Yet Another Table

Ancient table macro reworked and extended for ConTEXt

Abstract

Described is a module for the typesetting of tables. The module resembles the LaTeX `tabular` environment but is in fact based on a much older package, the origins of which are lost to the author.

Keywords

ConTeXt, table

1 Introduction

My ConTEXt `hvdm-tbx` module for tables has been long in production. It originated when the author began to feel himself somewhat uncomfortable with the LaTeX `tabular` environment for tables. Nothing bad said about LaTeX, just a personal feeling. Developing a private solution for tables seemed a nice way to make a module that is easy to use from my point of view and at the same time offered a chance to implement everything I felt necessary and/or convenient. In order to avoid clashes with the macro naming of other packages, the calling name has been made `tablex` instead of just simply `table`. Look at the trailing 'x' as something 'extra'. As is said in the abstract, the first ideas about how to typeset a table and the earliest lines of code came from an old macro package. So old in fact, that I have sought in vain to find it either among my old backups or on the internet. But an hommage to the unknown author of that package may not be lacking here.

2 Table structure

The contents of a table must be enclosed in macros `\starttablex` and `\stoptablex`. The first argument of the opening macro denotes the column structure, the second is optional and can be used to specify tablewide settings. The cells in the first column of the example are aligned left, those in the second column are centered and those in the third are put to the right. The optional argument in this example specifies the text style. The last row ends with `\nr`. It will be explained later that this ends the row without placing a visible separator. Separating the columns is done with `&` and `|`. The whole table is enclosed in a frame.

```
\usemodule[hvdm-tbx]
\starttablex[lcr][style=italic]
  \mulcols[3]{tablex}\cr
  1 & 2 & 3 \nr
  cell-1 | cell-2 | cell-3\nr
\stoptablex
```

tablex		
1	*2*	*3*
cell-1	*cell-2*	*cell-3*

After typesetting a table, its dimensions become available in the three macros `\tablexwidth`, `\tablexheight` and `\tablexdepth`.
In the example above these are:
width=93.94391pt
height=34.99997pt
depth=0.0pt.
Having access to these values is useful when for example the width of a table is needed for refined placement.

Tables can be adorned with a separate title, placed either above or below the table proper. See the following example.

```
\usemodule[hvdm-tbx]
\starttablex[lcr]
  [title=Title for this table,
   titlestyle=italic,bgcolor=lightgray]
  \mulcols[3]{tablex}\cr
  cell-1 | cell-2 | cell-3\nr
\stoptablex
```

tablex		
cell-1	cell-2	cell-3

Title for this table

3 Two types of cells

The table cells can be of two kinds. The first kind is cells where the width of the columns is dynamic, i.e. the widest cell determines the width of its column. Cells with fixed dimensions are the second kind of cell provided by the module. In this case the cell dimensions will be specified in the optional second argument of `\starttablex`.

In both cases it is the first argument where the alignment of the cells in the corresponding column is specified. The possibilities differ for the two kinds of cells. First the three usual cases of left, right and centered horizontal alignment with no variation in the vertical direction. These are for columns with dynamic width.

- `l` = horizontal left, vertical at baseline
- `c` = horizontal centered, vertical at baseline
- `r` = horizontal right, vertical at baseline

An example of text placement for all three of the above options follows. In the table on the left the standard `\strut` is placed in each cell after the text, on the right there is no strut at all. It is possible to suppress these cellbound struts in favour of the one strut that is normally placed at the end of each row. But this may have irregular baselines in adjacent cells as a consequence, depending on the exact cell content. See lateron for a demonstration of this effect.

```
\starttablex[lcr][strut=yes]
  left | center | right\cr
  a | b | q\nr
\stoptablex
\quad
\starttablex[lcr][strut=no]
  left | center | right\cr
  a | b | q\nr
\stoptablex
```

left	center	right
a	b	q

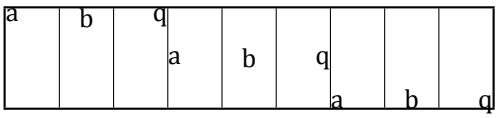
left	center	right
a	b	q

When the cell dimensions have fixed values, the following alignments can be chosen.

