The \texttt{hhline} package*

David Carlisle

2020/01/04

This file is maintained by the \LaTeX{} Project team. Bug reports can be opened (category \texttt{tools}) at https://latex-project.org/bugs.html.

\hspace{\fill}

Abstract

\texttt{\textbackslash hhline} produces a line like \texttt{\textbackslash hline}, or a double line like \texttt{\textbackslash hline\textbackslash hline}, except for its interaction with vertical lines.

1 Introduction

The argument to \texttt{\textbackslash hhline} is similar to the preamble of an \texttt{array} or \texttt{tabular}. It consists of a list of tokens with the following meanings:

\begin{itemize}
\item \texttt{=} A double hline the width of a column.
\item \texttt{-} A single hline the width of a column.
\item \texttt{~} A column with no hline.
\item \texttt{|} A vline which ‘cuts’ through a double (or single) hline.
\item \texttt{:} A vline which is broken by a double hline.
\item \texttt{#} A double hline segment between two vlines.
\item \texttt{t} The top half of a double hline segment.
\item \texttt{b} The bottom half of a double hline segment.
\item \texttt{* \{3\}{{{#}}} expands to \texttt{==###==#}, as in the \texttt{*}-form for the preamble.
\end{itemize}

If a double vline is specified (| | or ::) then the hlines produced by \texttt{\textbackslash hhline} are broken. To obtain the effect of an hline ‘cutting through’ the double vline, use a \# or omit the vline specifiers, depending on whether or not you wish the double vline to break.

The tokens \texttt{t} and \texttt{b} must be used between two vertical rules. \texttt{|tb|} produces the same lines as \#, but is much less efficient. The main use for these are to make constructions like \texttt{|t}: (top left corner) and \texttt{:b|} (bottom right corner).

If \texttt{\textbackslash hhline} is used to make a single hline, then the argument should only contain the tokens \texttt{-}, \texttt{~} and \texttt{|} (and \texttt{*}-expressions).

\*This file has version number v2.04, last revised 2020/01/04.
An example using most of these features is:

\begin{tabular}{||cc||c|c||}
\hline
\text{a}&\text{b}&\text{c}&\text{d}\
\hline
\text{1}&\text{2}&\text{3}&\text{4}\
\hline
\text{i}&\text{j}&\text{k}&\text{l}\
\hline
\text{w}&\text{x}&\text{y}&\text{z}\
\hline
\end{tabular}

The lines produced by \LaTeX's \texttt{\\hline} consist of a single (\TeX primitive)  \\texttt{\\hrule}. The lines produced by \texttt{\\hhline} are made up of lots of small line segments. \LaTeX will place these very accurately in the .\texttt{dvi} file, but the program that you use to print the .\texttt{dvi} file may not line up these segments exactly. (A similar problem can occur with diagonal lines in the \texttt{picture} environment.)

If this effect causes a problem, you could try a different driver program, or if this is not possible, increasing \texttt{\\arrayrulewidth} may help to reduce the effect.

2 The Macros

\HH@box Makes a box containing a double hline segment. The most common case, both rules of length \texttt{\\doublerulesep} will be stored in \texttt{\box1}, this is not initialised until \texttt{\\hhline} is called as the user may change the parameters \texttt{\\doublerulesep} and \texttt{\\arrayrulewidth}. The two arguments to \HH@box are the widths (ie lengths) of the top and bottom rules.

\def\HH@box#1#2{\vbox{\hrule ~@height \arrayrulewidth ~@width #1 \vskip \doublerulesep \hrule ~@height \arrayrulewidth ~@width #2}}

\HH@add Build up the preamble in the register \texttt{\toks0}.

\def\HH@add#1{\toks0=\expandafter{\the\toks0#1}}

\HH@xexpast We ‘borrow’ the version of \texttt{@xexpast} from Mittelbach’s \texttt{array.sty}, as this allows \texttt{#} to appear in the argument list.

