 \def\FV@SetupFont{%
455   \FV@BaseLineStretch
456   \ifx\@currsize\small\normalsize\else\small\fi\@currsize
457   \FV@FontSize
458   \FV@FontFamily}
459
460 \else
461
462 \def\FV@SetupFont{%
463   \FV@BaseLineStretch
464   \FV@FontSize
465   \FV@FontFamily
466   \FV@FontSeries
```

```

467 \FV@FontShape
468 \selectfont
469 %% DG/SR modification begin - May. 18, 1998 (added code to turn off ligatures)
470 \@noligs}
471 %% DG/SR modification end

472 \fi

```

\FV@FontSize

```

473 \define@key{FV}{fontsize}{%
474 \def\@tempa{#1}%
475 \ifx\@tempa\FV@Auto
476 \let\FV@FontSize\relax
477 \else
478 \def\FV@FontSize{#1}%
479 \fi}

```

\KV@FV@fontsize@default

```

480 \def\KV@FV@fontsize@default{\let\FV@FontSize\relax}

```

\FV@BaseLineStretch

```

481 \define@key{FV}{baselinestretch}[auto]{%
482 \def\@tempa{#1}%
483 \ifx\@tempa\FV@Auto
484 \let\FV@BaseLineStretch\relax
485 \else
486 \def\FV@BaseLineStretch{\def\baselinestretch{#1}}%
487 \fi}

```

\KV@FV@baselinestretch@default

```

488 \def\KV@FV@baselinestretch@default{\let\FV@BaseLineStretch\relax}

```

```

489 \define@key{FV}{fontfamily}{%
490 \@ifundefined{FV@fontfamily@#1}%
491 {\def\FV@FontScanPrep{\def\FV@FontFamily{\fontfamily{#1}}}
492 {\csname FV@fontfamily@#1\endcsname}}

```

\FV@FontSeries

```

493 \define@key{FV}{fontseries}{%
494 \def\@tempa{#1}%
495 \ifx\@tempa\FV@Auto
496 \let\FV@FontSeries\relax
497 \else
498 \def\FV@FontSeries{\fontseries{#1}}%
499 \fi}

```

\FV@FontShape

```

500 \define@key{FV}{fontshape}{%
501 \def\@tempa{#1}%

```

```

502 \ifx\@tempa\FV@Auto
503 \let\FV@FontShape\relax
504 \else
505 \def\FV@FontShape{\fontshape{#1}}%
506 \fi}

```

Font family styles have to define \FV@FontScanPrep and \FV@FontFamily.

\FV@MakeActive

```

507 \def\FV@MakeActive#1{%
508 \catcode'#1=\active
509 \def\next##1{\expandafter\def\expandafter\FV@MakeUnActive\expandafter{%
510 \FV@MakeUnActive\def##1{\string##1}}}%
511 \begingroup\lccode'~='#1\relax\expandafter\next\expandafter~\endgroup}

```

\FV@MakeUnActive

```

512 \def\FV@MakeUnActive{}
513 \begingroup
514 \catcode'\='=\active

```

\FV@fontfamily@tt

```

515 \gdef\FV@fontfamily@tt{%
516 \def\FV@FontScanPrep{\FV@MakeActive\'}%
517 %% SR modification begin - 1995
518 %% \def\FV@FontFamily{\tt'{{\string'}}}
519 \def\FV@FontFamily{\ttfamily\edef'{{\string'}}}
520 %% SR modification end

```

\FV@fontfamily@cmtt

```

521 \gdef\FV@fontfamily@cmtt{%
522 \def\FV@FontScanPrep{\FV@MakeActive\'}%
523 \def\FV@FontFamily{\edef'{{\string'}}\fontfamily{cmtt}}
524 \endgroup

```

\FV@fontfamily@cmtt-spanish

```

525 \@namedef{FV@fontfamily@cmtt-spanish}{%
526 \def\FV@FontScanPrep{}%
527 \def\FV@FontFamily{\fontfamily{cmtt}}

```

Fix me

\FV@fontfamily@courier

```

528 \@namedef{FV@fontfamily@courier}{%
529 \def\FV@FontScanPrep{}%
530 %% SR modification begin - 1995
531 %% \def\FV@FontFamily{\fontfamily{rprcr}}
532 \def\FV@FontFamily{\fontfamily{pcr}}
533 %% SR modification end

```

```

\FV@fontfamily@helvetica
534 \@namedef{FV@fontfamily@helvetica}{%
535   \def\FV@FontScanPrep{}%
536 %% SR modification begin - 1995
537 %%   \def\FV@FontFamily{\fontfamily{rphv}}
538   \def\FV@FontFamily{\fontfamily{phv}}
539 %% SR modification end

540 \fvset{fontfamily=tt,fontsize=auto,fontshape=auto,fontseries=auto,
541   baselinestretch=auto}

   \FV@DefineWhiteSpace
   We just define the active characters to be ordinary commands, which are easier to
   redefine. We do with any macros that use verbatim text.

542 \begingroup
543 \catcode'\ =\active
544 \catcode'\^^I=\active

\FV@DefineWhiteSpace

545 \gdef\FV@DefineWhiteSpace{\def {\FV@Space}\def^^I{\FV@Tab}}%

546 \endgroup

\FancyVerbDefineActive

547 \define@key{FV}{defineactive}[]{\def\FancyVerbDefineActive{#1\relax}}
548 \define@key{FV}{defineactive*}{%
549   \expandafter\def\expandafter\FancyVerbDefineActive\expandafter{%
550     \FancyVerbDefineActive#1\relax}}
551 \fvset{defineactive}

   \FV@Space:

\FV@Space

552 \define@booleankey{FV}{showspaces}%
553   {\def\FV@Space{\FancyVerbSpace}}%
554   {\def\FV@Space{\ }}

555 {\catcode'\ =12 \gdef\FancyVerbSpace{\tt }}
556 \fvset{showspaces=false}

   \FV@Tab:

\FV@Tab

557 \def\FV@Tab{\hbox to\FancyVerbTabSize\fontdimen2\font{\hss\FV@TabChar}}

\FancyVerbTabSize

558 \define@key{FV}{tabsize}{%
559   \@tempcnta=#1\relax
560   \ifnum\@tempcnta>100
561     \FV@Error{Tab size too large: '\the\@tempcnta'. (Max size = 100)}\FV@eha

```

```

562 \else
563 \edef\FancyVerbTabSize{\the\@tempcnta}%
564 \fi}

```

\FV@TabChar

```

565 \define@booleankey{FV}{showtabs}%
566 {\def\FV@TabChar{\FancyVerbTab}}%
567 {\let\FV@TabChar\relax}

```

```
568 \fvset{tabsize=8,showtabs=false}
```

Here is a weak attempt at a tab character. It may exceed the width of a space character when the verbatim font is small. The only proper way to do this is making it part of the verbatim font.

\FancyVerbTab

```

569 \def\FancyVerbTab{%
570 \valign{%
571 \vfil#\vfil\cr
572 \hbox{\scriptscriptstyle-}\cr
573 \hbox to 0pt{\hss\scriptscriptstyle\angle\mskip -.8mu}\cr
574 \hbox{\scriptstyle\mskip -3mu\mid\mskip -1.4mu}\cr}}

```

Obey Tabs:

```
575 \newbox\FV@TabBox
```

\FV@@ObeyTabsInit

```

576 \def\FV@@ObeyTabsInit{%
577 \@tempdimb=\FancyVerbTabSize\fontdimen\tw\font
578 \edef\FV@ObeyTabSize{\number\@tempdimb}%
579 \advance\@tempdimb\fontdimen\tw\font
580 \advance\@tempdimb-\FancyVerbTabSize sp % Allow for rounding errors.
581 \edef\FV@@ObeyTabSize{\number\@tempdimb}%
582 \let\FV@ObeyTabs\FV@@ObeyTabs
583 \let\FV@Tab\FV@TrueTab}

```

\FV@@ObeyTabs

```

584 \def\FV@@ObeyTabs#1{\setbox\FV@TabBox=\hbox{#1}\box\FV@TabBox}
585 \let\FV@ObeyTabs\relax

```

\FV@TrueTab

```

586 \def\FV@TrueTab{%
587 \egroup
588 \@tempdima=\FV@ObeyTabSize sp\relax
589 \@tempcnta=\wd\FV@TabBox
590 \advance\@tempcnta\FV@@ObeyTabSize\relax
591 \divide\@tempcnta\@tempdima
592 \multiply\@tempdima\@tempcnta

```

```

593 \advance\@tempdima-\wd\FV@TabBox
594 \setbox\FV@TabBox=\hbox\bgroup
595 \unhbox\FV@TabBox\kern\@tempdima\hbox to\z@\hss\FV@TabChar}}

```

`\FV@ObeyTabsInit`

```

596 \define@booleankey{FV}{obeytabs}%
597 {\let\FV@ObeyTabsInit\FV@ObeyTabsInit}%
598 {\let\FV@ObeyTabsInit\relax}

```

```

599 \fvset{obeytabs=false}
    \FancyVerbFormatCom

```

`\FancyVerbFormatCom`

```

600 \define@key{FV}{formatcom}[]{\def\FancyVerbFormatCom{#1}\relax}}
601 \define@key{FV}{formatcom*}{%
602 \expandafter\def\expandafter\FancyVerbFormatCom\expandafter{%
603 \FancyVerbFormatCom#1}\relax}}
604 \fvset{formatcom}

```

`\FancyVerbFormatLine`

```

605 \def\FancyVerbFormatLine#1{\FV@ObeyTabs{#1}}

```

## 12.13 List environments

Some parameters:

`\FV@XLeftMargin`

```

606 \define@key{FV}{xleftmargin}{\def\FV@XLeftMargin{#1}}
607 \let\FV@XLeftMargin\z@

```

`\FV@XRightMargin`