- `1` = horizontal left, vertical at top
- `2` = horizontal centered, vertical at top
- `3` = horizontal right, vertical at top
- `4` or `L` = horizontal left, vertical centered
- `5` or `C` = horizontal centered, vertical centered
- `6` or `R` = horizontal right, vertical centered
- `7` = horizontal left, vertical at bottom
- `8` = horizontal centered, vertical at bottom
- `9` = horizontal right, vertical at bottom

The width and height of the cell can be separately specified, its depth is always set to zero. The nine columns in the next example show all the possibilities for the placement of cell contents.

```
\starttablex[123456789]
  [cellwidth=2em,cellheight=8ex,strut=no]
  a | b | q | a | b | q | a | b | q\nr
\stoptablex
```



The placement of contents within cells is a tricky business for cells of fixed dimensions. The different heights caused by the ascender of the b and descender of the q play havoc with the baselines when forced to the top of the cell (options 1, 2 and 3). The baselines in the three cells on the left do not line up as one should expect. On the right the wobbling of the baselines which is caused by the descender of the q, has been neutralized by setting the depth of its containing box to zero. But of course that letter now descends below the frame. In the middle things are looking better.

Various strategies can be used to solve the problem demonstrated above, but there is no silver bullet. Here the following behaviour is chosen. All contents has the depth of its natural box set to zero. Then the vertical position varies as follows.

- Options 1, 2 and 3 have the top of the natural box placed directly against the top of the cell with a `\vfil` at the bottom.
- Options 3, 4 and 5 have the natural box placed between `\vfil`'s.
- Options 7, 8 and 9 have the baseline of the natural box placed directly against the bottom of the cell with a `\vfil` above.

One may dislike that strategy, but it works out

well when all contents has the same height and no depth. See the next example.

```
\starttablex[123456789]
  [cellwidth=2em,cellheight=8ex,strut=no]
  1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9\nr
\stoptablex
```

1	2	3	4	5	6	7	8	9

The solution to the first example is to use a strut adjacent to the item in each cell. This can be effectuated by turning on the `strut` parameter, as is the default behaviour. A much more pleasing table results.

```
\starttablex[123456789]
  [cellwidth=2em,cellheight=8ex,strut=yes]
  a | b | q | a | b | q | a | b | q\nr
\stoptablex
```

a	b	q	a	b	q	a	b	q

If that is not enough, an inset is put around the items in the cell by specifying the `celloffset` parameter. It can be given a specific value or most conveniently the depth of the strut in use.

```
\starttablex[123456789]
  [cellwidth=2em,cellheight=8ex,
  strut=yes,celloffset=strut]
  a | b | q | a | b | q | a | b | q\nr
\stoptablex
```

a	b	q	a	b	q	a	b	q

The following example demonstrates that a missing `cellheight` parameter defaults to a square cell. The `cellwidth` better be present, because it defaults to zero.

```
\starttablex[123456789]
  [cellwidth=2em,
  strut=no,celloffset=2pt]
  1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9\nr
\stoptablex
```

1	2	3	4	5	6	7	8	9

Note that it is possible to mix within a table columns with fixed and dynamic widths. The cells with dynamic width will then have their contents put on the same baseline as those of variants 7–9.

```
\starttablex[13c79]
  [cellwidth=2em,strut=no]
  1 | 3 | c | 7 | 9\cr
  1 | 3 | cccgccc | 7 | 9\nr
\stoptablex
```

1	3	c	7	9
1	3	cccgccc	7	9

In the above example the effect of a dynamic column between otherwise fixed cell columns is demonstrated. On close examination its is apparent that the descender on the letter g has an effect on the height of the cells in its row: they acquired a somewhat larger height.

