Red Hat Enterprise Linux Release 9.2 Manual Pages on 'gtbl.1' command

$ man gtbl.1

TBL(1)   General Commands Manual   TBL(1)

NAME

   tbl - format tables for troff

SYNOPSIS

   tbl [-Cv] [file ...]

DESCRIPTION

   This manual page describes the GNU version of tbl, which is part of the

   groff document formatting system. tbl compiles descriptions of tables

   embedded within troff input files into commands that are understood by

   troff. Normally, it should be invoked using the -t option of groff.

   It is highly compatible with Unix tbl. The output generated by GNU tbl

   cannot be processed with Unix troff; it must be processed with GNU

   troff. If no files are given on the command line or a filename of - is

   given, the standard input is read.

OPTIONS

   -C   Enable compatibility mode to recognize .TS and .TE even when

         followed by a character other than space or newline. Leader

         characters (\a) are handled as interpreted.

   -v   Print the version number.

LANGUAGE OVERVIEW

   tbl expects to find table descriptions wrapped in the .TS (table start)

   and .TE (table end) macros. Within each such table sections, another

   table can be defined by using the request .T& before the final command

.TE. Each table definition has the following structure:

Global options

> This is optional. This table part can use several of these op?
>
> tions distributed in 1 or more lines. The global option part
>
> must always be finished by a semi-colon ; .

Table format specification

> This part must be given, it is not optional. It determines the
>
> number of columns (cells) of the table. Moreover each cell is
>
> classified by being central, left adjusted, or numerical, etc.
>
> This specification can have several lines, but must be finished
>
> by a dot . at the end of the last line. After each cell defi?
>
> nition, column specifiers can be appended, but that's optional.

Cells are separated by a tab character by default. That can be changed

by the global option tab(c), where c is an arbitrary character.

SIMPLE EXAMPLES

> The easiest table definition is.
>
> > .TS
> >
> > c c c .
> >
> > This is   centered
> >
> > Well,   this also
> >
> > .TE
>
> By using c c c, each cell in the whole table will be centered. The
>
> separating character is here the default tab.
>
> The result is
>
> > This   is   centered
> >
> > Well,   this   also
>
> This definition is identical to
>
> > .TS
> >
> > tab(@);
> >
> > ccc.
> >
> > This@is@centered
> >
> > Well,@this@also
> >
> > .TE

Here, the separating tab character is changed to the letter @.

Moreover a title can be added and the centering directions can be changed to many other formats:

    .TS

    tab(@);

    c s s

    l c n .

    Title

    left@centers@123

    another@number@75

    .TE

The result is

            Title

    left   centers   123

    another   number   75

Here l means left-justified, and n means numerical, which is here right-justified.

USAGE

 Global options

    The line immediately following the .TS macro may contain any of the following global options (ignoring the case of characters ? Unix tbl only accepts options with all characters lowercase or all characters uppercase), separated by spaces, tabs, or commas:

    allbox Enclose each item of the table in a box.

    box   Enclose the table in a box.

    center Center the table (default is left-justified). The alternative keyword name centre is also recognized (this is a GNU tbl exten? sion).

    decimalpoint(c)

        Set the character to be recognized as the decimal point in nu? meric columns (GNU tbl only).

    delim(xy)

        Use x and y as start and end delimiters for eqn(1).

doublebox

Enclose the table in a double box.

doubleframe

Same as doublebox (GNU tbl only).

expand Make the table as wide as the current line length (providing a

column separation factor).   Ignored if one or more ?x? column

specifiers are used (see below).

In case the sum of the column widths is larger than the current

line length, the column separation factor is set to zero; such

tables extend into the right margin, and there is no column sep?

aration at all.

frame Same as box (GNU tbl only).

linesize(n)

Set lines or rules (e.g. from box) in n-point type.

nokeep Don't use diversions to prevent page breaks (GNU tbl only).

Normally tbl attempts to prevent undesirable breaks in boxed ta?

bles by using diversions.   This can sometimes interact badly

with macro packages' own use of diversions?when footnotes, for

example, are used.

nospaces

Ignore leading and trailing spaces in data items (GNU tbl only).

nowarn Turn off warnings related to tables exceeding the current line

width (GNU tbl only).

tab(x) Use the character x instead of a tab to separate items in a line

of input data.