```

608 \define@key{FV}{xrightmargin}{\def\FV@XRightMargin{#1}}
609 \let\FV@XRightMargin\z@

```

`\if@FV@ResetMargins`

```

610 \define@booleankey{FV}{resetmargins}%
611 {\let\if@FV@ResetMargins\iftrue}
612 {\let\if@FV@ResetMargins\iffalse}
613 \fvset{resetmargins=false}

```

`\FV@ListParameterHook`

```

614 \define@key{FV}{listparameters}{\def\FV@ListParameterHook{#1}}
615 \def\FV@ListParameterHook{}

```

`\FancyVerbHFuzz`

```

616 \define@key{FV}{hfuzz}{%
617 \@tempdima=#1}\relax
618 \edef\FancyVerbHFuzz{\number\@tempdima sp}}
619 \fvset{hfuzz=2pt}

```

`\FV@InterLinePenalty`

```
620 \define@booleankey{FV}{samepage}%
621 {\def\FV@InterLinePenalty{\interlinepenalty\@M}}%
622 {\let\FV@InterLinePenalty\relax}
623 \fvset{samepage=false}
```

Lists:

`\FV@List{}` is a rewriting of `\list{...}\item[]`. The rewrite gives me more control. I might not get the vertical spacing exactly the same, but it is more likely that it will get better than worse.

The verbatim environment consists of a series of `\hbox`'s inserted in vertical mode.

We need to take care of the following:

- leftmargin
- rightmargin
- topskip
- botskip
- toppenalty
- botpenalty
- interlinepenalties

Note: A verbatim environment immediately after an `\item` starts on the same line as the `\item`'s label, unless we reset margins. The user can instead have the environment start on a new line by inserting '`\`' between `\item` and the environment.

`\FV@List`

```
624 \def\FV@List#1{%
625   \begingroup
626   \FV@UseKeyValues
627   \FV@LeaveVMode
628   \if@inlabel\else\setbox\@labels=\box\voidb@x\fi
629   \FV@ListNesting{#1}%
630   \FV@ListParameterHook
631   \FV@ListVSpace
632   \FV@SetLineWidth
633   \FV@InterLinePenalty
634   \let\FV@ProcessLine\FV@ListProcessLine@i
635   \FV@CatCodes
636   \FV@FormattingPrep
637   \FV@ObeyTabsInit
638   \FV@BeginListFrame}
```

Cases where we need to leave vmode:

- After an in-line section (`\if@noskipsec=T`).
- After an `\item` command, if we reset margins.

Then we end `\vmode`, using `@noparlist` as a flag if in `vmode`. (Not the usual meaning of `@noparlist`.)

`\FV@LeaveVMode`

```
639 \def\FV@LeaveVMode{%
640   \if@noskipsec
641     \leavevmode
642   \else
643     \if@FV@ResetMargins\if@inlabel\leavevmode\fi\fi
644   \fi
645   \ifvmode \@noparlisttrue\else \@noparlistfalse\unskip\par\fi}
```

`\FV@ListNesting`

```
646 \def\FV@ListNesting#1{%
647   \if@FV@ResetMargins
648     \@listdepth=\z@
649   \else
650     \ifnum \@listdepth>5\relax
651       \@toodeep
652     \else
653       \advance \@listdepth \@ne
654     \fi
655   \fi
656   \rightmargin\z@
657   \csname @list\romannumeral\the \@listdepth\endcsname
658   \ifnum#1=\z@
659     \rightmargin\z@
660     \leftmargin\z@
661   \fi}
```

`\FV@ListVSpace` contains selected parts of `\@trivlist` and `\@item`.

Here are the cases:

Vmode not in label or after `@NOBREAK`:

```
<topskip>    = \topsep + \partopsep + \parskip
<botskip>    = \topsep + \partopsep
<toppenalty> = \@beginparpenalty
<botpenalty> = \@endparpenalty
```

Vmode in label:

```
<topskip>    = \parskip                % Expected anyway.
<botskip>    = \topsep + \partopsep    % Omitted in TeX – a bug?
<toppenalty> = None
<botpenalty> = \@endparpenalty        % Ditto.
```

Vmode after @nobreak:

```
<topskip>    = \parskip
<botskip>    = \topsep + \partopsep
<toppenalty> = None
<botpenalty> = \@endparpenalty
```

Hmode:

```
<topskip>    = \topsep + \parskip
<botskip>    = \topsep
<toppenalty> = \@beginparpenalty
<botpenalty> = \@endparpenalty
```

Notes:

- Except when in label or after nobreak, `\parskip` is added with `\addvspace`, so that net space is:  
$$\text{MAX}\{\text{\topsep} (+\text{\partopsep}) + \text{\parskip}, \text{\lastskip}\}$$
  
(The usual `\@item` works the same way.)
- `\parskip` is added afterwards by a new paragraph, if any.
- `<botskip> == \@topsepadd`

`\FV@ListVSpace`

```
662 \def\FV@ListVSpace{%
663   \@topsepadd\topsep
664   \if@nolist\advance\@topsepadd\partopsep\fi
665   \if@inlabel
666     \vskip\parskip
667   \else
668     \if@nobreak
669       \vskip\parskip
670       \clubpenalty\@M
671     \else
672       \addpenalty\@beginparpenalty
673       \@topsep\@topsepadd
674       \advance\@topsep\parskip
675       \addvspace\@topsep
676     \fi
677   \fi
678   \global\@nobreakfalse
679   \global\@inlabelfalse
680   \global\@minipagefalse
681   \global\@newlistfalse}

\leftmargin      := totalleftmargin
\rightmargin     := totalrightmargin
\@totalleftmargin := totalleftmargin of enclosing environment.
```

`\FV@SetLineWidth`

```
682 \def\FV@SetLineWidth{%
683   \if@FV@ResetMargins\else
684     \advance\leftmargin\@totalleftmargin
685     \fi
686   \advance\leftmargin\FV@XLeftMargin\relax
687   \advance\rightmargin\FV@XRightMargin\relax
688   \linewidth\hsize
689   \advance\linewidth-\leftmargin
690   \advance\linewidth-\rightmargin
691   \hfuzz\FancyVerbHFuzz\relax}
```

We have to insert the right interline penalties (`\interlinepenalty`, `\clubpenalty`, `\widowpenalty`). We could process the environment as one long paragraph and let  $\TeX$  insert the penalties, but this might cause problems for a very long environment.

**Line 1** : Insert `\@labels` (maybe) plus `current_line`

**Line 2** : Save current line

**Line 3** : `penalty = interline + club` ; Insert last line ; Save current line.

**Line 4** : `penalty = interline` ; Insert last line ; Save current line.

Then at the end:

**Next line = 1** : Add null line.

**Next line = 2** : Nothing.

**Next line = 3** : `penalty = interline + club + widow` ; Insert last line.

**Next line = 4** : `penalty = interline + widow` ; Insert last line.

`\FV@ListProcessLine`

```
692 \def\FV@ListProcessLine#1{%
693   \hbox to \hsize{%
694     \kern\leftmargin
695     \hbox to \linewidth{%
696       \FV@LeftListNumber
697       \FV@LeftListFrame
698       \FancyVerbFormatLine{#1}\hss
699 %% DG/SR modification begin - Jan. 28, 1998 (for numbers=right add-on)
700 %%       \FV@RightListFrame}%
701       \FV@RightListFrame
702       \FV@RightListNumber}%
703 %% DG/SR modification end
704     \hss}}
```

```

\FV@ListProcessLine@i
705 \def\FV@ListProcessLine@i#1{%
706   \hbox{%
707     \ifvoid\@labels\else
708       \hbox to \z@\kern\@totalleftmargin\box\@labels\hss}%
709     \fi
710     \FV@ListProcessLine{#1}}%
711 \let\FV@ProcessLine\FV@ListProcessLine@ii}

\FV@ListProcessLine@ii
712 \def\FV@ListProcessLine@ii#1{%
713   \setbox\@tempboxa=\FV@ListProcessLine{#1}%
714   \let\FV@ProcessLine\FV@ListProcessLine@iii}

\FV@ListProcessLine@iii
715 \def\FV@ListProcessLine@iii#1{%
716   {\advance\interlinepenalty\clubpenalty\penalty\interlinepenalty}%
717   \box\@tempboxa
718   \setbox\@tempboxa=\FV@ListProcessLine{#1}%
719   \let\FV@ProcessLine\FV@ListProcessLine@iv}

\FV@ListProcessLine@iv
720 \def\FV@ListProcessLine@iv#1{%
721   \penalty\interlinepenalty
722   \box\@tempboxa
723   \setbox\@tempboxa=\FV@ListProcessLine{#1}}%

\FV@EndList
724 \def\FV@EndList{%
725   \FV@ListProcessLastLine
726   \FV@EndListFrame
727   \@endparenv
728   \endgroup
729   \@endpetrue}

\FV@ListProcessLastLine
730 \def\FV@ListProcessLastLine{%
731   \ifx\FV@ProcessLine\FV@ListProcessLine@iv
732     {\advance\interlinepenalty\widowpenalty\penalty\interlinepenalty}%
733     \box\@tempboxa
734   \else
735     \ifx\FV@ProcessLine\FV@ListProcessLine@iii
736       {\advance\interlinepenalty\widowpenalty
737         \advance\interlinepenalty\clubpenalty
738         \penalty\interlinepenalty}%
739       \box\@tempboxa
740     \else
741       \ifx\FV@ProcessLine\FV@ListProcessLine@i
742         \FV@Error{Empty verbatim environment}{}%

```

```

743     \FV@ProcessLine{ }%
744     \fi
745     \fi
746 \fi}

```

Verbatim environment:

Verbatim

```

747 \def\FV@VerbatimBegin{\FV@List\z@}
748 \def\FV@VerbatimEnd{\FV@endList}

```

\FVB@Verbatim