It is even possible to specify the width of individual columns. This is done by affixing the column number to the `cellwidth` parameter. In the following example the second and third columns have been given a different width from the standard.

```
\starttablex[5555][cellwidth=2em,
  cellwidth2=4em,cellwidth3=5em]
  1   | 2   | 3   | 4   \cr
  2em | 4em | 5em | 2em \nr
\stoptablex
```

1	2	3	4
2em	4em	5em	2em

The height of rows can be adapted on the fly. Do this preferably directly behind a separator. With `\rowheight[dimension]` the size of one row is changed. The next row will revert to

the previous value. For a permanent change use `\Rowheight[dimension]`.

```
\starttablex[CCC]
  [cellwidth=2cm,cellheight=6mm]
  6mm|6mm|6mm\cr\rowheight[3mm]
  3mm|3mm|3mm\cr
  6mm|6mm|6mm\cr\Rowheight[3mm]
  3mm|3mm|3mm\cr
  3mm|3mm|3mm\nr
\stoptablex
```

6mm	6mm	6mm
3mm	3mm	3mm
6mm	6mm	6mm
3mm	3mm	3mm
3mm	3mm	3mm

4 Placement

Tables are just hboxes and thus can be put besides one another or aligned left or right or centered on the line. The optional parameter `align` governs the alignment. Setting `align` = `left`, `middle` (default) and `right` puts them into the corresponding alignment, value `none` leaves the hbox containing the table as is. The following examples show the effect of the first three options.

left

middle

right

The vertical placement can be changed with the `location` parameter, having values `top`, `middle` (default) and `bottom`. A feature needed to place tables of varying height on a line, lined up as required. The first two tables in the next example are aligned in the `middle`, the third and fourth table are typeset with `location` = `top/bottom`.

middle

middle
middle

top

bottom

Apart from the placement of the table as a whole, there is the placement of contents in the cells. The first controlling item in a column is the alignment selector of the template. But one can force another alignment for individual cells. In the example below the natural alignment of the template row is changed to left, center and right in the subsequent rows. This is accomplished by enclosing the contents of the cells in `\l`, `\c`, `\r`, respectively.

: l-template	c-template	r-template
natural : left	center	right
\l : left	center	right
\c : left	center	right
\r : left	center	right

Often the contents of the cells is simple and short. But now and then more information must be placed which doesn't fits on a short line. In that case one can put it in the cell inside a vertical box. Placement is effected with the macros `\v` and `\t` for respectively a `\vbox` and a `\vtop`. The next example demonstrates the different vertical line up for `\vbox` and `\vtop` placement.

```
\starttablex[ccc]
  vbox|vbox|vtop\cr
  \v{\hbox{verticalvertical}\hbox{stuff}}|
  \v{\hbox{vertical}\hbox{stuff}
    \hbox{vertical}\hbox{stuff}}|
  \t{\hbox{vertical}\hbox{stuff}}\cr
  vbox|vbox|vtop\nr
\stoptablex
```

vbox	vbox	vtop
verticalvertical stuff	vertical stuff vertical stuff	vertical stuff
vbox	vbox	vtop

These macros are of limited use in the case of fixed cell dimensions. Here the alignment is not perfect, because the recalculation of row height has no effect on the placement within cells earlier in the row. The example clearly demonstrates this: the contents of the first cell of the middle row is not vertically centered, in contrast to that of the third one. Furthermore the width of the first column does not adapt to the width of the box and its contents stick out on the right .

```
\starttablex[CCC][cellwidth=15mm,
  cellheight=5mm]
  < same as previous example >
```

vbox	vbox	vbox	vtop
verticalvertical stuff	vertical stuff vertical stuff	vertical stuff	vertical stuff
vbox	vbox	vbox	vtop

