The \texttt{tabulary} package*

David Carlisle

2024/06/04

1 User Documentation

\begin{tabulary}{⟨\texttt{length}⟩}{⟨\texttt{pream}⟩} ... \end{tabulary}

The rather daft name may change in a later release but it is a pun on \texttt{tabularx}, which itself was a pun on \texttt{tabular*}...

These environments work pretty much like the standard tabular environment (or more correctly, the enhanced version from the array package) except that there are more possibilities for the column types.

\textbf{LCRJ} These new ‘uppercase’ column types are only activated in the \texttt{tabulary} environment. In order to make the total table width equal to ⟨\texttt{length}⟩ the LCRJ columns are converted to \texttt{p} columns (with \texttt{\raggedright}, \texttt{\centering}, or \texttt{\raggedleft} or normal justification respectively applied). The width of these converted columns is proportional to the natural width of the longest entry in each column.

To stop very narrow columns being too ‘squeezed’ by this process any columns that are narrower than \texttt{\tymin} are set to their natural width. This length may be set with \texttt{\setlength} and is arbitrarily initialised to 10 pt. (If you know that a column will be narrow, it may be preferable to use, say, \texttt{c} rather than \texttt{C} so that the \texttt{tabulary} mechanism is never invoked on that column.)

Similarly one very large entry can force its column to be too wide. So to prevent this, all columns with natural length greater than \texttt{\tymax} are set to the same width (with the proportion being taken as if the natural length was equal to \texttt{\tymax}). This is initially set to twice the text width.

Narrow \texttt{p} columns are sometimes quite hard to set, and so you may redefine the command \texttt{\tyformat} to be any declarations to make just after the \texttt{\centering} or \texttt{\ragged}... declaration. By default it redefines \texttt{\everypar} to insert a zero space at the start of every paragraph, so the first word may be hyphenated. (See DogBook).

As the environment makes a standard \LaTeX{} box, it will be indented by the paragraph indent at the start of a paragraph, and so will not fit on a line if given argument \texttt{\textwidth} unless it is preceded by \texttt{\noindent} or is in a \texttt{center} environment or some other environment with zero paragraph indent.

*This file has version number v0.11, last revised 2024/06/04.
2 Features

You can use \texttt{\texttt{\textbackslash multicolumn}} but if the multicolumn text turns out to be longer than the final calculated widths of the columns that it spans, then the final table will be too wide.

\verb+ doesn’t work. (except in restricted version as in \texttt{tabularx})

The whole table is evaluated twice, so take care with some \TeX constructions that may have side effects like writing to files.

3 Options

The following package option is defined:

\texttt{debugshow} Causes a lot of stuff to appear on the terminal. I find this invaluable, you may find it less so.
## 4 Examples

<table>
<thead>
<tr>
<th>C columns</th>
<th>J columns</th>
<th>L, R and C columns, and <code>\texttt{\textbackslash multicolumn}</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>the rain in spain (an @ expr.)</td>
<td>the rain in spain (an @ expr.)</td>
<td>some multicolumn text across columns 2–4</td>
</tr>
<tr>
<td>falls mainly on the plain</td>
<td>falls mainly on the plain</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>c</td>
<td>c</td>
<td>c</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>b</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>(an @ expr.) c</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td></td>
</tr>
</tbody>
</table>
The following examples attempt to show the effect of the `\texttt{tymin}` and `\texttt{tymax}` parameters. One should also perhaps note that `\texttt{tymax}` refers to the total column width (including any inter-column space, rules etc) but `\texttt{tymin}` just refers to the width of the column entry (like the argument to the standard `p` column).

\texttt{tymin=0pt}
\texttt{tymax=\maxdimen}
Note how the first column is ‘squeezed’. In fact it is in such a narrow column that even ‘a’ produces an overfull box warning!

\[
\begin{tabular}{|c|c|c|c|}
\hline
a & b & c & d \\
\hline
b & e & c & c \\
\hline
b & c & c & c \\
\hline
\end{tabular}
\]

\texttt{tymax=20pt}
\texttt{tymax=\maxdimen}
Here increase `\texttt{tymax}` so that columns b and a are not so narrow. ‘a’ is set to its natural width, and ‘b’ is set to `\texttt{tymin}`.