The global options must end with a semicolon. There might be white?

space between an option and its argument in parentheses.

Table format specification

After global options come lines describing the format of each line of

the table.   Each such format line describes one line of the table it?

self, except that the last format line (which you must end with a pe?

riod) describes all remaining lines of the table. A single-key charac?

ter describes each column of each line of the table.   Key characters

can be separated by spaces or tabs. You may run format specifications for multiple lines together on the same line by separating them with commas.

You may follow each key character with specifiers that determine the font and point size of the corresponding item, that determine column width, inter-column spacing, etc.

The longest format line defines the number of columns in the table; missing format descriptors at the end of format lines are assumed to be L.   Extra columns in the data (which have no corresponding format entry) are ignored.

The available key characters are:

a,A   Center longest line in this column and then left-justifies all
    other lines in this column with respect to that centered line.
    The idea is to use such alphabetic subcolumns (hence the name of
    the key character) in combination with L; they are called sub?
    columns because A items are indented by 1n relative to L en?
    tries. Example:
        .TS
        tab(;);
        ln,an.
        item one;1
        subitem two;2
        subitem three;3
        .T&
        ln,an.
        item eleven;11
        subitem twentytwo;22
        subitem thirtythree;33
        .TE
    Result:
        item one   1
         subitem two   2
          subitem three   3

    item eleven   11

     subitem twentytwo   22

     subitem thirtythree   33

c,C   Center item within the column.

l,L   Left-justify item within the column.

n,N   Numerically justify item in the column: Units positions of num?

bers are aligned vertically. If there is one or more dots adja?

cent to a digit, use the rightmost one for vertical alignment.

If there is no dot, use the rightmost digit for vertical align?

ment; otherwise, center the item within the column. Alignment

can be forced to a certain position using ?\&?; if there is one

or more instances of this special (non-printing) character

present within the data, use the leftmost one for alignment.

Example:

   .TS

   n.

   1

   1.5

   1.5.3

   abcde

   a\&bcde

   .TE

Result:

   1

    1.5

   1.5.3

   abcde

    abcde

If numerical entries are combined with L or R entries ? this can

happen if the table format is changed with .T& ? center the

widest number (of the data entered under the N specifier regime)

relative to the widest L or R entry, preserving the alignment of

all numerical entries. Contrary to A type entries, there is no

extra indentation.

Using equations (to be processed with eqn) within columns which use the N specifier is problematic in most cases due to tbl's algorithm for finding the vertical alignment, as described above.   Using the global delim option, however, it is possible to make tbl ignore the data within eqn delimiters for that pur‐ pose.

r,R   Right-justify item within the column.

s,S   Span previous item on the left into this column. Not allowed for the first column.

^   Span down entry from previous row in this column.   Not allowed for the first row.

,-   Replace this entry with a horizontal line. Note that ?? and ?-? can be used for table fields only, not for column separator lines.

=   Replace this entry with a double horizontal line. Note that ?=? can be used for table fields only, not for column separator lines.

| The corresponding column becomes a vertical rule (if two of these are adjacent, a double vertical rule).

A vertical bar to the left of the first key letter or to the right of the last one produces a line at the edge of the table.

To change the data format within a table, use the .T& command (at the start of a line). It is followed by format and data lines (but no global options) similar to the .TS request.

Column specifiers

Here are the specifiers that can appear in suffixes to column key let‐ ters (in any order):

b,B   Short form of fB (make affected entries bold).

d,D   Start an item that vertically spans rows, using the ?^? column specifier or ?\^? data item, at the bottom of its range rather than vertically centering it (GNU tbl only). Example:

.TS

```
tab(;) allbox;

l l

l ld

r ^

l rd.

0000;foobar

T{

1111

.br

2222

T};foo

r;

T{

3333

.br

4444

T};bar

\^;\^

.TE
```

Result:

```
????????????????
?0000 ? foobar ?
????????????????
?1111 ?       ?
?2222 ?       ?
???????      ?
?   r ? foo    ?
????????????????
?3333 ?       ?
?4444 ?   bar ?
????????????????
```

e,E   Make equally-spaced columns. All columns marked with this spec‐

ifier get the same width; this happens after the affected column

widths have been computed (this means that the largest width value rules).