5 Frames

The module provides for three tailormade table-frame options. By default the frame has the type indicator `alternative` set to the value `normal`. It is a table with or without an outside frame. That outside frame is switched on and off by the parameter `frame`, having values `on/off` with `yes/no` as synonyms. The two other predefined types are the `open` and `opendouble` variants, chosen with parameter `alternative=open` and `alternative=opendouble`, respectively. The appearance of the outer frame in these two is obtained using the `before=` and `after=` optional parameters.

on	
1	2
3	4

off	
1	2
3	4

open	
1	2
3	4

opendouble			
1	2	3	4
5	6	7	8

A somewhat different fourth variant is the `\shortstack`. I must admit, a LaTeX-ism, for a table of one column. It has three argument of which the first two are optional. The first argument gives the alignment of the cells, the default is `c`. The second argument is put into the optional argument section of the table, after shutting the outer frame off. The third argument is the contents of the table. For convenience the `\nr` row separator can be typed as `\\`. This is done with the `command` parameter through which this meaning is assigned to the macro `\\`.

```
\shortstack[l][style=mono]
{long\\longer\\longest}
```

long
longer
longest

6 Rows

Rows can be separated by drawn rules: thin, thick or of a given thickness, and by invisible separators. Some of the separators place the strut currently in force at the end of the row, others don't – see below for who does what. The corresponding macros are:

- `\cr` rule of standard thickness (default value is `\linewidth`) and strut.
- `\CR` does as `\cr` with rule of greater thickness (default is `2\linewidth`).
- `\crule[dimension]` does as `\cr` with rule of specified thickness.
- `\crr`, `\CRR`, `\nrule[dimension]` same as the above, no strut placed at the end of the row; but take notice of the fact that because of the `strut=always` setting all cells could have received a strut anyway.
- `\nr`, `\nrr` typeset a rule with zero height, respectively with and without placing a strut.
- parameter `rulethickness=dimension` sets the thickness of the `\cr` and the `\nr` separator.
- parameter `framethickness=dimension` sets the thickness of the surrounding frame and the `\CR` separator.

\cr
\CR
\crule[1mm]
\nrr \nr

Further variations are the placement of a gap between two rows. The first variant `\rowsep` keeps the surrounding frame intact. The alternative is `\rowskip` which generates a break in the frame. A `\cr` after the skip places a visible separator rule, a following `\nr` an invisible one. Note the use of the separators `\crr` and `\nrr` instead of `\cr` and `\nr` in places where the strut spanning up a table row must be suppressed.

first row
rule before and after \rowsep \cr\rowsep[1mm]\crr
rule before and after \rowskip \cr\rowskip[1mm]\crr
rule before not after \rowskip \cr\rowskip[1mm]\nrr
last row

It is possible to place partial rules of various heights. The general macro is `\ccrule[#][height]`.

Specialized variants are \crrule, \CRrule and \nrrule on which the height argument is implicit. Their function is readily understood by comparison with the rule names above. Position these macros as if they were items in a cell, i.e. they make a pseudo row in the table. So they need to be separated with all variants of &, |, etc.

One caveat applies. Do not forget to end the row with \nr and do this with the previous row as well. Otherwise the rule that is typeset will interfere with the partial rules. Note how for columns 3–5 the vertical bars between the columns have been kept intact through judicious choice of the partial rules and the separators between them. Compare these with columns 5 and 6. Macro \norule is a convenient shortcut for \nrrule[1].

```
\starttablex[cccccccc]
  1|2|3|4\db5|6|7|8\nr
  \ccrule[2][2pt]|\norule|\norule\db
    \nrrule[2]&\ccrule[2][2pt]\nr
  1|2|3|4\db 5|6|7|8\nr
\stoptablex
```

