

proofread — Commands for inserting annotations*

Wybo Dekker[†]

Released 2019/04/25

Abstract

The `proofread` package defines a few \LaTeX commands that are useful when you proofread a latex document. These allow you to easily highlight text and add comments in the margin. Vim escape sequences are provided for inserting or removing these \LaTeX commands in the source. The package is based on code for a text highlighting command that was published by Antal Spector-Zabusky in [StackExchange](#).

Options are provided for displaying the document with extra line spacing, and for display of it in either uncorrected or corrected state, both without margin notes.

1 Usage

Note: this package is based on the `soul` package, so if you plan to highlight non-ASCII characters, you must compile your source with either `xetex`- or `luatex`-based compilers.

The commands described below display a highlighted phrase in your compiled document and place a comment in the margin, prefixed with a counter, which is indicated with `n` in the following. This counter is useful in the communication with the author of the document.

Previous versions of this package used the standard \LaTeX `\marginpar` to create notes in the margin. However, although these will not overwrite each other, they can not be used in tables, minipages, footnotes and more. Therefore, this version creates margin notes using the `marginnote` package, in order to make them work in tables, minipages, footnotes and more. This implies that margin notes are placed at the vertical position where they are called, instead of being automatically stacked. So if several notes are generated on the same line, they overwrite each other. You can prevent this by prefixing the second note on the line with `\skp` or `\skp[1]`, the third note with `\skp[2]` and so on. Multiline notes may need larger numbers.

The following supposes that you have installed vim-facilities as described in section 2 (“Installing the Vim commands”).

`\del` `\del{phrase}` displays `phrase` and places `n: delete` in the margin. In the vim editor, you can generate this code by selecting the phrase and typing `<escape>d`. After typing this escape sequence, you will be in normal mode, behind the closing brace.

`\yel` `\yel[comment]{phrase}` displays `phrase` and places `n: comment` in the margin. In the vim editor, you can generate this code by selecting the phrase and typing `<escape>y`. After typing this escape sequence, you will be in insert mode between the square bracket pair, ready to insert the *comment*.

`\add` `\add{phrase}` displays `phrase` and places `n: add` in the margin. In the vim editor, you can generate this code *after* the current cursor position by typing `<escape>a`, or *before*

*This file describes version v1.04, last revised 2019/04/25.

[†]E-mail: wybo@dekkerdocumenten.nl

`esc i` the cursor position with `<escape>i`. After typing these escape sequences, you will be in insert mode between the braces pair, ready to type what should be added.

`\rep` `\rep{phrase}{replacement}` displays `replacement` and places `n: was: phrase` in the margin. In the vim editor, you can generate this code by selecting the phrase and typing `<escape>r`. After typing this escape sequence, you will be in insert mode between the second pair of braces, ready to insert the new content.

`\com` `\com{comment}` is used by `\del`, `\yel`, `\add`, and `\rep` to place `n: comment` in the margin. You can use it to place comment in the margin without text highlighting. In the vim editor, `esc c` you can insert the command by typing `<escape>c`. After typing this escape sequence, you will be in insert mode between the braces pair, ready to type your comment.

`\hilite` `\hilite[options]{phrase}` is the command on which the above commands are based. It was published by Antal Spector-Zabusky in [StackExchange](#). It highlights the phrase with the default colour, yellow, using the default fill opacity, 0.25; but using the options, you can change this. For example, `this phrase` was highlighted with blue, with a 3pt width line in yellow around it, with the command:

```
\hilite[fill=blue,draw=yellow,opacity=.5,line width=3pt]
```

Both the fill color and the draw color get 50% opacity, but you can set each individually with the `fill opacity` and `draw opacity` options. See the documentation of the `tikz` package for more options.

2 Installing the Vim commands

The `proofread` distribution comes with a Vimball archive named `proofread.vmb`. Edit that file in vim and run the command: `:so %`. This will install the necessary files in your `~/.vim` directory, plus the Ruby executable `proofread` in your `~/bin` directory. The latter should be in your `PATH` of course, and you'll have to set its executable flag (`chmod +x ~/bin/proofread`). Finally, in order to use these facilities, you'll have to add this line at the end of your Latex source: `% vim: syntax=proofread`.

