

The multicoltab package

Andres Zanzani*

Version 0.8 – 2026-09-04

Abstract

`multicoltab` provides a non-floating table environment whose source rows can continue in the next column of a `multicols` layout or on a later page. It achieves this without replacing LaTeX’s output routine: each source row is emitted as an independent `tabular` or `tabularx`. A break can therefore occur between rows but never in the middle of one. This manual describes the public interface, the layout model, source-row parsing, measurement, execution constraints, continuation commands, and the structure of the implementation.

Contents

1 Purpose and scope	2
1.1 What the package is and is not	2
2 Installation and requirements	2
3 Quick start	3
4 Table columns and layout columns	3
4.1 Live three-column example	3
5 Execution model	4
6 Source-row syntax and parser rules	4
6.1 Row terminators	4
6.2 Cell counts	5
7 Options	5
7.1 Break penalties and headings	5
8 Column specifications	5
8.1 Natural and fixed widths	6
8.2 Modifiers, aliases, and repeated specifications	6
8.3 Spanning cells	7
9 Cell evaluation and side effects	7
10 Row formatting, rules, and colours	7
11 Headings	8
11.1 Declaring a heading	8
11.2 Repeating the last heading in place	8

*azanzani@gmail.com

12 Explicit continuation	9
12.1 Next <code>multicols</code> column	9
12.2 Next ordinary page	9
13 Implementation architecture	9
13.1 Preamble processing	10
13.2 Measurement and reconstructed preamble	10
14 Diagnostics and troubleshooting	10
15 Limitations and compatibility	11
16 Testing and maintenance	11
17 Files in the distribution	11
18 License and contact	12

1 Purpose and scope

Standard table environments are boxes: LaTeX cannot split a `tabular` at a page boundary, and `longtable` is incompatible with `multicol` because both packages need control of the output routine. `multicoltab` covers the missing case: a non-floating table or structured list that must flow through the columns created by `multicols`.

The package has a deliberately narrow contract. A source row is the smallest unbreakable unit. The package may move a complete row to the next layout column or page, but it never splits a row internally. This is suitable for glossaries, price lists, game statistics, compact catalogues, terminology, and other data where a source row is short enough to be kept together.

1.1 What the package is and is not

Use <code>multicoltab</code> when	Use another mechanism when
Rows must flow through <code>multicols</code> columns or ordinary later pages.	A conventional multi-page table needs captions, footnotes, page headers, or footers. Use <code>longtable</code> outside <code>multicols</code> .
Each source row can remain unbroken.	A single cell must break across layout columns or pages. Restructure the data or use ordinary prose.
A table must participate in normal paragraph flow rather than float.	A floating table with placement control is required. Use <code>table</code> .

2 Installation and requirements

For a local installation, place `multicoltab.sty` in the document directory. For a persistent installation, place it in a directory searched by your TeX distribution and refresh the file-name database if required.

The package requires a current LaTeX2e format with `expl3` support and loads `xparse`, `array`, `tabularx`, and `keyval`. It does *not* load `multicol`: load that package only when the document uses `multicols`.

```
\usepackage{multicol}      % Only when multicols is needed
\usepackage{multicoltab}
```

The distribution includes regression tests for pdfTeX, LuaTeX, and XeTeX. The examples and manual are ordinary LaTeX documents and may be compiled with any of those engines.

3 Quick start

The complete public environment is:

```
\begin{multicoltab}[<options>]{<column specification>}
  <rows>
\end{multicoltab}
```

Every normal row uses the familiar alignment syntax:

```
left cell & right cell \\\
```

The final row terminator is optional. The following document creates a normal two-column structured list. The table width is the current `\linewidth`, so it automatically fits one `multicols` column.

```
\begin{multicols}{2}
\begin{multicoltab}{@{}p{1.8cm}X@{}}
  Armor & Reduces the damage received by a character. \\\
  Shield & Improves a character's defense. \\\
  Torch & Provides light in dark places.
\end{multicoltab}
\end{multicols}
```

4 Table columns and layout columns

There is no two-column limitation. The number of columns in the *table* is independent of the number of columns in the surrounding *layout*. A table can have two, three, four, or more explicit columns. Likewise, `multicols` may have any column count accepted by `multicol`.

The following example has a three-column table in a two-column layout. It does not use `\multicolumn`.

```
\begin{multicols}{2}
\begin{multicoltab}{@{}lXX@{}}
  Name & First value & Second value \\\
  Armor & 2 & 5 \\\
  Shield & 3 & 6
\end{multicoltab}
\end{multicols}
```

For example, the same source form works with three or four layout columns. The supplied columns example document combines three and four table columns with three- and four-column layouts.