1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8

7 Columns

Columns are formed by placing separators between the items in a table row. The simplest are & and |, shortcuts for \tb and \vb respectively. The first one places an invisible separator, the second a vertical bar of the width defined by parameter rulethickness. Around these separators horizontal whitespace is put, offsetting the contents of the table cell from the column separators. The size of the surrounding white can be set with parameter columnsep. In the example below the zero value in the second table forces the cell contents tight against the separators.

```
\starttablex[rl][rulethickness=2pt]
  right & left\nr
  right | left\nr
\stoptablex
\quad
\starttablex[rl]
  [rulethickness=2pt,columnsep=0pt]
  right & left\nr
  right | left\nr
\stoptablex
```

right	left
right	left

right	left
right	left

There are several separators to choose from.

- \tb or & is an invisible separator.
  In the form \tb[separator] a custom separator is placed.
- \vb or | is a vertical bar of size rulethickness.
- \VB is a vertical bar of size framethickness.
- \vbsep[dimension] is a vertical bar of given size.
- \db is a double vertical bar of size rulethickness separated by whitespace rulesep.
- \DB is like db but has size framethickness.
- \dbsep[dimension] is a double vertical bar of given size.

The example below has the separator which is used named at its left. Note that between the last two columns in the second row the \tb separator has been used to place a custom separator, an equals sign in this case.

```
\starttablex[cccccc]
  [rulesep=2pt,framethickness=2pt]
  1\vb 2\VB 3\db 4\DB 5 &6\nr
  vb\vb VB\VB db\db DB\DB
  $3\times4$\tb[=] $12$\nr
\stoptablex
```

1	2	3	4	5	6
vb	VB	db	DB	3×4	= 12

A second equivalent series of separators has variable whitespace before and after it.

- \tbs[size][before][after] is an invisible separator with the given amounts of whitespace before and after, a missing after defaults to the same space as is put before.
- \vbs is the equivalent of \vb.
- \VBS is the equivalent of \VB.
- \vbssep is the equivalent of \vbsep.
- \dbs is the equivalent of \db
- \DBS is the equivalent of \DB.
- \dbssep is the equivalent of \dbsep.

8 Skipping and combining columns

One can combine columns and skip columns. Skip-

ping is provided by two macros. A specified number of columns is skipped with `\skipcols[#]`. The remaining columns in a row may be skipped all at once with `\skipall`. An example follows.

```
\starttablex[ccccccccc]
1|2|3|4|5|6|7|8|9\cr
1|\skipcols[2]|3|4|\skipcols[3]|9\cr
1|2|\skipall\cr
1|2|3|4|5|6|7|8|9\nr
\stoptablex
```

1	2	3	4	5	6	7	8	9
1			3	4				9
1	2							
1	2	3	4	5	6	7	8	9

Combining columns is done with the `\mulcols` macros. There are five variants, all combine the number of columns given in the argument. They do not adapt the total width of the columns spanned. Therefore a long text may stick out of the available space.

- `\cmulcols[#]` or `\mulcols[#]` has its contents centered.
- `\\mulcols[#]` contents forced to the left.
- `\rmulcols[#]` contents forced to the right.
- `\\mulcols[#]` contents horizontally centered, in case of fixed cell dimensions vertically forced to the top.
- `\bmulcols[#]` contents horizontally centered, in case of fixed cell dimensions vertically forced to the bottom.

no cellwidth				
five				
five and much much too long				
1	three			5
five left				
five right				
top				
bottom				
1	2	3	4	5

cellwidth=5mm				
five				
five and much much too long				
1	three			5
five left				
five right				
top				
bottom				
1	2	3	4	5

A variant that does adapt to the width of its contents is `\Mulcols`. Note that the extra space goes into the rightmost column, which may an unwanted side effect.

Mulcols				
five and not too long anymore				
1	2	3	4	5

9 Struts

For the selection of struts the module presents three possibilities.