`esc u` The majority of the escape commands that are installed by this procedure have been described above. However, there are two others: The vim sequence `<escape>u` will *undo* the nearest of the three letter commands (`\del`, `\add`, `\yel`, `\rep`, `\com`) described above, which starts *before* the current cursor position (which may be even on the first character after the starting `\`). So if it sees `\del{something}` it will replace that with `something`, and if it sees `\add{something}` it will remove that.

`esc h` The counterpart is the `<escape>h` sequence, which will *honor* the first command found before the cursor. If it sees `\del{something}` it will remove it, and if it sees `\add{something}` it will replace it with `something`. The following table illustrates what happens with these two escape sequences:

	after <code><escape>u</code>	after <code><escape>h</code>
<code>\del{something}</code>	<code>something</code>	
<code>\add{something}</code>		<code>something</code>
<code>\yel{something}</code>	<code>something</code>	<code>something</code>
<code>\yel[remark]{something}</code>	<code>something</code>	<code>something</code>
<code>\com{something}</code>		
<code>\rep{old}{new}</code>	<code>old</code>	<code>new</code>

As you see, both sequences simply undo `\yel` and `\com` commands, because they are comments, not corrections.

3 Package options

Options are provided, displaying the document in various spacing and correction modes.

<code>onehalfspacing</code>	The <code>onehalfspacing</code> option displays the document with 1.5 times the normal line spacing.
<code>doublespacing</code>	The <code>doublespacing</code> option displays the document with doubled line spacing.
<code>uncorrected</code>	The <code>uncorrected</code> displays the document in its uncorrected state, without notes in the margin.
<code>corrected</code>	The <code>corrected</code> displays the document in its corrected state, without notes in the margin.
<code>frame</code>	Finally, the <code>frame</code> option is provided for visually handicapped users; it causes .5pt black borders to be drawn around the colored backgrounds around <code>\marked</code> texts.

4 Implementation

```
1 \langle *package \rangle
```

Option handling: For the `onehalfspacing` and `doublespacing` options we need the `setspace` package, but the memoir class has its own version for this. So if `\DoubleSpacing` is defined, we redefined the other commands needed. The `PR@spaced` remembers if margin notes need a shift up.

```
2 \RequirePackage{marginnote}
3 \ifx\undefined\DoubleSpacing
4   \RequirePackage{setspace}
5 \else
6   \let\setstretch\setSpacing
7   \let\onehalfspacing\OnehalfSpacing
8   \let\doublespacing\DoubleSpacing
9 \fi
10 \newif\ifPR@spaced\PR@spacedfalse
11 \newdimen\PR@unit\PR@unit.6\baselineskip
12 \DeclareOption{onehalfspacing}{\onehalfspacing\PR@spacedtrue\PR@unit.525\baselineskip}
13 \DeclareOption{doublespacing}{\doublespacing\PR@spacedtrue\PR@unit.35\baselineskip}
```

The default is to show corrections; the `corrected` and `uncorrected` options will show the document without those in either corrected or uncorrected state.

```
14 \newif\ifPR@corrected\PR@correctedfalse
15 \newif\ifPR@uncorrected\PR@uncorrectedfalse
16 \DeclareOption{corrected}{\PR@correctedtrue}
17 \DeclareOption{uncorrected}{\PR@uncorrectedtrue}
```

The `frame` option draws a black 0.5pt frame around colored areas for the visually disabled.

```
18 \newif\ifPR@frame\PR@framefalse
19 \newdimen\PR@lw\PR@lw=0pt
20 \DeclareOption{frame}{\PR@lw=.5pt\PR@frametrue}
21 \ProcessOptions
```

The following code for a text highlighting command (here renamed to `\hilite` was published by Antal Spector-Zabusky in [StackExchange](#).

```
22 \RequirePackage{soul}
23 \RequirePackage{tikz}
24 \usetikzlibrary{calc}
25 \usetikzlibrary{decorations.pathmorphing}
26
27 \newcommand{\PR@defhiliter}[3][[]]{%
28   \tikzset{every hiliter/.style={fill=#2,fill opacity=#3, #1}}%
29 }
30
31 \PR@defhiliter{yellow}{.25}
```

```

32
33 \newcommand{\PR@hilite@Dohilite}{
34   \fill [ decoration = {random steps, amplitude=1pt, segment length=15pt}
35     , outer sep = -15pt, inner sep = 0pt, decorate
36     , every hiliter, this hiliter ]
37     ($(\begin hilite)+(0,8pt)$) rectangle ($(\end hilite)+(0,-3pt)$) ;
38 }
39
40 \newcommand{\PR@hilite@Beginhilite}{
41   \coordinate (begin hilite) at (0,0) ;
42 }
43
44 \newcommand{\PR@hilite@Endhilite}{
45   \coordinate (end hilite) at (0,0) ;
46 }
47
48 \newdimen\PR@hilite@previous
49 \newdimen\PR@hilite@current