4.1 Live three-column example

Item	Value	Effect	Shield	5	Improves defense.
Armor	2	Reduces damage received.	Torch	1	Provides light in dark places.

5 Execution model

Understanding the execution model is important when designing tables and when placing macros with side effects in cells. The environment processes its body in three conceptual stages.

1. **Collection.** The `+b` environment argument collects the complete table body before any row is emitted. Consequently, verbatim-like commands that require normal character-by-character input scanning are not generally safe.
2. **Preparation.** The package scans the direct column tokens in the preamble. In global natural-width mode, direct natural columns `l`, `c`, and `r` are measured across the source rows. Fixed and `X` columns are not measured by `multicoltab`.
3. **Emission.** The body is parsed again and every source row is typeset as a separate alignment. A directly recognized preamble without `X` uses `tabular`; a preamble with `X`, aliases, or constructs whose expansion is unknown uses `tabularx`. LaTeX then sees ordinary vertical material between rows and may break there.

Conceptually, a table such as

```
\begin{multicoltab}{lX}  
  A & first row \\  
  B & second row  
\end{multicoltab}
```

becomes a vertical sequence resembling:

```
\begin{tabularx}{<width>}{<prepared preamble>} A & first row \\\end{tabularx}  
<row penalty and optional spacing>  
\begin{tabularx}{<width>}{<prepared preamble>} B & second row \\\end{tabularx}
```

This is the reason rows can continue across columns. Features that intrinsically span multiple ordinary alignment rows remain outside the package’s scope; the supported standalone `booktabs` rules are handled as special single items.

6 Source-row syntax and parser rules

6.1 Row terminators

At top level, `\\` ends a source row. The package also accepts the usual optional suffixes:

Source ending	Meaning
<code>\\</code>	Normal row boundary.
<code>*</code>	Normal alignment ending plus a prohibitive outer break penalty.
<code>\\[<length>]</code>	Add vertical space within the emitted alignment row.
<code>*[<length>]</code>	Combine the no-break marker and alignment row space.

The final terminator may be omitted. A `\\` inside braces is not seen as an outer row terminator; for example, a nested valid alignment may contain its own row endings. This parser protection does not make arbitrary grouped content valid inside a table cell. For an ordinary in-cell line break, use `\newline`.

The parser trims top-level surrounding spaces and removes top-level `\par` tokens. Thus paragraph breaks written inside cells are not supported; use `\newline` instead.

At top level, `\toprule`, `\midrule`, and `\bottomrule` are recognized as standalone rule items. They may use the standard `booktabs` form without a following row terminator. An explicit terminator is also accepted when a rule is written as its own source item. Optional rule-width arguments are passed unchanged to `booktabs`.

6.2 Cell counts

Rows should supply the number of cells required by their effective preamble. The final emitted `tabular` or `tabularx` remains responsible for the usual `array` diagnostics when a row contains too many cells, too few cells, or an invalid alignment construct. This design intentionally preserves familiar LaTeX error messages instead of creating a parallel cell-counting API.

7 Options

The optional environment argument is a comma-separated **keyval** list. All options are local to the environment.

Option	Default	Effect
<code>width=<length></code>	<code>\linewidth</code>	Target width passed to <code>tabularx</code> . It has an exact stretching effect only when the preamble has an <code>X</code> column or other stretchable material.
<code>row-sep=<length></code>	<code>Opt</code>	Vertical glue placed after each emitted source row.
<code>break-penalty=<integer></code>	<code>0</code>	Penalty after a normal row. Positive values discourage a break; negative values encourage one.
<code>natural-widths=global</code>	<code>global</code>	Measure direct natural columns across the body and reuse the widths in every emitted row.
<code>natural-widths=local</code>	—	Do not perform package-level natural-cell measurement; every natural column keeps its normal per-row width.

For example:

```
\begin{multicoltab}[width=.96\linewidth,row-sep=.6ex,
                    break-penalty=50]{@{}p{.3\linewidth}X@{}}
  Short label & A longer description that may wrap. \\
  Another label & Another description.
\end{multicoltab}
```

7.1 Break penalties and headings

The `break-penalty` setting applies to normal rows. A generated heading receives a prohibitive penalty, so it stays with the following row. The penalty is placed before `row-sep`; the glue therefore does not introduce a separate permitted break after the heading.

8 Column specifications

The mandatory preamble is passed to `array/tabularx` for final typesetting. `multicoltab` additionally recognizes a conservative subset of direct column tokens to decide which natural widths can be synchronized.