- Use the standard `\strut` of the font in use when typesetting starts. On `\starttablex` select this option with `strut=yes` or `strut=on`.
- Do not use a strut, i.e. make a null strut. Select this option with `strut=no` or `strut=off`.
- Define the strut yourself. To this end the values of optional parameters `strutheight` and `strutdepth` are used; they default to value 0pt. Select this one with `strut=tablex`.

By default a strut is placed in every cell, but this can be disabled for a given row from a specific cell onwards. Do this by calling the macro `\notablestrut` in the first cell of the row. At the start of the next row the strut is automatically reenabled with `\dotablestrut`. This way to disables the struts also disables the strut that is placed at the end of the row. In the example a large strut is used and disabled at the start of the second row. Use this mechanism when a specific row must be compressed in the vertical dimension.

```
\starttablex[ccccc][strut=tablex,
  strutheight=3mm,strutdepth=3mm]
  1|2|3|4|5\cr
  \notablestrut 1|2|3|4|5\cr
  1|2|3|4|5\nr
\stoptablex
```

1	2	3	4	5
1	2	3	4	5
1	2	3	4	5

A second customization of strut placement is given by the parameter `cellstrut`. The value `cellstrut=always` places a strut in every cell, while `never` disables these struts. Contrary to `\notablestrut` this will keep the strut at the end of each row alive. The effect of `never` can be seen below in the irregular baselines in the table on the right.

```
\starttablex[321][cellwidth=6mm,
  strut=yes,cellstrut=always]
  q & t & b\nr
\stoptablex
\quad
\starttablex[321][cellwidth=6mm,
  strut=yes,cellstrut=never]
  q & t & b\nr
\stoptablex
```

q	t	b

q	t	b

In the case of fixed cell dimensions the value `strutdepth` will be used as `celloffset` with the optional parameter setting `celloffset=strut`.

10 Font changes

Unless set on a call to `\setuptablex`, the font used for typesetting the table is the current font. Among the optional parameters one finds the `font` parameter which may given any value that is acceptable to `\switchtobodyfont`. If this parameter is present, it is the first to be applied.

Secondly the parameter `style` is used. It provides many alternatives, for example to change the style to italic, use a smaller font, etcetera.

Finally a parameter `fontchange[dimension]` is applied. This parameter is exclusively used for changing the size of the font. A positive dimension increases the size by that amount, a negative one diminishes it.

11 Colors

Coloring the text is simple. Just set the optional parameter `foregroundcolor` or simply `color` to the color required. Of course the general ConTEXt-macro `\color[]` can be used to change the color at will. When the color name has been made into a macro, changing the color of the whole cell is even simpler; as can be seen in the third cell.

```
\starttablex[cc][color=blue]
  blue\vb a \color[red]{red} rose\cr
  \orange orange\vb blue\nr
\stoptablex
```

blue	a red rose
orange	blue

Separators between rows and within rows get the color of the outer frame by default, i.e. the value `framecolor`.

```
\starttablex[cccc][framecolor=red]
  a\vb b\db c\vbsep[2mm]d\cr
  e\vb f\db g\vbsep[2mm]h\nr
\stoptablex
```

a	b	c	d
e	f	g	h

But all column and row separators can be colored differently by appending a color between brackets behind them. In the example below the outer frame, the separators between the rows and those between the columns all have been given a different color. Colored horizontal rules stay within the outer frame.

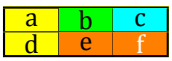
```
\starttablex[cccc][framecolor=darkgreen]
  a \VB[blue] b \DB[blue]
  c \vbsep[2mm][blue] d
  \crule[1mm][cyan]
  e \VB[red] f\DB[red]
  g \vbsep[2mm][red] h\nr
\stoptablex
```

a	b	c	d
e	f	g	h

Backgrounds may be colored when the parameter `background` has been set to the value `color`, as is the default. The optional parameter `backgroundcolor` or `bgcolor` governs the general background of the table and is white by default. Individual cells can be given a background color

by placing macro \bg[color] somewhere in the first cell to receive individual background coloring. That color remains in effect until either another color is set or an empty \bg is encountered. Set the parameter background to none or off in order to disable background coloring regardless of the color macros present. There is one restriction: coloring the background of individual cells is implemented for cells with fixed dimensions only.

```
\starttablex[CCC]
  [bgcolor=yellow,cellwidth=2em]
  a\vb\bg[green] b\vb\bg[cyan] c\cr
  \bg d\vb e\bg[orange]\vb\white f\nr
\stoptablex
```