\def\HH@xexpast#1*#2#3#4\@@{\@tempcnta #2 \toks0={#1}\@temptokena={#3}\let\@temp@toks\relax
\let\the@toksz\relax \let\the@toks\relax
\def\@tempa{\the@toksz}\ifnum\@tempcnta >0 \@whilenum\@tempcnta >0\do
\edef\@tempa{\@tempa\the@toks}\advance \@tempcnta \m@ne
\let \@tempb \HH@xexpast\else
\let \@tempb \HH@xexpast\fi

2
\hhline

Use a simplified version of \@mkpream to break apart the argument to \hhline. Actually it is oversimplified. It assumes that the vertical rules are at the end of the column. If you were to specify c|@\{xx\}| in the array argument, then \hhline would not be able to access the first vertical rule. (It ought to have an @ option, and add \leaders up to the width of a box containing the @-expression. We use a loop made with \futurelet rather than \@tfor so that we can use # to denote the crossing of a double hline with a double vline.

\if\firstamp is true in the first column and false otherwise.

\if\tempswa is true if the previous entry was a vline (:, | or #).

\def\hhline#1{\omit\@firstamptrue\@tempswafalse

Put two rules of width \doublerulesep in \box1

If Mittelbach’s array.sty is loaded, we do not need the negative \hskip’s around vertical rules.

\def\tempc{\ifx\extrarowheight\HH@undef\hskip-.5\arrayrulewidth\fi}

Now expand the \*-forms and add dummy tokens ( \relax and ‘) to either end of the token list. Call \HH@let to start processing the token list.

\HH@let Discard the last token, look at the next one.

\HH@loop The main loop. Note we use \ifx rather than \if in version 2 as the new token ~ is active.

\def\HH@loop{% If next token is ‘, stop the loop and put the lines into this row of the alignment.

\if\tempb\def\next##1{\the\toks@\cr}\else\let\next\HH@let \ |

|, add a vertical rule (across either a double or single hline).

\if\tempb\if\tempswa\HH@add{\hskip\doublerulesep}\fi\tempswatrue

\HH@add{\@tempc\vline\@tempc}\else \;,

; add a broken vertical rule (across a double hline).

\if\tempb\if\tempswa\HH@add{\hskip\doublerulesep}\fi\tempswatrue

\HH@add{\@tempc\vline\@tempc}\else 

#, add a double hline segment between two vlines.

\if\tempswafalse \if\tempswa\HH@add{\hskip\doublerulesep}\fi\tempswatrue

\HH@add{\@tempc\vline\@tempc\copy\@ne\@tempc\vline\@tempc}\else

~, A column with no hline (this gives an effect similar to \cline).

\if\tempswafalse \if\tempswa\HH@add{\hskip\doublerulesep}\fi\tempswatrue

\HH@add{\@tempc\vline\@tempc\copy\@ne\@tempc\vline\@tempc}\else

\HH@add{\@fil}\else

3
-, add a single hline across the column.
37 \elif\@tempb-\@tempsvafalse
38 \ift\firstamp\firstampfalse\else\HH@add{&\omit}\fi
39 \HH@add{\leaders\hrule@height\arrayrulewidth\hfil}\else
=, add a double hline across the column.
40 \elif\@tempb=\@tempsvafalse
41 \ift\firstamp\firstampfalse\else\HH@add{&\omit}\fi
42 \HH@add{\rlap{\copy\@ne}\leaders\copy\@ne\hfil\llap{\copy\@ne}}\else
43 t, add the top half of a double hline segment, in a \rlap so that it may be used
44 \ift\@tempb t\HH@add{\rlap{\HH@box\doublerulesep\z@}}\else
45 b, add the bottom half of a double hline segment in a \rlap so that it may be
46 \ift\@tempb b\HH@add{\rlap{\HH@box\z@\doublerulesep}}\else
47 space, Gobble the space and loop again.
48 \ift\@tempb\@stoken\let\next\HH@spacelet\else
49 Otherwise ignore the token, with a warning.
50 \PackageWarning{hhline}{\meaning\@tempb space ignored in \noexpand\hhline argument%}
51 \MessageBreak%}
52 \fi\fi\fi\fi\fi\fi\fi\fi\fi\fi
53 Go around the loop again.
54 \next}
55 \HH@spacelet Helper macro to gobble a space token and continue the loop.
56 \lowercase{\def\HH@spacelet}{\futurelet\@tempb \HH@loop}
57 (/package)