Token	Meaning in <code>multicoltab</code>
<code>l</code> , <code>c</code> , <code>r</code>	Natural columns. In global mode, a direct occurrence is converted to a measured fixed-width paragraph column while retaining left, centred, or right alignment.
<code>p{width}</code> , <code>b{width}</code>	Fixed-width paragraph columns supplied by <code>array</code> . They are not measured by this package.
<code>X</code>	<code>tabularx</code> paragraph column that absorbs remaining width. Multiple <code>X</code> columns use the normal <code>tabularx</code> width algorithm.
<code>w{align}{width}</code> , <code>W{align}{width}</code>	Fixed-width single-line <code>array</code> columns. Their grouped arguments are preserved.
<code>>{code}</code> , <code>@{code}</code> , <code>!{code}</code> , <code> </code>	Standard <code>array</code> modifiers and separators are preserved for final output.

8.1 Natural and fixed widths

Use `l`, `c`, and `r` for short labels or values that need a uniform natural width across rows. Use a fixed paragraph column or `X` for prose that must wrap. A typical robust preamble is:

```
@{}p{2cm}X@{}
```

The first cell has a stable label width; the second uses the rest of the available line width. Multiple `X` columns share available width:

```
\begin{multicoltab}{@{}lXX@{}}
  Name & First value & Second value \\
  Armor & 2 & 5
\end{multicoltab}
```

Relative `X` widths use the normal `\hsize` convention. The assigned relative values must sum to the number of `X` columns.

```
\begin{multicoltab}
  >{\hsize=.5\hsize}X>{\hsize=1.5\hsize}X
  Short label & A wider description column.
\end{multicoltab}
```

By default, `X` behaves as a top-aligned `p` column. The standard `tabularx` hook changes this for following `X` columns:

```
\renewcommand{\tabularxcolumn}[1]{m{#1}}
```

8.2 Modifiers, aliases, and repeated specifications

Modifiers remain available for final output. Their code is not replayed during natural-width measurement because doing so would require executing arbitrary preamble code while measuring cells. Therefore, if a modifier, a custom column alias, or a repeated `*{n}{...}` specification is combined with natural columns, the package keeps the source natural columns per-row and issues a warning instead of making an unreliable global measurement.

This is a deliberate safety boundary. If synchronized natural widths are required, write direct column tokens explicitly and avoid modifiers on those natural columns. Otherwise select local natural widths, fixed-width paragraph columns, or `X` columns.

8.3 Spanning cells

A literal `\multicolumn` at the start of a cell is supported. Its span advances the logical column position during measurement; its content does not contribute to any individual natural-column maximum because there is no unique way to distribute a spanning width among the covered columns.

```
\begin{multicoltab}{111}  
  \multicolumn{2}{c}{Grouped values} & Total \\  
  Armor & Shield & 7  
\end{multicoltab}
```

Do not hide a spanning command inside an arbitrary macro when global natural width measurement matters. A row beginning with `\rowcolor` followed by `\multicolumn` is typeset correctly, but global natural-width synchronization is conservatively disabled for that table and a warning is issued.

9 Cell evaluation and side effects

This section is critical for macros that modify document state. The package does not measure contents in `X`, `p`, `m`, `b`, `w`, or `W` columns. In global mode it measures direct natural `l`, `c`, and `r` cells once in a box before final output.

For a directly recognized preamble without `X`, rows are emitted with `tabular`. Combining this with `natural-widths=local` gives one package-level execution of each cell. However, a preamble containing `X` uses `tabularx`. `tabularx` may typeset its body repeatedly while it solves the width of its `X` columns. This is standard `tabularx` behaviour and cannot be generically eliminated while retaining its width algorithm.

Do not place arbitrary global side effects in cells of a table containing `X`. In particular, calculate counters, random values, file writes, index entries, or custom global assignments before the environment and insert the already calculated value in the cell.

```
% Preferred: calculate state before the table.  
\stepcounter{example}  
\edef\examplevalue{\arabic{example}}  
\begin{multicoltab}{lX}  
  Example & \examplevalue  
\end{multicoltab}
```

LaTeX counters and writes receive some protection from `tabularx` trial typesetting, but arbitrary package state does not. Treat side-effect-free cell contents as the portable rule.