f,F   Either of these specifiers may be followed by a font name (ei? ther one or two characters long), font number (a single digit), or long name in parentheses (the last form is a GNU tbl exten? sion). A one-letter font name must be separated by one or more blanks from whatever follows.

i,I   Short form of fI (make affected entries italic).

m,M   This is a GNU tbl extension. Either of these specifiers may be followed by a macro name (either one or two characters long), or long name in parentheses. A one-letter macro name must be sepa? rated by one or more blanks from whatever follows. The macro which name can be specified here must be defined before creating the table. It is called just before the table's cell text is output. As implemented currently, this macro is only called if block input is used, that is, text between ?T{? and ?T}?. The macro should contain only simple troff requests to change the text block formatting, like text adjustment, hyphenation, size, or font. The macro is called after other cell modifications like b, f or v are output. Thus the macro can overwrite other modification specifiers.

p,P   Followed by a number, this does a point size change for the af? fected fields. If signed, the current point size is incremented or decremented (using a signed number instead of a signed digit is a GNU tbl extension). A point size specifier followed by a column separation number must be separated by one or more blanks.

t,T   Start an item vertically spanning rows at the top of its range rather than vertically centering it.

u,U   Move the corresponding column up one half-line.

v,V   Followed by a number, this indicates the vertical line spacing to be used in a multi-line table entry. If signed, the current vertical line spacing is incremented or decremented (using a

signed number instead of a signed digit is a GNU tbl extension).

A vertical line spacing specifier followed by a column separa‐
tion number must be separated by one or more blanks. No effect
if the corresponding table entry isn't a text block.

w,W   Minimum column width value.   Must be followed either by a
troff(1) width expression in parentheses or a unitless integer.
If no unit is given, en units are used. Also used as the de‐
fault line length for included text blocks.   If used multiple
times to specify the width for a particular column, the last en‐
try takes effect.

x,X   An expanded column. After computing all column widths without
an x specifier, use the remaining line width for this column.
If there is more than one expanded column, distribute the re‐
maining horizontal space evenly among the affected columns (this
is a GNU extension). This feature has the same effect as speci‐
fying a minimum column width.

z,Z   Ignore the corresponding column for width-calculation purposes,
this is, don't use the fields but only the specifiers of this
column to compute its width.

A number suffix on a key character is interpreted as a column separa‐
tion in en units (multiplied in proportion if the expand option is on ?
in case of overfull tables this might be zero). Default separation is
3n.

The column specifier x is mutually exclusive with e and w (but e is not
mutually exclusive with w); if specified multiple times for a particu‐
lar column, the last entry takes effect: x unsets both e and w, while
either e or w overrides x.

Table data

The format lines are followed by lines containing the actual data for
the table, followed finally by .TE. Within such data lines, items are
normally separated by tab characters (or the character specified with
the tab option). Long input lines can be broken across multiple lines
if the last character on the line is ?\? (which vanishes after concate‐

nation).

Note that tbl computes the column widths line by line, applying \w on each entry which isn't a text block. As a consequence, constructions like

```
.TS
c,l.
\s[20]MM
MMMM
.TE
```

fail; you must either say

```
.TS
cp20,lp20.
MM
MMMM
.TE
```

or

```
.TS
c,l.
\s[20]MM
\s[20]MMMM
.TE
```

A dot starting a line, followed by anything but a digit is handled as a troff command, passed through without changes. The table position is unchanged in this case.

If a data line consists of only ?_? or ?=?, a single or double line, respectively, is drawn across the table at that point; if a single item in a data line consists of only ?_? or ?=?, then that item is replaced by a single or double line, joining its neighbours. If a data item consists only of ?_? or ?\=?, a single or double line, respectively, is drawn across the field at that point which does not join its neigh? bours.

A data item consisting only of ?\Rx? (?x? any character) is replaced by repetitions of character ?x? as wide as the column (not joining its

neighbours).

A data item consisting only of ?\^? indicates that the field immedi? ately above spans downward over this row.

Text blocks

A text block can be used to enter data as a single entry which would be too long as a simple string between tabs. It is started with ?T{? and closed with ?T}?. The former must end a line, and the latter must start a line, probably followed by other data columns (separated with tabs or the character given with the tab global option).