```

749 \def\FVB@Verbatim{\FV@VerbatimBegin\FV@Scan}

```

\FVE@Verbatim

```

750 \def\FVE@Verbatim{\FV@VerbatimEnd}

```

```

751 \DefineVerbatimEnvironment{Verbatim}{Verbatim}{}

```

With `\UseVerbatim`, we have to take care of some of the things `\end{ }` would do.

\FV@UseVerbatim

```

752 \def\FV@UseVerbatim#1{%
753   \FV@VerbatimBegin#1\FV@VerbatimEnd
754   \@doendpe\global\@ignorefalse\ignorespaces}

```

\VerbatimInput

```

755 \def\VerbatimInput{\FV@Command}{VerbatimInput}}

```

\FVC@VerbatimInput

```

756 \def\FVC@VerbatimInput#1{\FV@UseVerbatim{\FV@Input{#1}}}

```

LVerbatim environment:

LVerbatim

```

757 \def\FV@LVerbatimBegin{\FV@List\@ne}
758 \def\FV@LVerbatimEnd{\FV@endList}

```

\FVB@LVerbatim

```

759 \def\FVB@LVerbatim{\FV@LVerbatimBegin\FV@Scan}

```

\FVE@LVerbatim

```

760 \def\FVE@LVerbatim{\FV@LVerbatimEnd}

```

```

761 \DefineVerbatimEnvironment{LVerbatim}{LVerbatim}{}

```

\FV@LUseVerbatim

```

762 \def\FV@LUseVerbatim#1{%
763   \FV@LVerbatimBegin#1\FV@LVerbatimEnd
764   \@doendpe\global\@ignorefalse\ignorespaces}

```

```

\LVerbatimInput
765 \def\LVerbatimInput{\FV@Command}{\LVerbatimInput}}

\FVC@LVerbatimInput
766 \def\FVC@LVerbatimInput#1{\FV@LUseVerbatim{\FV@Input{#1}}}

      Frames:

\FV@Frame@none
767 \def\FV@Frame@none{%
768   \let\FV@BeginListFrame\relax
769   \let\FV@LeftListFrame\relax
770   \let\FV@RightListFrame\relax
771   \let\FV@endListFrame\relax}

\FV@Frame@single
772 \def\FV@Frame@single{%
773   \let\FV@BeginListFrame\FV@BeginListFrame@Single
774   \let\FV@LeftListFrame\FV@LeftListFrame@Single
775   \let\FV@RightListFrame\FV@RightListFrame@Single
776   \let\FV@endListFrame\FV@endListFrame@Single}

\FV@Frame@lines
777 \def\FV@Frame@lines{%
778   \let\FV@BeginListFrame\FV@BeginListFrame@Lines
779   \let\FV@LeftListFrame\relax
780   \let\FV@RightListFrame\relax
781   \let\FV@endListFrame\FV@endListFrame@Lines}

\FV@Frame@topline
782 \def\FV@Frame@topline{%
783   \let\FV@BeginListFrame\FV@BeginListFrame@Lines
784   \let\FV@LeftListFrame\relax
785   \let\FV@RightListFrame\relax
786   \let\FV@endListFrame\relax}

\FV@Frame@bottomline
787 \def\FV@Frame@bottomline{%
788   \let\FV@BeginListFrame\relax
789   \let\FV@LeftListFrame\relax
790   \let\FV@RightListFrame\relax
791   \let\FV@endListFrame\FV@endListFrame@Lines}

\FV@Frame@leftline
792 %% To define a frame with only a left line
793 \def\FV@Frame@leftline{%
794   % To define the \FV@FrameFillLine macro (from \FV@BeginListFrame)
795   \ifx\FancyVerbFillColor\relax

```

```

796 \let\FV@FrameFillLine\relax
797 \else
798 \@tempdima\FV@FrameRule\relax
799 \multiply\@tempdima-\tw@
800 \edef\FV@FrameFillLine{%
801 {\noexpand\FancyVerbFillColor{\vrule\@width\number\@tempdima sp}}%
802 \kern-\number\@tempdima sp}}%
803 \fi
804 \let\FV@BeginListFrame\relax
805 \let\FV@LeftListFrame\FV@LeftListFrame@Single
806 \let\FV@RightListFrame\relax
807 \let\FV@endListFrame\relax}

```

\FV@BeginListFrame@Single

```

808 \def\FV@BeginListFrame@Single{%
809 \lineskip\z@
810 \baselineskip\z@
811 \ifx\FancyVerbFillColor\relax
812 \let\FV@FrameFillLine\relax
813 \else
814 \@tempdima\FV@FrameRule\relax
815 \multiply\@tempdima-\tw@
816 \edef\FV@FrameFillLine{%
817 {\noexpand\FancyVerbFillColor{\vrule\@width\number\@tempdima sp}}%
818 \kern-\number\@tempdima sp}}%
819 \fi
820 %% DG/SR modification begin - May. 19, 1998
821 %% \FV@SingleFrameLine
822 \FV@SingleFrameLine{\z@}%
823 %% DG/SR modification end
824 \penalty\@M
825 \FV@SingleFrameSep
826 \penalty\@M}

```

\FV@Label

```

827 %% DG/SR modification begin - May. 19, 1998
828 \define@key{FV}{label}{%
829 \def\@tempa{#1}%
830 \ifx\@tempa\FV@None
831 \let\FV@LabelBegin\relax
832 \let\FV@LabelEnd\relax
833 \else
834 \FV@Label@i#1\@nil%
835 \fi}

```

\FV@Label@i

```

836 \def\FV@Label@i{\@ifnextchar[{\FV@Label@ii}{\FV@Label@ii []}]

```

\FV@Label@ii

```
837 \def\FV@Label@ii[#1]#2\@nil{%
838   \def\@tempa{#1}%
839   \ifx\@tempa\empty
840     \def\FV@LabelBegin{#2}%
841   \else
842     \def\FV@LabelBegin{#1}%
843     \def\FV@LabelPositionBottomLine{\@ne}%
844   \fi
845   \def\FV@LabelEnd{#2}}

846 \fvset{label=none}
```

\FV@LabelPosition

```
847 \define@key{FV}{labelposition}[none]{%
848   \@ifundefined{FV@LabelPosition@#1}%
849   {\FV@Error{Label position '#1' not defined.}\FV@eha}%
850   {\@nameuse{FV@LabelPosition@#1}}}
```

\FV@LabelPosition@none

```
851 \def\FV@LabelPosition@none{%
852   \let\FV@LabelPositionTopLine\relax%
853   \let\FV@LabelPositionBottomLine\relax}
```

\FV@LabelPosition@topline

```
854 \def\FV@LabelPosition@topline{%
855   \def\FV@LabelPositionTopLine{\@ne}%
856   \let\FV@LabelPositionBottomLine\relax}
```

\FV@LabelPosition@bottomline

```
857 \def\FV@LabelPosition@bottomline{%
858   \let\FV@LabelPositionTopLine\relax%
859   \def\FV@LabelPositionBottomLine{\@ne}}
```

\FV@LabelPosition@all

```
860 \def\FV@LabelPosition@all{%
861   \def\FV@LabelPositionTopLine{\@ne}%
862   \def\FV@LabelPositionBottomLine{\@ne}}

863 \fvset{labelposition=topline}
864 %% DG/SR modification end
```

\FV@SingleFrameLine

```
865 %% DG/SR modification begin - May. 19, 1998
866 %% \def\FV@SingleFrameLine{%
867 \def\FV@SingleFrameLine#1{%
868 %% DG/SR modification end
869   \hbox to\z@{%
870     \kern\leftmargin
```

```

871%% DG/SR modification begin - Jun. 22, 1998
872  \ifnum#1=\z@
873    \let\FV@Label\FV@LabelBegin
874  \else
875    \let\FV@Label\FV@LabelEnd
876  \fi
877  \ifx\FV@Label\relax
878%% DG/SR modification end
879    \FancyVerbRuleColor{\vrule \@width\linewidth \@height\FV@FrameRule}%
880%% DG/SR modification begin - Jun. 22, 1998
881  \else
882    \ifnum#1=\z@
883      \setbox\z@\hbox{\strut\enspace\FV@LabelBegin\enspace\strut}%
884    \else
885      \setbox\z@\hbox{\strut\enspace\FV@LabelEnd\enspace\strut}%
886    \fi
887    \@tempdimb=\dp\z@
888    \advance\@tempdimb -.5\ht\z@
889    \@tempdimc=\linewidth
890    \advance\@tempdimc -\wd\z@
891    \divide\@tempdimc\tw@
892    \ifnum#1=\z@ % Top line
893      \ifx\FV@LabelPositionTopLine\relax
894        \FancyVerbRuleColor{\vrule \@width\linewidth \@height\FV@FrameRule}%
895      \else
896        \FV@FrameLineWithLabel
897      \fi
898    \else % Bottom line
899      \ifx\FV@LabelPositionBottomLine\relax
900        \FancyVerbRuleColor{\vrule \@width\linewidth \@height\FV@FrameRule}%
901      \else
902        \FV@FrameLineWithLabel
903      \fi
904    \fi
905  \fi
906%% DG/SR modification end
907  \hss}}

```

\FV@FrameLineWithLabel

```

908%% DG/SR modification begin - May. 19, 1998
909\def\FV@FrameLineWithLabel{%
910  \ht\z@\@tempdimb\dp\z@\@tempdimb%
911  \FancyVerbRuleColor{%
912    \vrule \@width\@tempdimc \@height\FV@FrameRule
913    \raise\@tempdimb\box\z@
914    \vrule \@width\@tempdimc \@height\FV@FrameRule}}
915%% DG/SR modification end

```

\FV@BeginListFrame@Lines