10 Row formatting, rules, and colours

No wrapper is required for ordinary row formatting. Place normal table commands directly before the row cells. When `xcolor` is loaded with its `table` option, `\rowcolor` colours the following source row.

```
Potion & Restores a small number of hit points. \\  
\rowcolor{gray!20}  
Rope & Useful for climbing and tying equipment. \\  
Map & Helps the group avoid getting lost.
```

There are two supported ways to use the three main `booktabs` rules. Use standalone rules when the heading is not stored or repeated. The usual `booktabs` syntax is accepted:

```

\toprule
Header & Value \\
\midrule
Entry & Description \\
\bottomrule

```

Standalone rules are emitted as empty alignments. This preserves the `booktabs` rule spacing without passing a source-row terminator after the rule's `\noalign` material. Optional rule widths are accepted as usual.

Use the optional before/after arguments of `\mthead` when the heading and its rules must be stored and printed again after an explicit continuation. `\mthead` also keeps that heading with its first data row. It is not needed merely to draw a rule.

11 Headings

11.1 Declaring a heading

`\mthead` records and prints a heading. Its syntax is:

```
\mthead[<before>][<after>]{<cells>} \\
```

The cell material is always the mandatory braced argument. A command that formats the heading row, such as `\rowcolor`, must be placed inside that argument rather than between the optional arguments and the cell material. For `\rowcolor`, load `xcolor` with its `table` option: `\usepackage[table]{xcolor}`.

```

\mthead[\toprule][\midrule]{%
  \rowcolor{gray!20}%
  \textbf{Patron} & \textbf{Archetypal resonances}%
} \\
Patron A & Description

```

The optional before/after material is inserted around the heading cells inside the emitted alignment. This is the heading-aware form of rule usage:

```

\mthead[\toprule][\midrule]
  {\textbf{Item} & \textbf{Description}} \\
Armor & Reduces damage. \\
Shield & Improves defense.

```

The heading has a prohibitive outer break penalty and remains with its next emitted data row. A heading command must be a complete source row.

11.2 Repeating the last heading in place

`\mtheadrepeat` prints the most recently declared heading again without forcing a column or page break. It is useful for a visible manual subdivision:

```

\mthead{\textbf{Item} & \textbf{Description}} \\
Armor & Reduces damage. \\

\mtheadrepeat \\
Torch & Provides light.

```

It is an error to use `\mtheadrepeat` before an earlier `\mthead`.

12 Explicit continuation

`multicol` decides automatic balancing and column breaks only after it receives vertical material. It cannot reliably call back into a table to insert a repeated heading at every arbitrary automatic break. When a repeated heading is mandatory, use an explicit continuation directive.

12.1 Next multicol's column

Inside `multicols`, `\multicoltabbreak` forces the next layout column and repeats the declared heading:

```
\begin{multicols}{2}
\begin{multicoltab}{@{}p{1.8cm}X@{}}
  \mhead{\textbf{Item} & \textbf{Description}} \\
  Armor & Reduces damage. \\
  \multicoltabbreak \\
  Torch & Provides light.
\end{multicoltab}
\end{multicols}
```

12.2 Next ordinary page

Outside `multicols`, `\multicoltabpagebreak` forces a new page and repeats the heading:

```
\begin{multicoltab}{@{}p{1.8cm}X@{}}
  \mhead{\textbf{Item} & \textbf{Description}} \\
  Map & Helps avoid getting lost. \\
  \multicoltabpagebreak \\
  Rations & Food and water for the journey.
\end{multicoltab}
```

Both continuation commands require a preceding `\mhead`, must be complete source rows, and are executed only in the emission pass. Do not use `\multicoltabpagebreak` inside `multicols`; use `\multicoltabbreak` there instead.

13 Implementation architecture

The public environment is intentionally small, but the implementation has four separate responsibilities. This overview is useful for maintainers and for users diagnosing boundary cases.

Component	Responsibility
Environment setup	Initializes keys, captures the source preamble, and scopes changes with a group.
Preamble scanner	Classifies direct column tokens, records whether natural widths are safe to synchronize, and chooses <code>tabular</code> or <code>tabularx</code> .
Measurement pass	Splits normal rows at top-level <code>&</code> , skips fixed and <code>X</code> cells, records maximum widths for direct natural cells, and advances over literal <code>\multicolumn</code> spans.
Row parser and emitter	Reads top-level row terminators, recognizes heading and continuation directives, emits an independent alignment for each row, and supplies the outer break penalty and spacing.

The implementation is contained in `multicoltab.sty`. Internal control sequence names use the `multicoltab@` prefix or expl3 names beginning with `l_multicoltab_`. They are implementation details, not a public API. Documents should use only the environment, the four marker commands `\mthead`, `\mtheadrepeat`, `\multicoltabbreak`, and `\multicoltabpagebreak`, and the three supported `booktabs` rule commands.