\[
\begin{tabular}{|c|c|c|c|}
\hline
a & b & b & b \\
\hline
b & c & c & c \\
\hline
\end{tabular}
\]

\texttt{tymin=20pt}
\texttt{tymax=200pt}
In the previous example, the large d column dominated the table, being a lot wider than the c column. By reducing `\texttt{tymax}` can limit the width of column d producing more even column widths, but now producing an entry for d that is longer than that for c.

\[
\begin{tabular}{|c|c|c|c|}
\hline
a & b & b & b \\
\hline
b & e & e & e \\
\hline
\end{tabular}
\]
5 The Code

1 ⟨∗package⟩
2 Rollback.
3 \DeclareRelease{v0.10}{1995/10/08}{tabulary-v010.sty}
4 \DeclareCurrentRelease{}{2024-06-01}
5 This version needs a current array package
6 \RequirePackage{array}{2024/05/23}
7 \catcode'\Z=14
8 \DeclareOption{debugshow}{\catcode'\Z=9\relax}
9 \ProcessOptions
10 \arraybackslash
11 Borrowed from tabularx.
12 \def\arraybackslash{\let\=\@arraycr}
13 \@finalstrut
14 Bug fixed version from December 1995 \LaTeX{} release. Old bug going back to \LaTeX{}2.09...
15 \def\@finalstrut#1{%
16 \unskip\ifhmode\nobreak\fi\vrule\@width\z\@height\z\@depth\dp#1}
17 \TY@count
18 Counter so that we know what column we are hacking around in.
19 \newcount\TY@count
20 \tabulary
21 Top level macro for standard form.
22 \def\tabulary{%
23 \let\TY@final\tabular
24 \let\endTY@final\endtabular
25 \TY@tabular
26 \TY@tabular
27 Looks a lot like tabularx at this stage. Grab everything into a token register.
28 \def\TY@tabular#1{%
29 \edef\TY@{\@currenvir}%
30 {\ifnum0='\fi}
31 At this point need to save locally things that \tabulary{} will globally mess up. These
32 are restored at the end of the environment.
33 \@ovxx\TY@linewidth
34 \@ovyy\TY@tablewidth
35 \count@\z
36 \@tempswatrue
37 \@whilesw\if@tempswa\fi{%
38 \advance\count@\@ne
39 \expandafter\ifx\csname TY@F\the\count@\endcsname\relax
40 \expandafter\let\csname TY@SF\the\count@\expandafter\endcsname\csname TY@F\the\count@\endcsname
41 \expandafter\let\csname TY@S\the\count@\expandafter\endcsname\csname TY@\the\count@\endcsname
42 }%
43 \@tempswafalse
44 \@whilesw\if@tempswa\fi{%
45 \@tempswafalse
46 \@tempswafalse
47 }%
Placing this here means that nested tabulars will get this definition but that's probably OK, the extra code for LCR etc shouldn't do any harm.

\let\@classz
\let\verb
\toks@{}
\TY@get@body

\TY@@mkpream
Saved version.
\let\TY@@mkpream\@mkpream
\TY@mkpream
TY version.
\ExplSyntaxOn
\def\TY@mkpream{%
  \def\@addamp{%
    \if@firstamp \@firstampfalse \else
      \global\advance\TY@count\@ne
      \edef\@preamble{\@preamble & \noexpand\tbl_update_cell_data:}
    \fi
  }%
  \def\@acol{%
    \TY@subwidth\col@sep
    \@addtopreamble{\hskip\col@sep}
  }%
  \let\@arrayrule\TY@arrayrule
  \let\@classvi\TY@classvi
  \def\@classv{\save@decl
    \expandafter\NC@ecs\@nextchar\extracolsep\extracolsep\@@__tbl
    \sbox\z@{\d@llarbegin\@nextchar\d@llarend}%
    \TY@subwidth{\wd\z@}%
    \@addtopreamble{\d@llarbegin\the@toks\the\count@\relax\d@llarend}%
    \prepnext@tok}
  }%
  \global\let\@mkpream\TY@@mkpream
  \TY@@mkpream}
\ExplSyntaxOff

\TY@arrayrule
Pull this out so the colortbl support below can redefine
\def\TY@arrayrule{%
  \TY@subwidth\arrayrulewidth
  \@addtopreamble{\vline}
}
\ TYPOclassvi Pull this out so the colorbl Support below can redefine
\ TYPOtab First run a tabular with all the column types fudged so that the widths of any
rules or \-expressions are noted.
\ TYPOtabarray
\ TYPOwidth Just a shorthand to access a column width macro.
\ TYPOsubwidth Subtract a width from the current column width and also the total line table
width and the target line width.
\ endtabulary First run one modified tabular, making sure to add a blank row (cf longtable) to
the end in case the user supplied last row is hidden by an hline or something.