```

916 \def\FV@BeginListFrame@Lines{%
917   \begingroup
918     \lineskip\z@skip
919 %% DG modification begin - June 18, 1997 (effect of \baselineskip too earlier)
920 %%   \baselineskip\z@skip
921 %%   \FV@SingleFrameLine
922 %% DG/SR modification begin - May. 19, 1998
923 %%   \FV@SingleFrameLine
924   \FV@SingleFrameLine{\z@}%
925 %% DG/SR modification end
926   \kern-0.5\baselineskip\relax
927   \baselineskip\z@skip
928 %% DG modification end
929   \kern\FV@FrameSep\relax
930 \endgroup}%

```

\FV@EndListFrame@Lines

```

931 \def\FV@EndListFrame@Lines{%
932   \begingroup
933     \baselineskip\z@skip
934     \kern\FV@FrameSep\relax
935 %% DG/SR modification begin - May. 19, 1998
936 %%   \FV@SingleFrameLine
937   \FV@SingleFrameLine{\@ne}%
938 %% DG/SR modification end
939 \endgroup}

```

\FV@SingleFrameSep

```

940 \def\FV@SingleFrameSep{%
941   \hbox to \z@{%
942     \kern\leftmargin
943     \hbox to\linewidth{%
944       \FancyVerbRuleColor{%
945 %% DG modification begin - June 18, 1997 (\FV@FrameSep missing)
946         \ifx\FancyVerbFillColor\relax
947           \vrule\@width Opt\@height\FV@FrameSep\relax
948         \fi
949 %% DG modification end
950         \vrule\@width\FV@FrameRule\relax
951         \ifx\FancyVerbFillColor\relax
952           \hfil
953         \else
954           {\FancyVerbFillColor\leaders\hrule\@height\FV@FrameSep\hfil}%
955         \fi
956 %% DG modification begin - June 18, 1997 (\FV@FrameSep missing)
957         \ifx\FancyVerbFillColor\relax
958           \vrule\@width Opt\@height\FV@FrameSep\relax
959         \fi
960 %% DG modification end

```

```

961      \vrule\@width\FV@FrameRule\relax}}%
962      \hss}}

```

\FV@LeftListFrame@Single

```

963 \def\FV@LeftListFrame@Single{%
964   \strut
965   {\FancyVerbRuleColor{\vrule \@width\FV@FrameRule}}}%
966   \FV@FrameFillLine
967 %% DG modification begin - June 18, 1997 (to fill color on left side)
968 %%   \kern\FV@FrameSep}
969   \ifx\FancyVerbFillColor\relax
970     \kern\FV@FrameSep
971   \else
972     {\noexpand\leavevmode\FancyVerbFillColor{\vrule\@width\FV@FrameSep}}}%
973   \fi}
974 %% DG modification end

```

\FV@RightListFrame@Single

```

975 \def\FV@RightListFrame@Single{%
976 %% DG modification begin - June 18, 1997 (to fill color on right side)
977 %%   \kern\FV@FrameSep
978   \ifx\FancyVerbFillColor\relax
979     \kern\FV@FrameSep
980   \else
981     {\noexpand\leavevmode\FancyVerbFillColor{\vrule\@width\FV@FrameSep}}}%
982   \fi
983   {\noexpand\leavevmode\FancyVerbRuleColor{\vrule\@width\FV@FrameRule}}}%
984 %% DG modification end

```

\FV@endListFrame@Single

```

985 \def\FV@endListFrame@Single{%
986   \penalty\@M
987   \FV@SingleFrameSep
988   \penalty\@M
989 %% DG/SR modification begin - May. 19, 1998
990 %%   \FV@SingleFrameLine}
991   \FV@SingleFrameLine{\@ne}}
992 %% DG/SR modification end

```

\FV@FrameRule

```

993 \define@key{FV}{framerule}{%
994   \@tempdima=#1\relax
995   \edef\FV@FrameRule{\number\@tempdima sp\relax}}

```

\KV@FV@framerule@default

```

996 \def\KV@FV@framerule@default{\let\FV@FrameRule\fbboxrule}

```

```

\FV@FrameSep
997 \define@key{FV}{framesep}{%
998   \@tempdima=#1\relax
999   \edef\FV@FrameSep{\number\@tempdima sp\relax}}

\KV@FV@framesep@default
1000 \def\KV@FV@framesep@default{\let\FV@FrameSep\fbboxsep}

1001 \fvset{framerule,framesep}

\FancyVerbRuleColor
1002 \define@key{FV}{rulecolor}{%
1003   \def\@tempa{#1}%
1004   \ifx\@tempa\FV@None
1005     \let\FancyVerbRuleColor\relax
1006   \else
1007     \let\FancyVerbRuleColor\@tempa
1008   \fi}

\FancyVerbFillColor
1009 \define@key{FV}{fillcolor}{%
1010   \def\@tempa{#1}%
1011   \ifx\@tempa\FV@None
1012     \let\FancyVerbFillColor\relax
1013   \else
1014     \let\FancyVerbFillColor\@tempa
1015   \fi}

1016 \fvset{rulecolor=none,fillcolor=none}

\FV@Frame@double
1017 \def\FV@Frame@double{%
1018   \let\FV@FrameBegin\FV@FrameBegin@double
1019   \let\FV@FrameLine\FV@FrameLine@double
1020   \let\FV@FrameEnd\FV@FrameEnd@double}

1021 \define@key{FV}{frame}[none]{%
1022   \@ifundefined{FV@Frame@#1}{%
1023     {\FV@Error{Frame style ‘#1’ not defined.}\FV@eha}%
1024     {\@nameuse{FV@Frame@#1}}}}

1025 \fvset{frame=none}

Code line numbers:
1026 \newcounter{FancyVerbLine}

\FV@SetLineNo
1027 \define@key{FV}{firstnumber}[auto]{%
1028   \def\@tempa{#1}\def\@tempb{auto}%
1029   \ifx\@tempa\@tempb

```

```

1030 \def\FV@SetLineNo{%
1031 \c@FancyVerbLine\FV@CodeLineNo
1032 \advance\c@FancyVerbLine\m@ne}%
1033 \else
1034 \def\@tempb{last}%
1035 \ifx\@tempa\@tempb
1036 \let\FV@SetLineNo\relax
1037 \else
1038 %% DG/SR modification begin - Jan. 19, 1998
1039 %% \def\FV@SetLineNo{\c@FancyVerbLine#1}%
1040 \def\FV@SetLineNo{%
1041 \c@FancyVerbLine#1
1042 \advance\c@FancyVerbLine\m@ne}%
1043 %% DG/SR modification end
1044 \fi
1045 \fi}

```

\if@FV@NumberBlankLines

```

1046 \define@booleankey{FV}{numberblanklines}%
1047 {\let\if@FV@NumberBlankLines\iftrue}
1048 {\let\if@FV@NumberBlankLines\iffalse}
1049 \fvset{numberblanklines=true}

```

\FV@refstepcounter

```

1050 %% DG/SR modification begin - May. 20, 1998
1051 %% \def\refstepcounter#1{% Adapted from latex.ltx
1052 \def\FV@refstepcounter#1{%
1053 %% DG/SR modification end
1054 \stepcounter{#1}%
1055 \protected@edef\@currentlabel
1056 {\csname p@#1\endcsname\arabic{FancyVerbLine}}

```

\FV@StepLineNo

```

1057 \def\FV@StepLineNo{%
1058 \FV@SetLineNo%
1059 \def\FV@StepLineNo{%
1060 \if@FV@NumberBlankLines%
1061 \FV@refstepcounter{FancyVerbLine}%
1062 \else%
1063 \ifx\FV@Line\empty%
1064 \else%
1065 \FV@refstepcounter{FancyVerbLine}%
1066 \fi%
1067 \fi}%
1068 \FV@StepLineNo}

```

`\theFancyVerbLine`

```
1069 %% DG/SR modification begin - 1995
1070 %% \def\theFancyVerbLine{\rm\tiny\arabic{FancyVerbLine}}
1071 \def\theFancyVerbLine{\rmfamily\tiny\arabic{FancyVerbLine}}
1072 %% DG/SR modification end
```

```
1073 \define@key{FV}{numbers}[none]{%
1074   \@ifundefined{FV@Numbers@#1}%
1075     {\FV@Error{Numbers style ‘#1’ not defined.}\FV@eha}%
1076     {\@nameuse{FV@Numbers@#1}}}
```

(D.G. – Dec. 20, 1995 and Jan. 28, 1998):

Add-on to allow a step when printing the lines counter (“stepnumber” keyword)

Add-on to allow the counter to be printed on right side (numbers=right)

`\FV@Numbers@none`

```
1077 %% DG modification begin - Dec. 20, 1995 and Jan. 28, 1998
1078 %% \def\FV@Numbers@none{\let\FV@LeftListNumber\relax}
1079 \def\FV@Numbers@none{%
1080 \let\FV@LeftListNumber\relax
1081 \let\FV@RightListNumber\relax}
```

```
1082 \newcount\FV@StepNumber
1083 \define@key{FV}{stepnumber}{\FV@StepNumber#1}
```

`\KV@FV@stepnumber@default`

```
1084 \def\KV@FV@stepnumber@default{\FV@StepNumber\@ne}
1085 \fvset{stepnumber}
```

`\FV@Numbers@left`

```
1086 %% DG modification begin - Dec. 20, 1995
1087 %% \def\FV@Numbers@left{%
1088 %%   \def\FV@LeftListNumber{\hbox to\z@{%
1089 %%     \hss\theFancyVerbLine\kern\FV@NumberSep}}}
1090 \def\FV@Numbers@left{%
1091 %% DG/SR modification begin - Apr. 28, 1998
1092   \let\FV@RightListNumber\relax
1093 %% DG/SR modification end
1094   \def\FV@LeftListNumber{%
1095     \@tempcnta=\FV@CodeLineNo
1096     \@tempcntb=\FV@CodeLineNo
1097     \divide\@tempcntb\FV@StepNumber
1098     \multiply\@tempcntb\FV@StepNumber
1099     \ifnum\@tempcnta=\@tempcntb
1100 %% DG/SR modification begin - Apr. 28, 1998
1101 %%   \hbox to\z@{\hss\theFancyVerbLine\kern\FV@NumberSep}%
1102     \if\FV@NumberBlankLines
1103       \hbox to\z@{\hss\theFancyVerbLine\kern\FV@NumberSep}%
1104     \else
```