12 Variables and setup

All parameters that can be given a value in the option argument of \starttablex can also be given an initial value with the macro

\setuptablex[..=..]

Next follows a list of variables occurring in the table typesetting. Be warned, changing them on the fly can easily work havoc on the table, because the underlying \halign may behave wildly when not treated mildly. Variables having a name starting with \tablex are internal variables. The others are settable on \setuptablex and \starttable. Note that for these a parameter named param ends up in macro \tablexparam. Be aware of the fact that changes to macros and registers inside a cell have a local effect only, it is the way \halign works. Lasting changes require the application of \global changes.

- tabsep Glue normally placed at the start and end of a row and around separators.
- location Vertical placement of the table. values: none, top, middle (default), bottom.
- align Horizontal placement of the table. values: none, left, middle (default), right.
- alternative Table style framed or open type. values: framed (default), open, opendouble.
- before Code to execute just before the table is typeset.
- after Code to execute just after the table is typeset.
- strut Select a strut. values: yes, on for the standard \strut of the font (default); table the given strutheight and strutdepth; no, off no strut.
- cellstrut governs a strut in every cell for value always (default) or never.
- strutheight Height custom strut, default 0pt.
- strutdepth Depth custom strut, default 0pt.
- rulethickness Size of standard horizontal and vertical separators.
- rulesep Distance between bars in double separator.
- columnsep Glue around separators.
- cellwidth Width of fixed dimension cells. cellwidth# Width of column number #.
- cellheight Height of fixed dimension cells, defaults to cellwidth.
- celloffset Inset for fixed dimension cells. values: dimension (default, initial 0pt) or strut for the value of strutdepth.
- color Color of text, defaults to black; may also be given as foregroundcolor.
- framecolor Color of outside frame and separators, defaults to black.
- background Enable background coloring. values: color, yes, on (default), none, no, off.
- bgcolor Color of general background, defaults to white; may also be given as backgroundcolor.
- font General font for text. If not empty executes contents within \switchtobodyfont[], defaults to empty, applied first.
- style Change style of general text font. Applied after font and before fontchange. values: bold, italic, bolditalic, italicbold, slant, slanted, boldslanted, slantedbold, smallcaps, oldstyle, mediaeval, normal, serif, regular, roman, sans, sansserif, mono, type, teletype, handwritten, calligraphic, big, small, smallmono, tiny, tinymono
- fontchange Change size of font with the amount given, defaults to empty, applied last.
- title Text of the title, if empty the title is suppressed.
- titlefont Used in fontswitch for the title.
- titlestyle Used as style setting for the title.
- titlelocation Location of the title, values are top or bottom (default); use none to suppress even if a title text has been given.
- titlecolor Color of the title.
- titledistance Distance between table and title.
- command Gets a value that is defined as macro \\, the default is \cr.
- \tablexwidth, \tablexheight,

\tablexdepth Width, height and depth of the table just typeset.
- \tablexcellwdt, \tablexcellhgt Dimen registers for width and height of the current cell in case of fixed dimensions.
- \tablexcolumns Count register for number of columns in template.
- \tablexcolumn Count register current column.
- \ifinfirstcol Tells when in first column.
- \ifdimensionsfixed Tells if fixed dimensions.
- \gettablexcellwidthnumber Resolves to width of given column when fixed dimensions.

Hans van der Meer
H.vanderMeer@uva.nl