13.1 Preamble processing

The scanner intentionally does not expand custom column types. Expansion could run arbitrary user code and would make a general parser dependent on private `array` internals. Direct `l`, `c`, `r`, `X`, `p`, `m`, `b`, `w`, and `W` tokens are recognized, along with the argument groups of the standard separators and modifiers. Unknown constructs cause the conservative fallback described in Section 14.

13.2 Measurement and reconstructed preamble

In global mode, each direct natural token is reconstructed as a fixed-width paragraph column. The width is the maximum measured cell width plus a small allowance. That allowance avoids a word whose natural width is exactly the measured maximum being immediately reconsidered for hyphenation in the paragraph box. Centre and right alignment are recreated with the usual `\centering`/`\raggedleft` and `\arraybackslash` pattern.

Fixed columns retain their source specification, and `X` retains normal `tabularx` behaviour. If the preamble is not safe to measure, natural columns also retain their source form and the warning below is issued.

14 Diagnostics and troubleshooting

Message or symptom	Cause and response
Natural-column widths cannot be synchronized for this preamble	The preamble contains a modifier, alias, repetition, or another construct that the safe direct-token scanner does not measure. Use fixed/ <code>X</code> columns, write direct tokens explicitly, or accept local natural widths.
Heading repeated before it was declared	Place <code>\mthead</code> before <code>\mtheadrepeat</code> , <code>\multicoltabbreak</code> , or <code>\multicoltabpagebreak</code> .
<code><command></code> used outside <code>multicoltab</code>	Heading and continuation markers are parser directives, valid only as complete rows in the environment.
Repeated counters or custom global changes	A table uses <code>X</code> , so <code>tabularx</code> trial typesetting is evaluating cell material. Calculate the value before the table.
Misplaced <code>\noalign</code> from a rule command	A rule command was used in a form not supported by the row parser. Use standalone <code>\toprule</code> , <code>\midrule</code> , or <code>\bottomrule</code> rows, or put heading rules in the optional arguments of <code>\mthead</code> .

15 Limitations and compatibility

- A row cannot break internally. A row taller than the available layout area is moved as a whole and may still overflow an exceptionally small area.
- `\multirow` across source rows is not supported. Every source row is a separate alignment.
- Floats, captions, and footnotes retain the normal restrictions of `multicol`. This package does not add float or longtable features.
- The body is collected before rows are emitted. Avoid `\verb` and similar verbatim input commands in cells unless another package supplies a verified verbatim-safe mechanism.
- Paragraph breaks written as `\par` in cells are removed by the row parser. Use `\newline` for a line break within a cell.
- Automatic repeated headings at arbitrary `multicol` breaks are not available. Use an explicit continuation directive when repetition is required.
- A direct `\multicolumn` is handled during measurement; its spanning content does not set individual natural widths.
- Standalone rules are supported for `\toprule`, `\midrule`, and `\bottomrule` from `booktabs`. Other `booktabs` commands and `\hline`/`\cline` rows are not supported.

16 Testing and maintenance

The distribution contains `build.lua` and an `l3build` regression test in `testfiles/regression.lvt`. The test suite covers fixed and X measurement avoidance, global natural measurement, local execution without X, multi-column and spanning rows, colours, preamble forms, row-ending suffixes, standalone `booktabs` rules, explicit page continuation, and heading protection.

From the package directory, run:

```
l3build check
l3build doc
```

The first command executes the regression suite with pdfTeX, LuaTeX, and XeTeX. The second compiles this manual and every listed example. A release archive should contain the package file, README, license, manual source and PDF, examples and their PDFs, changelog, build configuration, and regression test sources; it should not contain TeX auxiliary files such as `.aux`, `.log`, or `.synctex.gz`.

17 Files in the distribution

File	Purpose
<code>multicoltab.sty</code>	Package implementation.
<code>multicoltab-doc.tex</code>	Source of this technical manual.
<code>multicoltab-doc.pdf</code>	Compiled manual.
<code>README.md</code>	Short overview and CTAN-facing quick reference.
<code>CHANGELOG.md</code>	Release history.
<code>LICENSE</code>	LPPL license notice.
<code>*-example.tex</code>	Standalone examples.
<code>*-example.pdf</code>	Compiled example output.
<code>build.lua</code> and <code>testfiles/</code>	<code>l3build</code> configuration and tests.

18 License and contact

This work is released under the LaTeX Project Public License, version 1.3c or later. A license notice is included in LICENSE; the full text is at <https://www.latex-project.org/lppl/>.

Questions, bug reports, and suggestions may be sent to azanzani@gmail.com.