```

1105     \ifx\FV@Line\empty
1106     \else
1107         \hbox to\z@{\hss\theFancyVerbLine\kern\FV@NumberSep}%
1108     \fi
1109     \fi
1110 %% DG/SR modification end
1111 \fi}}

```

\FV@Numbers@right

```

1112 \def\FV@Numbers@right{%
1113 %% DG/SR modification begin - Apr. 28, 1998
1114 \let\FV@LeftListNumber\relax
1115 %% DG/SR modification end
1116 \def\FV@RightListNumber{%
1117 \@tempcnta=\FV@CodeLineNo
1118 \@tempcntb=\FV@CodeLineNo
1119 \divide\@tempcntb\FV@StepNumber
1120 \multiply\@tempcntb\FV@StepNumber
1121 \ifnum\@tempcnta=\@tempcntb
1122 %% DG/SR modification begin - Apr. 28, 1998
1123 %% \hbox to\z@{\kern\FV@NumberSep\theFancyVerbLine\hss}%
1124 \if\FV@NumberBlankLines
1125 \hbox to\z@{\kern\FV@NumberSep\theFancyVerbLine\hss}%
1126 \else
1127 \ifx\FV@Line\empty
1128 \else
1129 \hbox to\z@{\kern\FV@NumberSep\theFancyVerbLine\hss}%
1130 \fi
1131 \fi
1132 %% DG/SR modification end
1133 \fi}}
1134 %% DG modification end

```

\FV@NumberSep

```

1135 \define@key{FV}{numbersep}{%
1136 \@tempdima=#1\relax
1137 \edef\FV@NumberSep{\number\@tempdima sp\relax}}
1138 \fvset{numbers=none,numbersep=12pt,firstnumber=auto}

```

## 12.14 BVerbatim

BVerbatim

```

1139 \def\FV@BVerbatimBegin{%
1140 \begingroup
1141 \FV@UseKeyValues
1142 \FV@BeginVBox
1143 \let\FV@ProcessLine\FV@BProcessLine

```

```

1144 \FV@FormattingPrep
1145 \FV@ObeyTabsInit}%
1146 \def\FV@BVerbatimEnd{\FV@EndVBox\endgroup}

\FV@BeginVBox
1147 \def\FV@BeginVBox{%
1148 \leavevmode
1149 \hbox\ifx\FV@boxwidth\relax\else to\FV@boxwidth\fi\bgroup
1150 \ifcase\FV@baseline\vbox\or\vtop\or$\vcenter\fi\bgroup}

\FV@EndVBox
1151 \def\FV@EndVBox{\egroup\ifmmode$\fi\hfil\egroup}

\FV@boxwidth
1152 \define@key{FV}{boxwidth}{%
1153 \def\@tempa{#1}\def\@tempb{auto}%
1154 \ifx\@tempa\@tempb
1155 \let\FV@boxwidth\relax
1156 \else
1157 \@tempdima=#1\relax
1158 \edef\FV@boxwidth{\number\@tempdima sp}%
1159 \fi}

\KV@FV@boxwidth@default
1160 \def\KV@FV@boxwidth@default{\let\FV@boxwidth\relax}

\FV@baseline
1161 \define@key{FV}{baseline}{%
1162 \if t#1\@empty\let\FV@baseline\@ne\else
1163 \if c#1\@empty\let\FV@baseline\tw@\else\let\FV@baseline\z@\fi
1164 \fi}

1165 \fvset{baseline=b,boxwidth}

\FV@BProcessLine
1166 \def\FV@BProcessLine#1{\hbox{\FancyVerbFormatLine{#1}}}

\FV@BVerbatim
1167 \def\FV@BVerbatim{\FV@BVerbatimBegin\FV@Scan}

\FV@EVerbatim
1168 \def\FV@EVerbatim{\FV@BVerbatimEnd}

1169 \DefineVerbatimEnvironment{BVerbatim}{BVerbatim}{}

\FV@BUseVerbatim
1170 \def\FV@BUseVerbatim#1{\FV@BVerbatimBegin#1\FV@BVerbatimEnd}

```

\BVerbatimInput

```
1171 \def\BVerbatimInput{\FV@Command}{\BVerbatimInput}}
```

\FVC@BVerbatimInput

```
1172 \def\FVC@BVerbatimInput#1{\FV@BUseVerbatim{\FV@Input{#1}}}
```

## 12.15 Save verbatim

SaveVerbatim

```
1173 \def\SaveVerbatim{\FV@Environment}{\SaveVerbatim}}
```

\FVB@SaveVerbatim

```
1174 \def\FVB@SaveVerbatim#1{%
1175   \@bsphack
1176   \begingroup
1177     \FV@UseKeyValues
1178   %%   \FV@BeginVBox
1179   %%   \let\FV@ProcessLine\FV@BProcessLine
1180   %%   \FV@FormattingPrep
1181   %%   \FV@ObeyTabsInit%
1182   %%
1183   \def\SaveVerbatim@Name{#1}%
1184   \gdef\FV@TheVerbatim{}%
1185   \def\FV@ProcessLine##1{%
1186     \expandafter\gdef\expandafter\FV@TheVerbatim\expandafter{%
1187       \FV@TheVerbatim\FV@ProcessLine{##1}}}%
1188   \gdef\FV@TheVerbatim{}%
1189   \FV@Scan}
```

\FVE@SaveVerbatim

```
1190 \def\FVE@SaveVerbatim{%
1191   \expandafter\global\expandafter\let
1192   \csname FV@SV@\SaveVerbatim@Name\endcsname\FV@TheVerbatim
1193   %% \expandafter\gdef
1194   %% \csname FV@SV@\SaveVerbatim@Name\endcsname{\FV@TheVerbatim}
1195   %% \FV@endVBox
1196   %% \endgroup}
1197   \endgroup\@esphack}
```

SaveVerbatim

```
1198 \DefineVerbatimEnvironment{SaveVerbatim}{SaveVerbatim}{}
```

\FV@CheckIfSaved

```
1199 \def\FV@CheckIfSaved#1#2{%
1200   \@ifundefined{FV@SV@#1}%
1201   {\FV@Error{No verbatim text has been saved under name ‘#1’}\FV@eha}%
1202   {#2{\csname FV@SV@#1\endcsname}}}
```

```

\UseVerbatim
1203 \def\UseVerbatim{\FV@Command}{\UseVerbatim}}

\FVC@UseVerbatim
1204 \def\FVC@UseVerbatim#1{\FV@CheckIfSaved{#1}{\FV@UseVerbatim}}

\LUseVerbatim
1205 \def\LUseVerbatim{\FV@Command}{\LUseVerbatim}}

\FVC@LUseVerbatim
1206 \def\FVC@LUseVerbatim#1{\FV@CheckIfSaved{#1}{\FV@LUseVerbatim}}

\BUseVerbatim
1207 \def\BUseVerbatim{\FV@Command}{\BUseVerbatim}}

\FVC@BUseVerbatim
1208 \def\FVC@BUseVerbatim#1{\FV@CheckIfSaved{#1}{\FV@BUseVerbatim}}

```

## 12.16 Verbatim out

```

1209 \newwrite\FV@OutFile

\VerbatimOut
1210 \def\VerbatimOut{\FV@Environment}{\VerbatimOut}}

\FVB@VerbatimOut
1211 \def\FVB@VerbatimOut#1{%
1212   \@bsphack
1213   \begingroup
1214     \FV@UseKeyValues
1215     \FV@DefineWhiteSpace
1216     \def\FV@Space{\space}%
1217     \FV@DefineTabOut
1218     \def\FV@ProcessLine{\immediate\write\FV@OutFile}%
1219     \immediate\openout\FV@OutFile #1\relax
1220     \let\FV@FontScanPrep\relax
1221 %% DG/SR modification begin - May. 18, 1998 (to avoid problems with ligatures)
1222     \let\@noligs\relax
1223 %% DG/SR modification end
1224     \FV@Scan}

\FVE@VerbatimOut
1225 \def\FVE@VerbatimOut{\immediate\closeout\FV@OutFile\endgroup\@esphack}

VerbatimOut
1226 \DefineVerbatimEnvironment{VerbatimOut}{VerbatimOut}{\}

```

\FV@DefineTabOut

```
1227 \def\FV@DefineTabOut{%
1228   \def\FV@Tab{}%
1229   \@tempcnta=\FancyVerbTabSize\relax
1230   \loop\ifnum\@tempcnta>\z@
1231     \edef\FV@Tab{\FV@Tab\space}%
1232     \advance\@tempcnta\m@ne
1233   \repeat}
```

## 12.17 Short verbatim

\SaveVerb

Note “\outer\def^^M{”.

This is so that verbatim commands report an error when encountering an end-of-line, rather than scanning to the end of the file each time there is a missing verbatim delimiter.

If scanning fails (and thus  $\TeX$  ignores \FV@GetVerb), #1 is defined to be empty, a group is ended, but \FancyVerbAfterSave is not invoked.

\FV@Command