```

\hilite

```

50 \DeclareRobustCommand*\hilite[1][ ]{%
51   \ifPR@frame%
52     \tikzset{this hiliter/.style={#1,draw=black,line width=\PR@lw}}%
53   \else%
54     \tikzset{this hiliter/.style={#1}}%
55   \fi
56   \SOUL@setup
57   %
58   \def\SOUL@preamble{%
59     \begin{tikzpicture}[overlay, remember picture]
60       \PR@hilite@Beginhilite
61       \PR@hilite@Endhilite
62     \end{tikzpicture}%
63   }%
64   %
65   \def\SOUL@postamble{%
66     \begin{tikzpicture}[overlay, remember picture]
67       \PR@hilite@Endhilite
68       \PR@hilite@Dohilite
69     \end{tikzpicture}%
70   }%
71   %
72   \def\SOUL@everyhyphen{%
73     \discretionary{%
74       \SOUL@setkern\SOUL@hyphkern
75       \SOUL@sethyphenchar
76       \tikz[overlay, remember picture] \PR@hilite@Endhilite ;%
77     }{%
78     }{%
79       \SOUL@setkern\SOUL@charkern
80     }%
81   }%
82   %
83   \def\SOUL@everyexhyphen##1{%
84     \SOUL@setkern\SOUL@hyphkern
85     \hbox{##1}%
86     \discretionary{%
87       \tikz[overlay, remember picture] \PR@hilite@Endhilite ;%
88     }{%
89     }{%
90       \SOUL@setkern\SOUL@charkern
91     }%

```

```

92 }%
93 %
94 \def\SOUL@everysyllable{%
95   \begin{tikzpicture}[overlay, remember picture]
96     \path let \p0 = (begin hilite), \p1 = (0,0) in \pgfextra
97       \global\PR@hilite@previous=\y0
98       \global\PR@hilite@current =\y1
99     \endpgfextra (0,0) ;
100    \ifdim\PR@hilite@current < \PR@hilite@previous
101      \PR@hilite@Dohilite
102      \PR@hilite@Beginhilite
103    \fi
104    \end{tikzpicture}%
105    \the\SOUL@syllable
106    \tikz[overlay, remember picture] \PR@hilite@Endhilite ;%
107  }%
108  \SOUL@
109 }

```

Reduce minimum vertical space between margin paragraphs; if the memoir class is active, use the outer margin:

```

110 \AtBeginDocument{\marginparpush2pt}
111 \ifx\undefined\marginparmargin\else\marginparmargin{outer}\fi

```

We need a save box and a counter for prefixing the margin paragraphs:

```

112 \newcount\PR@markerno\PR@markerno=1

```

`\com` Make a raggedright margin note, in footnote fontsize, prefixed with the counter plus a colon:

```

113 \newcommand{\com}[1]{%
114   \marginnote{%
115     \setstretch{1}%
116     \raggedright%
117     \footnotesize%
118     \the\PR@markerno: #1%
119   }[\PR@skip]%
120   \global\PR@skipOpt
121   \global\advance\PR@markerno1%
122 }

```

`\skp`

```

123 \newdimen\PR@skip\PR@skip0\PR@unit
124 \newcommand{\skp}[1][1]{%
125   \PR@skip#1\PR@unit%
126 }

```

`\del`

```

127 \newcommand{\del}[1]{%
128   \com{delete}%
129   \hilite[red]{#1}%
130 }

```

`\yel`

```

131 \newcommand{\yel}[2][ ]{%

```

```

132 \com{#1}%
133 \hilite{#2}%
134 }

```

\add

```

135 \newcommand{\add}[1]{%
136 \com{add}%
137 \hilite[green]{#1}%
138 }

```

\rep

```

139 \newcommand{\rep}[2]{%
140 \com{was: #1}%
141 \hilite[blue]{#2}%
142 }