\ TYPOclassvi \def\TYPOclassvi{\ifcase \@lastchclass
\@acol \or
\TYPOsubwidth\doublerulesep
\@addtopreamble{\hskip \doublerulesep}\or
\@acol \or
\@classvii
\fi}

\ TYPOtab \def\TYPOtab{%
\setbox\z@\hbox{\bgroup
\advance\dimen@-#1\relax
\global\advance\TYPOlinewidth-#1\relax}
\TYPOtabarray

\ TYPOwidth \def\TYPOwidth#1{\expandafter#1\csname TYPO@the\the\TYPOcount\endcsname}

\ TYPOsubwidth \def\TYPOsubwidth#1{%
\advance\dimen@-#1\relax
\global\advance\TYPOlinewidth-#1\relax}

\ endtabulary
Check that $\tymin$ is not too large.

$\dimen@ \TY@linewidth$
$\divide\dimen@ \TY@count$
$\ifdim\dimen@ < \tymin$
$\TY@warn{\tymin \ dimen@}$
$\tymin \dimen@$

Now take the last row apart, cf longtable or appendix D.

$\setbox\tw@ = \hbox{\unhbox\@ne}$
$\loop$
$\@tempdima = \lastskip$
$\ifdim\@tempdima > 0$
Z \message{ecs=\the\@tempdima}^J$
$\global\advance\TY@linewidth - \@tempdima$
$\fi$
$\unskip$
$\setbox\tw@ = \lastbox$
$\ifhbox\tw@$
Z \message{Col \the\TY@count: Initial=\the\wd\tw@/space}$
$\ifdim\wd\tw@ > \tymax$
\message{\wd\tw@/tymax}
$\message{\@max\space}$
$\else$
\message{\@spaces\space}$
$\fi$
$\TY@width\dimen@$
$\message{\the\dimen@/space}$
$\advance\dimen@/\wd\tw@$
$\message{\Final=\the\dimen@/space}$
$\TY@width\xdef\the\dimen@/ \tymin$
$\message{< \tymin}$
$\global\advance\TY@linewidth - \dimen@$
$\message{\the\dimen@/}$
$\else$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{***}$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{\the\dimen@/}$
$\else$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{***}$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{\the\dimen@/}$
$\else$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{***}$
$\expandafter\ifx\csname TY@F\the\TY@count\endcsname \z@$
$\message{\the\dimen@/}$
$\else$
A bit cheap just doing this four times, but prevents any possibilities of looping.

Reset the counter.

Reset the LCRJ column definition to set paragraphs to the calculated widths.

Restore counter values.

Run a second tabular, and for the star form, unbox it.

Finish off by restoring global commands.