```
1234 \def\SaveVerb{\FV@Command}{\SaveVerb}}
```

\FVC@SaveVerb

```
1235 \begingroup
1236 \catcode'\^^M=\active%
1237 \gdef\FVC@SaveVerb#1#2{%
1238   \@namedef{FV@SV@#1}{}%
1239   \begingroup%
1240     \FV@UseKeyValues%
1241     \FV@CatCodes%
1242     \outer\def^^M{\FV@EOL}%
1243     \global\let\@tempg\FancyVerbAfterSave%
1244     \catcode'#2=12%
1245     \def\@tempa{\def\FancyVerbGetVerb####1###2}%
1246     \expandafter\@tempa\string#2{\endgroup\@namedef{FV@SV@#1}{##2}\@tempg}%
1247     \FancyVerbGetVerb\FV@EOL}%
1248 \endgroup
```

\FV@EOL

```
1249 \def\FV@EOL{%
1250   \endgroup
1251   \FV@Error%
1252   {Could not find the end delimiter of a short verb command}%
1253   {You probably just forget the end delimiter of a \string\Verb\space or
1254     \string\SaveVerb^^J%
1255     command, or you broke the literal text across input lines.^^J%
1256     Hit <return> to procede.}}
```

```

\FancyVerbAfterSave
1257 \define@key{FV}{aftersave}{\def\FancyVerbAfterSave{#1}}
1258 \fvset{aftersave=}

\FV@UseVerb
1259 \def\FV@UseVerb#1{\mbox{\FV@UseKeyValues\FV@FormattingPrep#1}}

\UseVerb
1260 \def\UseVerb{\FV@Command}{\UseVerb}}

\FVC@UseVerb
1261 \def\FVC@UseVerb#1{%
1262   \@ifundefined{FV@SV@#1}%
1263   {\FV@Error{Short verbatim text never saved to name ‘#1’}\FV@eha}%
1264   {\FV@UseVerb{\@nameuse{FV@SV@#1}}}}

\Verb
1265 \def\Verb{\FV@Command}{\Verb}}

\FVC@Verb
1266 \begingroup
1267 \catcode'\^^M=\active%
1268 \gdef\FVC@Verb#1{%
1269   \begingroup%
1270     \FV@UseKeyValues%
1271     \FV@FormattingPrep%
1272     \FV@CatCodes%
1273     \outer\def^^M{}%
1274     \catcode'#1=12%
1275     \def\@tempa{\def\FancyVerbGetVerb####1###2}%
1276     \expandafter\@tempa\string#1{\mbox{##2}\endgroup}%
1277     \FancyVerbGetVerb\FV@EOL}%
1278 \endgroup

\DefineShortVerb
1279 \def\DefineShortVerb{\FV@Command}{\DefineShortVerb}}

\FVC@DefineShortVerb
1280 \def\FVC@DefineShortVerb#1{%
1281   \@ifundefined{FV@CC@\string#1}%
1282   {\FVC@@DefineShortVerb#1}%
1283   {\FV@Error{\‘expandafter\@gobble\string#1’ is already a short
1284     verb character.}\FV@eha}}

\FVC@@DefineShortVerb
1285 \def\FVC@@DefineShortVerb#1{%
1286   \begingroup
1287     \lccode'\~=#1%

```

```

1288 \lowercase{\gdef\@tempg{\edef~}\global\let\@temp~}%
1289 \endgroup
1290 \expandafter\let\csname FV@AC@\string#1\endcsname\@temp~
1291 \expandafter\edef\csname FV@CC@\string#1\endcsname{\the\catcode'#1}%
1292 \expandafter\let\csname FV@KV@\string#1\endcsname\FV@KeyValues
1293 \@tempg{%
1294 \let\noexpand\FV@KeyValues\expandafter\noexpand
1295 \csname FV@KV@\string#1\endcsname
1296 \noexpand\FV@Verb\expandafter\@gobble\string#1}%
1297 \expandafter\def\expandafter\dospecials\expandafter{\dospecials\do#1}%
1298 \expandafter\def\expandafter\@sanitize\expandafter{\@sanitize\@makeother#1}%
1299 \catcode'#1=active}%

```

`\UndefineShortVerb`

```

1300 \def\UndefineShortVerb#1{%
1301 \@ifundefined{FV@CC@\string#1}%
1302 {\FV@Error{'\expandafter\@gobble\string#1' is not a short
1303 verb character}\FV@eha}%
1304 {\FV@UndefineShortVerb#1}}

```

`\FV@UndefineShortVerb`

```

1305 \def\FV@UndefineShortVerb#1{%
1306 \catcode'#1=\csname FV@CC@\string#1\endcsname
1307 %% DG/SR modification begin - Jun. 12, 1998
1308 \expandafter\let\csname FV@CC@\string#1\endcsname\relax
1309 %% DG/SR modification end
1310 \begingroup
1311 \lccode'\~='#1%
1312 \lowercase{\gdef\@tempg{\let~}}%
1313 \endgroup
1314 \expandafter\@tempg\csname FV@AC@\string#1\endcsname
1315 \def\@tempa##1\do#1##2\@nil##3\@nil##4\@nil{##3\def\dospecials{##1##2}\fi}%
1316 \expandafter\@tempa\dospecials\@nil\iftrue\@nil\do#1\@nil\iffalse\@nil\@nil
1317 \def\@tempa##1\@makeother#1##2\@nil##3\@nil##4\@nil{%
1318 ##3\def\@sanitize{##1##2}\fi}%
1319 \expandafter\@tempa\@sanitize\@nil\iftrue\@nil\do#1\@nil\iffalse\@nil\@nil}

```

Moving verbatim. Need to worry about using separate identifier for this class of verbatim, and

`\SaveMVerb`

```

1320 \def\SaveMVerb{\FV@Command}{\SaveMVerb}}

```

`\FV@SaveMVerb`

```

1321 \begingroup
1322 \catcode'\^^M=active%
1323 \gdef\FV@SaveMVerb#1#2{%
1324 \@ifundefined{FV@SVM@#1}{}%
1325 {\FV@Error{Moving verbatim name '#1' already used}%

```

```

1326      {I will overwrite the old definition. Hit <return> to continue.}}%
1327 \global\@namedef{FV@SVM@#1}{}%
1328 \begingroup%
1329   \let\FV@SavedKeyValues\FV@KeyValues%
1330   \FV@UseKeyValues%
1331   \FV@CatCodes%
1332   \outer\def^^M{}%
1333   \global\let\@tempg\FancyVerbAfterSave%
1334   \catcode'#2=12%
1335   \def\@tempa{\def\FancyVerbGetVerb####1###2}%
1336   \expandafter\@tempa\string#2{%
1337     \if@filesw
1338       \FV@DefineWhiteSpace%
1339       \let\FV@Space\space%
1340       \let\FV@Tab\space%
1341       \FV@MakeUnActive%
1342       \let\protect\string
1343       \immediate\write\@auxout{%
1344         \noexpand\SaveGVerb[\FV@SavedKeyValues]{#1}\string#2##2\string#2}%
1345       \fi
1346     \endgroup%
1347     \@namedef{FV@SV@#1}{##2}%
1348     \@tempg}%
1349   \FancyVerbGetVerb\FV@EOL}%
1350 \endgroup

```

\SaveGVerb

```
1351 \def\SaveGVerb{\FV@Command}{\SaveGVerb}
```

\FVC@SaveGVerb

```

1352 \begingroup
1353 \catcode'\^^M=\active%
1354 \gdef\FVC@SaveGVerb#1#2{%
1355   \global\@namedef{FV@SVG@#1}{}%
1356   \begingroup%
1357   \FV@UseKeyValues%
1358   \FV@CatCodes%
1359   \outer\def^^M{}%
1360   \catcode'#2=12%
1361   \def\@tempa{\def\FancyVerbGetVerb####1###2}%
1362   \expandafter\@tempa\string#2{\endgroup\global\@namedef{FV@SVG@#1}{##2}}%
1363   \FancyVerbGetVerb\FV@EOL}%
1364 \endgroup

```

\UseMVerb

```
1365 \def\UseMVerb{\protect\pUseMVerb}
```

\pUseMVerb

```
1366 \def\pUseMVerb{\FV@Command}{\pUseMVerb}
```

\FVC@pUseMVerb

```
1367 \def\FVC@pUseMVerb#1{%
1368   \expandafter\ifx \csname FV@SVM@#1\endcsname\relax
1369   \expandafter\ifx \csname FV@SVG@#1\endcsname\relax
1370     \@warning{Moving verbatim text not defined for name '#1'}\FV@eha
1371     {\bf ??}%
1372   \else
1373     \FV@UseVerb{\@nameuse{FV@SVG@#1}}%
1374   \fi
1375 \else
1376   \FV@UseVerb{\@nameuse{FV@SVM@#1}}%
1377 \fi}

1378 \expandafter\ifx\csname documentclass\endcsname\relax
```

lrbox

```
1379 \def\lrbox#1{%
1380   \edef\@tempa{%
1381     \endgroup
1382     \setbox#1\hbox{%
1383       \begingroup\aftergroup}%
1384     \def\noexpand\@currentvir{\@currentvir}}%
1385     %\def\noexpand\@currentvline{\@on@line}}%
1386   \@tempa
1387   \@endpefalse
1388   \bgroup
1389   \ignorespaces}
1390 \def\endlrbox{\unskip\egroup}