```

If one of the corrected or uncorrected options is used, redefine the above commands accordingly:

```

143 \ifPR@uncorrected
144 \ifPR@corrected
145 \PackageError{proofread}{You may not use both the corrected and uncorrected options}
146 \fi
147 \def\com#1{} \def\add#1{} \def\del#1{#1} \def\rep#1#2{#1}
148 \renewcommand{\yel}[2] [] {#2}\renewcommand{\hilite}[2] [] {#2}
149 \else
150 \ifPR@corrected
151 \def\com#1{} \def\add#1{#1} \def\del#1{} \def\rep#1#2{#2}
152 \renewcommand{\yel}[2] [] {#2}\renewcommand{\hilite}[2] [] {#2}
153 \fi
154 \fi
155
156 \endinput
157 </package>

```

5 Change History

v1.00	General: First public release 1	down marginnotes - vim commands are provided for removing notes, either honoring or undoing the correction. 1
v1.01	General: running counter was sometimes not advanced 1	
v1.02	General: - Now works in tables, footnotes, minipages and more - package options frame, corrected, uncorrected added - package options doublespacing, onehalfspacing added - skp command for manual moving	v1.03 General: added warning to use xetex or luatex for non-ASCII source. 1
		v1.04 General: removed overstriking in the <code>com</code> command, as it forces the text on a single line. 1

6 Index

Numbers written in *italics* 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.

	A	<code>\ifPR@frame</code> 18, 51	<code>\PR@spacedfalse</code> 10
<code>\add</code>	1, <u>135</u> , 147, 151	<code>\ifPR@spaced</code> 10	<code>\PR@spacedtrue</code> 12, 13
		<code>\ifPR@uncorrected</code> . 15, 143	<code>\PR@uncorrectedfalse</code> . 15
	C		<code>\PR@uncorrectedtrue</code> . . 17
<code>\com</code>	2, <u>113</u> , 128,	M	<code>\PR@unit</code> 11, 12, 13, 123, 125
	132, 136, 140, 147, 151	<code>\marginnote</code> 114	
<code>\corrected</code>	3		R
		O	<code>\rep</code> 2, <u>139</u>
	D	<code>\onehalfspacing</code> . . 3, 7, 12	
<code>\del</code>	1, <u>127</u> , 147, 151		S
<code>\doublespacing</code> . .	3, 8, 13		<code>\skp</code> 1, <u>123</u>
		P	<code>\SOUL@</code> 108
	E	<code>\PR@correctedfalse</code> . . . 14	<code>\SOUL@charkern</code> 79, 90
<code>\esc_a</code>	1	<code>\PR@correctedtrue</code> 16	<code>\SOUL@everyexhyphen</code> . . 83
<code>\esc_c</code>	2	<code>\PR@defhiliter</code> 27, 31	<code>\SOUL@everyhyphen</code> 72
<code>\esc_d</code>	1	<code>\PR@framefalse</code> 18	<code>\SOUL@everysyllable</code> . . . 94
<code>\esc_h</code>	2	<code>\PR@frametrue</code> 20	<code>\SOUL@hyphkern</code> 74, 84
<code>\esc_i</code>	2	<code>\PR@hilite@Beginhilite</code>	<code>\SOUL@postamble</code> 65
<code>\esc_r</code>	2 40, 60, 102	<code>\SOUL@preamble</code> 58
<code>\esc_u</code>	2	<code>\PR@hilite@current</code> . . .	<code>\SOUL@sethyphenchar</code> . . 75
<code>\esc_y</code>	1 49, 98, 100	<code>\SOUL@setkern</code> 74, 79, 84, 90
		<code>\PR@hilite@Dohilite</code> . .	<code>\SOUL@setup</code> 56
	F 33, 68, 101	<code>\SOUL@syllable</code> 105
<code>\frame</code>	3	<code>\PR@hilite@Endhilite</code> .	
	 44, 61, 67, 76, 87, 106	U
	H	<code>\PR@hilite@previous</code> . .	<code>\uncorrected</code> 3
<code>\hilite</code>	2, <u>50</u> 48, 97, 100	
		<code>\PR@lw</code> 19, 20, 52	
	I	<code>\PR@markerno</code> 112, 118, 121	Y
<code>\ifPR@corrected</code> 14, 144, 150		<code>\PR@skip</code> 119, 120, 123, 125	<code>\yel</code> 1, <u>131</u> , 148, 152