\def\TY@checkmin{% 
\let\TY@checkmin\relax
\ifdim\TY@tablewidth>\z@
\Gscale@div\TY@ratio\TY@linewidth\TY@tablewidth
% \changes{v0.9}{2008/12/01}{\cs{TY@linewidth}}
\ifdim\TY@tablewidth<\TY@linewidth
\def\TY@ratio{1}%
\else
\global\expandafter\let\csname TY@SF\the\count0\endcsname\relax
\expandafter\ifx\csname TY@SF\the\count0\endcsname\relax
\@tempswafalse
\else
\global\expandafter\let\csname TY@F\the\count0\endcsname\csname TY@S\the\count0\endcsname
\fi
\fi
\TY@linewidth\@ovxx
\TY@tablewidth\@ovyy
\ifnum0='{\fi}
\TY@checkmin
Check that no column is squeezed below \tymin. If it is, fix the width of that
column to \tymin and try again re-computing the ratio. (The new ratio will be
smaller, and may squeeze yet more rows, so need to iterate this, currently just do
it four times.)
\fi
\else
\TY@warn{}No suitable columns!}%
\def\TY@ratio{1}%
\fi
\count@\z@
\message{^^JLine Width: \the\TY@linewidth,}
\message{Natural Width: \the\TY@tablewidth,}
\message{Ratio: \TY@ratio^^J}%
\@tempdima\z@
\loop
\ifnum\count@<\TY@count
\advance\count@\@ne
\ifdim\csname TY@F\the\count@\endcsname>\tymin
\dimen@\csname TY@\the\count@\endcsname
\dimen@\TY@ratio\dimen@
\ifdim\dimen@<\tymin
\message{Column \the\count@ space ->}%
\global\expandafter\let\csname TY@F\the\count@\endcsname\tymin
\global\advance\TY@linewidth-\tymin
\global\advance\TY@tablewidth-\csname TY@\the\count@\endcsname
\let\TY@checkmin\TY@@checkmin
\else
\expandafter\xdef\csname TY@F\the\count@\endcsname{\the\dimen@}%
\advance\@tempdima\csname TY@F\the\count@\endcsname
\fi
\fi
\message{^^JTotal: \the\@tempdima^^J}%
\TY@@checkmin
\let\TY@@checkmin\TY@checkmin

\TY@linewidth Stores the target width.
\newdimen\TY@linewidth
\tyformat What to do with columns
\def\tyformat{\everypar{{\nobreak\hskip\z@skip}}}
tymin Columns narrower than this are not fudged.
\newdimen\tymin
\tymin=10pt

\tymin Columns wider than this are all treated alike and set to the same width, to stop one particularly long entry hijacking the entire table.
\newdimen\tymax
\tymax=2\textwidth
\testpatch Also add LCRJ although these don’t do anything useful except in tabulary.

\def\testpatch{}\chclass
\ifnum \lastchclass=6 \@ne \@chnum \@ne \else
\ifnum \lastchclass=7 5 \else
\ifnum \lastchclass=8 \tw@ \else
\ifnum \lastchclass=9 \thr@@
\else \z@
\ifnum \lastchclass = 10 \else
\edef\@nextchar\expandafter\string\@nextchar\@chnum\if \@nextchar c\z@ \else
\if \@nextchar l\@ne \else
\if \@nextchar r\tw@ \else
\if \@nextchar s6 \else
\if \@nextchar C7 \else
\if \@nextchar L8 \else
\if \@nextchar R9 \else
\if \@nextchar J10 \else
\z@ \@chclass
\if\@nextchar |\@ne \else
\if \@nextchar !6 \else
\if \@nextchar @7 \else
\if \@nextchar <8 \else
\if \@nextchar >9 \else
10
\@chnum
\if \@nextchar m\thr@@\else
\if \@nextchar p4 \else
\if \@nextchar b5 \else
\z@ \@chclass \@tempcnta\count@
\ifx\TY@box\TY@box@v
\global\advance\TY@count\@ne
\fi
\let\centering c\fi
\let\raggedright\noindent
\let\raggedleft\indent
\let\arraybackslash\relax
\prepnext@tok
\ifnum\@chnum<6
\global\expandafter\let\csname TY@F\the\TY@count\endcsname\z@
\fi
\ifnum\@chnum=6
\global\expandafter\let\csname TY@F\the\TY@count\endcsname\z@
\fi
\@addtopreamble{%
\fi
\fi \fi \fi \fi \fi \fi \fi
\fi
\fi \fi \fi \fi \fi \fi
\fi
\fi
\fi
\fi
\fi
\fi
\fi
\fi
\fi
\fi
\fi\fi\i

\TY@classz Here hacked around without the respect Frank’s code deserves...

\def\TY@classzz{%
\@classx
\@tempcnta\count@
\ifx\TY@box\TY@box@v
\global\advance\TY@count\@ne
\fi
\let\centering c%
\let\raggedright\noindent
\let\raggedleft\indent
\let\arraybackslash\relax
\prepnext@tok
\ifnum\@chnum<6
\global\expandafter\let\csname TY@F\the\TY@count\endcsname\z@
\fi
\ifnum\@chnum=6
\global\expandafter\let\csname TY@F\the\TY@count\endcsname\z@
\fi
\@addtopreamble{%
\fi
\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi

\hfill\hskipisp%
\TY@box

The argument is \centering etc depending on whether LCRJ is used. However in this version the entries are set in horizontal mode with definitions mimicking the standard lcr columns. Later \TY@box will be redefined to \TY@box@v which really sets the entries in vertical mode.