1391 \fi

1392 %% DG/SR modification begin - Mar 21 2000
1393 %%\@input{fancyvrb.rc}
1394 \InputIfFileExists{fancyvrb.cfg}{-}{-}
1395 %% DG/SR modification end
```

Unused code - don't take care! (DG/SR)  
Error messages to void:  
</fancyvrb>

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in **roman** refer to the code lines where the entry is used.

Symbols		
\!	..... 295	\@CustomVerbatimCommand .. <u>87</u> , 97, 98
\%	..... 153	\@CustomVerbatimEnvironment ...
\=	..... 301, 312	..... <u>105</u> , <u>106</u> , <u>107</u>
		\@noligs ..... 375, 470, <u>1222</u>

\[	296	\FancyVerbRuleColor	879, 894, 900, 911, 944, 965, 983, <u>1002</u>
\{	142, 329, 334, 339, 344, 349, 354	\FancyVerbSpace	553, 555
\}	142	\FancyVerbStartNum	166, 172, 201, 210
\]	297	\FancyVerbStartString	167, 169, 173, 200, 218
\^	101, 103, 104, 131, 133, 362, 378, 435, 544, 1236, 1267, 1322, 1353	\FancyVerbStopNum	178, 184, 229, 238, 245
\‘	514, 516, 522	\FancyVerbStopString	179, 181, 185, 228, 253
\~	1287, 1311	\FancyVerbTab	566, <u>569</u>
\_	132, 543, 554, 555	\FancyVerbTabSize	557, <u>558</u> , 577, 580, 1229
<b>B</b>			
\BUseVerbatim	<u>1207</u>	\FV@@@Gobble	268, 279, <u>283</u>
BVerbatim (environment)	60	\FV@@@CheckEnd	332, 337, 347, 352, 383
\BVerbatimInput	<u>1171</u>	\FV@@@CheckEnd@i	332
<b>C</b>			
\CustomVerbatimCommand	<u>97</u>	\FV@@@CheckEnd@ii	347
\CustomVerbatimEnvironment	<u>105</u>	\FV@@@CheckEnd@iii	352
<b>D</b>			
\define@booleankey	<u>66</u>	\FV@@@CheckEnd@iv	337
\DefineShortVerb	<u>1279</u>	\FV@@@Gobble	267, <u>270</u>
\DefineVerbatimEnvironment	113, 751, 761, 1169, 1198, 1226	\FV@@@CheckEnd	331, 336, 346, 351
<b>E</b>			
\empty	839, 1063, 1105, 1127	\FV@@@CheckEnd@i	331
environments:		\FV@@@CheckEnd@ii	346
BVerbatim	60	\FV@@@CheckEnd@iii	351
lrbox	68	\FV@@@CheckEnd@iv	336
LVerbatim	50	\FV@@@Command	79, 80, <u>81</u>
SaveVerbatim	62	\FV@@@FindStartString	216, <u>220</u>
Verbatim	50	\FV@@@FindStopString	251, <u>255</u>
VerbatimOut	63	\FV@@@Gobble	<u>266</u> , 280
<b>F</b>			
\FancyVerbAfterSave	1243, <u>1257</u> , 1333	\FV@@@Input	420, <u>422</u>
\FancyVerbCodes	129, <u>137</u>	\FV@@@ObeyTabs	582, <u>584</u>
\FancyVerbDefineActive	451, <u>547</u>	\FV@@@ObeyTabsInit	<u>576</u> , 597
\FancyVerbFillColor	795, 801, 811, 817, 946, 951, 954, 957, 969, 972, 978, 981, <u>1009</u>	\FV@@@ObeyTabSize	581, 590
\FancyVerbFormatCom	452, <u>600</u>	\FV@@@PreProcessLine	<u>191</u> , 212, 230, 247, 263
\FancyVerbFormatLine	<u>605</u> , 698, 1166	\FV@@@ReadLine	439, <u>441</u>
\FancyVerbGetLine	374, 375, <u>377</u>	\FV@@@ActiveWhiteSpace	126, <u>130</u>
\FancyVerbGetVerb	1245, 1247, 1275, 1277, 1335, 1349, 1361, 1363	\FV@@@AddToHook	<u>135</u> , 152, 160
\FancyVerbHFuzz	<u>616</u> , 691	\FV@@@Auto	<u>75</u> , 475, 483, 495, 502
		\FV@@@BadBeginError	364, <u>367</u>
		\FV@@@BadCodes	<u>318</u> , 339, 354, 358
		\FV@@@BadEndError	383, <u>391</u>
		\FV@@@baseline	1150, <u>1161</u>
		\FV@@@BaseLineStretch	455, 463, <u>481</u> , 488
		\FV@@@BeginListFrame	638, 768, 773, 778, 783, 788, 794, 804
		\FV@@@BeginListFrame@Lines	778, 783, <u>916</u>
		\FV@@@BeginListFrame@Single	773, <u>808</u>
		\FV@@@BeginScanning	290, <u>361</u>

\FV@BeginVBox	1142, <u>1147</u> , 1178	\FV@Error	31, 95, 110, 277, 319, 369, 393, 405, 416, 561, 742, 849, 1023, 1075, 1201, 1251, 1263, 1283, 1302, 1325
\FV@boxwidth	1149, <u>1152</u> , 1160	\FV@FindStartNum	204, <u>209</u>
\FV@BProcessLine	1143, <u>1166</u> , 1179	\FV@FindStartStop	190, <u>195</u> , 204, 207, 230, 232, 235, 240, 258
\FV@BUseVerbatim	<u>1170</u> , 1172, 1208	\FV@FindStartString	207, <u>214</u>
\FV@BVerbatimBegin	1139, 1167, 1170	\FV@FindStopNum	232, <u>237</u>
\FV@BVerbatimEnd	1146, 1168, 1170	\FV@FindStopString	235, <u>249</u>
\FV@CatCodes	124, 287, 419, 635, 1241, 1272, 1331, 1358	\FV@FontFamily	458, 465, 491, 518, 519, 523, 527, 531, 532, 537, 538
\FV@CatCodesHook	128, <u>134</u> , 152, 160	\FV@fontfamily@cmtt	<u>521</u>
\FV@CheckEnd	326, 330, 335, 345, 350, 381	\FV@fontfamily@cmtt-spanish	<u>525</u>
\FV@CheckEnd@i	330	\FV@fontfamily@courier	<u>528</u>
\FV@CheckEnd@ii	345	\FV@fontfamily@helvetica	<u>534</u>
\FV@CheckEnd@iii	350	\FV@fontfamily@tt	<u>515</u>
\FV@CheckEnd@iv	335	\FV@FontScanPrep	127, 491, 516, 522, 526, 529, 535, 1220
\FV@CheckIfSaved	<u>1199</u> , 1204, 1206, 1208	\FV@FontSeries	466, <u>493</u>
\FV@CheckScan	374, 375, <u>402</u>	\FV@FontShape	467, <u>500</u>
\FV@CodeLineNo	187, 189, 210, 238, 245, 447, 1031, 1095, 1096, 1117, 1118	\FV@FontSize	457, 464, <u>473</u> , 480
\FV@Command	<u>77</u> , 96, 755, 765, 1171, 1203, 1205, 1207, <u>1234</u> , 1260, 1265, 1279, 1320, 1351, 1366	\FV@FormattingPrep	446, 636, 1144, 1180, 1259, 1271
\FV@CommandChars	145, 150, 152	\FV@Frame@bottomline	<u>787</u>
\FV@CommentChar	<u>153</u> , 160	\FV@Frame@double	<u>1017</u>
\FV@DefineCheckEnd	289, <u>327</u>	\FV@Frame@leftline	<u>792</u>
\FV@DefineCommandChars	147, <u>149</u>	\FV@Frame@lines	<u>777</u>
\FV@DefineFindStart	195, 198, <u>199</u>	\FV@Frame@none	<u>767</u>
\FV@DefineFindStop	202, 211, 223, <u>227</u>	\FV@Frame@single	<u>772</u>
\FV@DefinePreProcessLine	196	\FV@Frame@topline	<u>782</u>
\FV@DefineTabOut	1217, <u>1227</u>	\FV@FrameBegin	1018
\FV@DefineWhiteSpace	450, <u>545</u> , 1215, 1338	\FV@FrameBegin@double	1018
\FV@eha	35, 277, 416, 561, 849, 1023, 1075, 1201, 1263, 1284, 1303, 1370	\FV@FrameEnd	1020
\FV@EndList	<u>724</u> , 748, 758	\FV@FrameEnd@double	1020
\FV@EndListFrame	726, 771, 776, 781, 786, 791, 807	\FV@FrameFillLine	794, 796, 800, 812, 816, 966
\FV@EndListFrame@Lines	781, 791, <u>931</u>	\FV@FrameLine	1019
\FV@EndListFrame@Single	776, <u>985</u>	\FV@FrameLine@double	1019
\FV@EndScanning	384, <u>397</u> , 410	\FV@FrameLineWithLabel	896, 902, <u>908</u>
\FV@EndVBox	1146, <u>1151</u> , 1195	\FV@FrameRule	798, 814, 879, 894, 900, 912, 914, 950, 961, 965, 983, <u>993</u> , 996
\FV@Environment	<u>99</u> , 111, 112, 114, 116, 1173, 1210	\FV@FrameSep	929, 934, 945, 947, 954, 956, 958, 968, 970, 972, 977, 979, 981, <u>997</u> , 1000
\FV@EnvironName	292, 293, 371, 382, 395, 398, 399, 405	\FV@GetKeyValues	83, <u>85</u> , 103
\FV@EOF	402, 403, <u>404</u>	\FV@GetLine	365, <u>373</u> , 387
\FV@EOL	1242, 1247, <u>1249</u> , 1277, 1349, 1363	\FV@Gobble	193, <u>271</u> , 284

\FV@InFile	241, 242, 259, 260, 412, 414, 415, 417, 425, 429, 437, 438	\FV@NumberSep	1089, 1101, 1103, 1107, 1123, 1125, 1129, 1135
\FV@Input	413, 756, 766, 1172	\FV@ObeyTabs	582, 585, 605
\FV@InterLinePenalty	620, 633	\FV@ObeyTabsInit	596, 637, 1145, 1181
\FV@KeyValues	79, 80, 86, 100, 119, 121, 122, 1292, 1294, 1329	\FV@ObeyTabSize	578, 588
\FV@Label	827, 873, 875, 877	\FV@OutFile	1209, 1218, 1219, 1225
\FV@Label@i	834, 836	\FV@ParseStart	163, 164
\FV@Label@ii	836, 837	\FV@ParseStop	175, 176
\FV@LabelBegin	831, 840, 842, 873, 883	\FV@PreProcessLine	188, 387, 427, 431