By default, the text block is formatted with the settings which were active before entering the table, possibly overridden by the m, v, and w tbl specifiers. For example, to make all text blocks ragged-right, insert .na right before the starting .TS (and .ad after the table). If either ?w? or ?x? specifiers are not given for all columns of a text block span, the default length of the text block (to be more precise, the line length used to process the text block diversion) is computed as L?C/(N+1), where ?L? is the current line length, ?C? the number of columns spanned by the text block, and ?N? the total number of columns in the table. Note, however, that the actual diversion width as re? turned in register \n[dl] is used eventually as the text block width. If necessary, you can also control the text block width with a direct insertion of a .ll request right after ?T{?.

Miscellaneous

The number register \n[TW] holds the table width; it can't be used within the table itself but is defined right before calling .TE so that this macro can make use of it.

tbl also defines a macro .T# which produces the bottom and side lines of a boxed table. While tbl does call this macro itself at the end of the table, it can be used by macro packages to create boxes for multi-page tables by calling it within the page footer. An example of this is shown by the -ms macros which provide this functionality if a table starts with .TS H instead of the standard call to the .TS macro.

tbl(1) should always be called before eqn(1) (groff(1) automatically takes care of the correct order of preprocessors).

GNU TBL ENHANCEMENTS

There is no limit on the number of columns in a table, nor any limit on the number of text blocks. All the lines of a table are considered in deciding column widths, not just the first 200. Table continuation (.T&) lines are not restricted to the first 200 lines.

Numeric and alphabetic items may appear in the same column.

Numeric and alphabetic items may span horizontally.

tbl uses register, string, macro and diversion names beginning with the digit 3. When using tbl you should avoid using any names beginning with a 3.

GNU TBL WITHIN MACROS

Since tbl defines its own macros (right before each table) it is neces? sary to use an ?end-of-macro? macro. Additionally, the escape charac? ter has to be switched off. Here an example.

```
.eo
.de ATABLE ..
.TS
allbox tab(;);
cl.
\$1;\$2
.TE
...
.ec
.ATABLE A table
.ATABLE Another table
.ATABLE And "another one"
```

Note, however, that not all features of tbl can be wrapped into a macro because tbl sees the input earlier than troff. For example, number formatting with vertically aligned decimal points fails if those num? bers are passed on as macro parameters because decimal point alignment is handled by tbl itself: It only sees ?\$1?, ?\$2?, etc., and there?

fore can't recognize the decimal point.

BUGS

You should use .TS H/.TH in conjunction with a supporting macro package for all multi-page boxed tables. If there is no header that you wish to appear at the top of each page of the table, place the .TH line im? mediately after the format section. Do not enclose a multi-page table within keep/release macros, or divert it in any other way.

A text block within a table must be able to fit on one page.

The bp request cannot be used to force a page-break in a multi-page ta? ble. Instead, define BP as follows

```
.de BP
.  ie '\\n(.z'' .bp \\$1
.  el \!.BP \\$1
..
```

and use BP instead of bp.

Using \a directly in a table to get leaders does not work (except in compatibility mode). This is correct behaviour: \a is an uninterpreted leader. To get leaders use a real leader, either by using a control A or like this:

```
.ds a \a
.TS
tab(;);
lw(1i) l.
A\*a;B
.TE
```

A leading and/or trailing ?|? in a format line, such as

```
|l r|.
```

gives output which has a 1n space between the resulting bordering ver? tical rule and the content of the adjacent column, as in

```
.TS
tab(#);
|l r|.
left column#right column
```

```
.TE
```

If it is desired to have zero space (so that the rule touches the con?

tent), this can be achieved by introducing extra ?dummy? columns, with

no content and zero separation, before and/or after, as in

```
.TS
tab(#);
r0|l r0|l.
#left column#right column#
.TE
```

The resulting ?dummy? columns are invisible and have zero width; note

that such columns usually don't work with TTY devices.

REFERENCE

Lesk, M.E.: "TBL ? A Program to Format Tables". For copyright reasons

it cannot be included in the groff distribution, but copies can be

found with a title search on the World Wide Web.

SEE ALSO

groff(1), troff(1)