The \texttt{hhline} package*

David Carlisle

2020/01/04

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

1 Introduction

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

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

If a double vline is specified (\| or ::) then the hlines produced by \texttt{\texttt{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{\texttt{hhline}} is used to make a single hline, then the argument should only contain the tokens \texttt{-}, \texttt{~} and \texttt{|} (and \*\-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||}
<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
</tr>
<tr>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
</tr>
</tbody>
</table>
\end{tabular}

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

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

2 The Macros

1 \texttt{(\textbackslash HH@box)}

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

2 \texttt{\texttt{\textbackslash def\HH@box#1#2{}}}

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

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

6 \texttt{\texttt{\textbackslash def\HH@add#1{}}}

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

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

7 \texttt{\texttt{\textbackslash def\HH@xexpast#1##2##3###\@0{}}}

8 \texttt{\texttt{\@tempcnta \#2}}

9 \texttt{\texttt{\texttt{\toks0=(#1)\@emptokena=(#3)\@}}}

10 \texttt{\texttt{\texttt{\let\the\@toksz\relax \let\the\@toks\relax}}}

11 \texttt{\texttt{\texttt{\def\@tempa{\the\@toksz}}}}

12 \texttt{\texttt{\texttt{\ifnum\@tempcnta > 0 \@whilenum\@tempcnta > 0\do}}}

13 \texttt{\texttt{\texttt{\if\@em\@tempa\@tempa\the\@toks\advance\@tempcnta \m@ne\fi}}}

14 \texttt{\texttt{\texttt{\let\@tempb \HH@xexpast \else}}}

15 \texttt{\texttt{\texttt{\let\@tempb \HH@xexpnoo \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
\global\setbox\@ne\HH@box\doublerulesep\doublerulesep
If Mittelbach's array.sty is loaded, we do not need the negative \hskip's around vertical rules.
\xdef\@tempc{\ifx\extrarowheight\HH@undef\hskip-.5\arrayrulewidth\fi}%
\begin{verbatim}
\HH@xexpast\relax#1*0x\@@\toks@{}\expandafter\HH@let\@tempa'}
\HH@let
\HH@loop
\begin{verbatim}
\HH@loop
\end{verbatim}
-, add a single hline across the column.
37 \ifx\@tempb-\@tempswafalse
38 \if@firstamp\@firstampfalse\else\HH@add{&\omit}\fi
39 \HH@add{\leaders\hrule\@height\arrayrulewidth\hfil}\else
=, add a double hline across the column.
40 \ifx\@tempb-\@tempswafalse
41 \if@firstamp\@firstampfalse\else\HH@add{&\omit}\fi
Put in as many copies of \box1 as possible with \leaders, this may leave gaps at the ends, so put an extra box at each end, overlapping the \leaders.
42 \HH@add
43 {\rlap{\copy\@ne}\leaders\copy\@one\hfil\llap{\copy\@one}}\else
44 \ifx\@tempb t\HH@add{\rlap{\HH@box\doublerulesep\z@}}\else
45 \ifx\@tempb b\HH@add{\rlap{\HH@box\z@\doublerulesep}}\else
space, Gobble the space and loop again.
46 \ifx\@tempb\@sptoken\let\next\HH@spacelet\else
Otherwise ignore the token, with a warning.
47 \PackageWarning{hhline}{\meaning\@tempb\space ignored in \noexpand\hhline argument%}
48 \MessageBreak%}
49 \fi\fi\fi\fi\fi\fi\fi
Go around the loop again.
51 \next}
\HH@spacelet Helper macro to gobble a space token and continue the loop.
52 \lowercase{\def\HH@spacelet}{\futurelet\@tempb \HH@loop}
53 (/package)