\FV@LabelEnd	832, 845, 875, 885	\FV@ProcessLine	194, 634, 711, 714, 719, 731, 735, 741, 743, 1143, 1179, 1185, 1187, 1218
\FV@LabelPosition	847	\FV@ReadLine	424, 434, 444
\FV@LabelPosition@all	860	\FV@refstepcounter	1050, 1061, 1065
\FV@LabelPosition@bottomline	857	\FV@RightListFrame	700, 701, 770, 775, 780, 785, 790, 806
\FV@LabelPosition@none	851	\FV@RightListFrame@Single	775, 975
\FV@LabelPosition@topline	854	\FV@RightListNumber	702, 1081, 1092, 1116
\FV@LabelPositionBottomLine	843, 853, 856, 859, 862, 899	\FV@SavedKeyValues	1329, 1344
\FV@LabelPositionTopLine	852, 855, 858, 861, 893	\FV@Scan	286, 749, 759, 1167, 1189, 1224
\FV@LeaveVMode	627, 639	\FV@SetLineNo	1027, 1058
\FV@LeftListFrame	697, 769, 774, 779, 784, 789, 805	\FV@SetLineWidth	632, 682
\FV@LeftListFrame@Single	774, 805, 963	\FV@SetupFont	449, 454
\FV@LeftListNumber	696, 1078, 1080, 1088, 1094, 1114	\FV@SingleFrameLine	821, 822, 865, 921, 923, 924, 936, 937, 990, 991
\FV@Line	194, 217, 252, 268, 270, 386, 423, 426, 442, 443, 1063, 1105, 1127	\FV@SingleFrameSep	825, 940, 987
\FV@List	624, 747, 757	\FV@Space	545, 552, 1216, 1339
\FV@ListNesting	629, 646	\FV@StepLineNo	192, 1057
\FV@ListParameterHook	614, 630	\FV@StepNumber	1082–1084, 1097, 1098, 1119, 1120
\FV@ListProcessLastLine	725, 730	\FV@Tab	545, 557, 583, 1228, 1231, 1340
\FV@ListProcessLine	692, 710, 713, 718, 723	\FV@TabBox	575, 584, 589, 593–595
\FV@ListProcessLine@i	634, 705, 741	\FV@TabChar	557, 565, 595
\FV@ListProcessLine@ii	711, 712	\FV@TheVerbatim	1184, 1186–1188, 1192, 1194
\FV@ListProcessLine@iii	714, 715, 735	\FV@TrueTab	583, 586
\FV@ListProcessLine@iv	719, 720, 731	\FV@UndefineShortVerb	1304, 1305
\FV@ListVSpace	631, 662	\FV@UseKeyValues	118, 626, 1141, 1177, 1214, 1240, 1259, 1270, 1330, 1357
\FV@LUseVerbatim	762, 766, 1206	\FV@UseVerb	1259, 1264, 1373, 1376
\FV@LVerbatimBegin	757, 759, 763	\FV@UseVerbatim	752, 756, 1204
\FV@LVerbatimEnd	758, 760, 763	\FV@VerbatimBegin	747, 749, 753
\FV@MakeActive	507, 516, 522	\FV@VerbatimEnd	748, 750, 753
\FV@MakeUnActive	509, 510, 512, 1341	\FV@XLeftMargin	606, 686
\FV@None	74, 144, 155, 830, 1004, 1011	\FV@XRightMargin	608, 687
\FV@Numbers@left	1086	\FVB@BVerbatim	1167
\FV@Numbers@none	1077	\FVB@LVerbatim	759
\FV@Numbers@right	1112	\FVB@SaveVerbatim	1174

<code>\FVB@Verbatim</code> .....	<u>749</u>
<code>\FVB@VerbatimOut</code> .....	<u>1211</u>
<code>\FVC@@DefineShortVerb</code> .. 1282,	<u>1285</u>
<code>\FVC@BUseVerbatim</code> .....	<u>1208</u>
<code>\FVC@BVerbatimInput</code> .....	<u>1172</u>
<code>\FVC@DefineShortVerb</code> .....	<u>1280</u>
<code>\FVC@LUseVerbatim</code> .....	<u>1206</u>
<code>\FVC@LVerbatimInput</code> .....	<u>766</u>
<code>\FVC@pUseMVerb</code> .....	<u>1367</u>
<code>\FVC@SaveGVerb</code> .....	<u>1352</u>
<code>\FVC@SaveMVerb</code> .....	<u>1321</u>
<code>\FVC@SaveVerb</code> .....	<u>1235</u>
<code>\FVC@UseVerb</code> .....	<u>1261</u>
<code>\FVC@UseVerbatim</code> .....	<u>1204</u>
<code>\FVC@Verb</code> .....	<u>1266</u> , 1296
<code>\FVC@VerbatimInput</code> .....	<u>756</u>
<code>\FVE@BVerbatim</code> .....	<u>1168</u>
<code>\FVE@LVerbatim</code> .....	<u>760</u>
<code>\FVE@SaveVerbatim</code> .....	<u>1190</u>
<code>\FVE@Verbatim</code> .....	<u>750</u>
<code>\FVE@VerbatimOut</code> .....	<u>1225</u>
<code>\fvset</code> .....	<u>76</u>
<b>I</b>	
<code>\if@FV@NumberBlankLines</code> .....	<u>1046</u> , 1060, 1102, 1124
<code>\if@FV@ResetMargins</code> .....	<u>610</u>
<code>\InputIfFileExists</code> .....	1394
<b>K</b>	
<code>\KV@@booleankey</code> .....	71, <u>72</u>
<code>\KV@booleankey</code> .....	69, <u>70</u>
<code>\KV@do</code> .....	121
<code>\KV@FV@baselinestretch@default</code>	<u>488</u>
<code>\KV@FV@boxwidth@default</code> .....	<u>1160</u>
<code>\KV@FV@firstline@default</code> .....	<u>171</u>
<code>\KV@FV@fontsize@default</code> .....	<u>480</u>
<code>\KV@FV@framerule@default</code> .....	<u>996</u>
<code>\KV@FV@framesep@default</code> .....	<u>1000</u>
<code>\KV@FV@gobble@default</code> .....	<u>284</u>
<code>\KV@FV@lastline@default</code> .....	<u>183</u>
<code>\KV@FV@stepnumber@default</code> ....	<u>1084</u>
<code>\KV@prefix</code> .....	120
<b>L</b>	
<code>lrbox (environment)</code> .....	68
<code>\LUseVerbatim</code> .....	<u>1205</u>
<code>LVerbatim (environment)</code> .....	50
<code>\LVerbatimInput</code> .....	<u>765</u>
<b>P</b>	
<code>\pUseMVerb</code> .....	<u>1366</u>
<b>R</b>	
<code>\RecustomVerbatimCommand</code> .....	<u>98</u>
<code>\RecustomVerbatimEnvironment</code> ..	<u>106</u>
<b>S</b>	
<code>\SaveGVerb</code> .....	1344, <u>1351</u>
<code>\SaveMVerb</code> .....	<u>1320</u>
<code>\SaveVerb</code> .....	1234, 1254
<code>\SaveVerbatim</code> .....	1173
<code>SaveVerbatim (environment)</code> .....	62
<code>\SaveVerbatim@Name</code> 1183, 1192, 1194	
<b>T</b>	
<code>\theFancyVerbLine</code> .....	<u>1069</u>
<b>U</b>	
<code>\UndefineShortVerb</code> .....	<u>1300</u>
<code>\UseMVerb</code> .....	<u>1365</u>
<code>\UseVerb</code> .....	<u>1260</u>
<code>\UseVerbatim</code> .....	<u>1203</u>
<b>V</b>	
<code>\V@@@footnotetext</code> .....	62, <u>64</u>
<code>\V@@footnotetext</code> .....	46, <u>48</u>
<code>\V@footnote</code> .....	40, 43
<code>\V@footnotetext</code> .....	39, 42, <u>45</u>
<code>\Verb</code> .....	1253, <u>1265</u>
<code>Verbatim (environment)</code> .....	50
<code>\VerbatimEnvironment</code> .....	288, <u>291</u>
<code>\VerbatimFootnotes</code> .....	<u>36</u>
<code>\VerbatimInput</code> .....	<u>755</u>
<code>\VerbatimOut</code> .....	<u>1210</u>
<code>VerbatimOut (environment)</code> .....	63

## Change History

v2.0 Beta

General: First version personally

shown by Timothy VAN ZANDT. ... 1

v2.5		
General: First public release. . . . .	1	
v2.6		
General: . Three bug corrected, options numberblanklines, label, labelposition and leftline added and few precisions. . . . .	1	
Add “leftline” value for the “frame” parameter (idea taken from a requirement of Christophe Roudet on June 9, 1998). . . . .	51	
Add the “label” and “labelposition” parameters (suggested by Michael Friendly on May 15, 1998). . . . .	52	
Add the “numberblanklines” parameter to allow to don’t number empty lines (suggested by Philippe Esperet on April 25, 1998). . . . .	58	
Correct a bug for the – ligature in		T1 encoding (reported by Mario Hassler on May 12, 1998). . . . . 39
		Correct a bug to undefine the macro associated to short verbatim character (reported by Mikhail Kolodin on June 12, 1998). . . . . 66
		Correct a bug which didn’t allow to use the label macro in verbatim env. (reported by Philippe Esperet on May 20, 1998). . . . . 58
	v2.7	
	General: DG/SR changed how fancyvrb.cfg included . . . . .	1
	v2.7a	
	General: firstline bug . . . . .	1
	v2.8	
	General: fixed bug with several trailing spaces (hv) . . . . .	1