\TY@box@v

The version to use in a final run, set the CLRJ columns in a parbox of the appropriate width.

\TY@tablewidth

The natural width of the table on the first run.
\edef\@tempd{\the\dimen\z@}\edef#1{\strip@pt\dimen}\end\forloop
\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.

\begin{package}
\ExplSyntaxOn
\cs_set:Npn \@tempa #1 {\int_compare:nNnT \g__tbl_col_int > 0}
\else
\def\@tempb{T\hskip\hsize}\fi
\else
\@acol \or
\@classvii
\fi\}%
\let\CT@start\relax
\end of at begin document
\end{package}

\verb support, uses same csnames as in TX so they share code if both loaded (this
version names tabulary in the warning though). See tabularx for documentation.
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

\# .......................... 439, 440  \@firststampfalse ... 52 312, 388, 391, 392
\* .......................... 430, 439  \@halignto .............. 100 \@tempa 341, 342, 435,
\@ .............................. 62  \@height ................. 10, 409 436, 437, 442,
@acol .......................... 56, 77, 416, 423 446, 458, 465, 474
@addamp .......................... 51  \@ifpackageexpanded 349 \@temppb ... 443, 445, 448
@addtopreamble .................. 65, 72, 76, 278, \@ifstar .................. 434 \@tempcnta 263, 363, 372
65, 72, 76, 278, \@lastchclass ...... \@temppd ........... 322, 334
379, 414, 419, 421  \@tempdimb ...........
\@array .......................... 89, 90 22, 123, 124,
\@arraycr .......................... 8 125, 201, 216,
67, 86, 350, 359, 360 221, 407, 408, 409
\@arrayrule .......................... 59  \@tempswafalse ... 26, 174
\@chclass .......................... 230, 247, 258  \@tempswattrue ... 22, 170
172, 204, 231, \@testpatch 230
\@chnum .......................... 231, 238, \@testpatch .......
240, 248, 265, 365 230
\@chnum ... 231, 238, \@undefined ........... 474
254, 272, 275, \@warning .................. 454
\@chclass .......................... 230, 247, 258
172, 204, 231, \@whilesw ........... 23, 171
254, 272, 275, \@width ... 10, 409, 421
\@chnum .......................... 231, 238, \[ .......................... 82
254, 272, 275, \] .......................... 82
\@currenvir .................. 17 \@nil .................. 372
\@depth .................. 10 \@ovxx ................. 19, 181
\@eha .................. 319 \@ovyy ................. 20, 182
\@elt .................. 101 \@preamble ........... 54
\@empty ................. 455 \@preamerr ........ 258
\@endpbox ................. . \@spaces ........ 135
. 287, 290, 291, \@startpbox ........
314, 388, 391, 392 204, 211, 212,
\@finalstrut ........... 2 \@startpbox ........
. 287, 290, 291, 216, 265, 365, 408
Change History

v0.1  
  General: Initial version

v0.10  
  General: support \cellcolor see  
            http://tex.stackexchange.com/a/185851/1090

v0.11  
  General: Handle p and b columns  
  like m gh/38
  Restore LaTeX counters after  
  trial typesetting gh/12
  Update to match latest array  
  package (rollback to v0.10 for
  older releases)  
  use \hskip 1sp to match array  
  2.3i from 1996

v0.2  
  General: Changed everything  
  except the name

v0.3  
  General: Changed everything  
  except the name: s and CLRS

v0.4  
  \TY@checkmin: \global added
  \xdef not \edef

v0.5  
  General: Further SPQR
  modifications to multi pass  
  table env

v0.6  
  General: Remove multi pass table  
  env and unboxed star form
  \TX@warn: macro added
  \TY@arrayrule: macro added
  \TY@classvi: macro added
  \TY@tabarray: new macro to
  support [t] optional arg

v0.7  
  \TY@tabarray: new macro to
  support [t] optional arg

v0.8  
  General: Rename S to J and ‘hide’  
  s (until it works)