Package: lt (via r-universe)

June 6, 2026

Type Package

Title Lightweight Tables via JSON Specs and JavaScript

Version 0.0.10

Description A lightweight grammar of tables. Build a table by declaring a JSON spec (titles, spanners, row groups, footnotes, formatters, etc.); a tiny vanilla JavaScript runtime builds the HTML table from the spec on page load. No 'sass', no 'V8', no 'htmlwidgets' — just base R and 'xfun' ('htmltools' is used only for the optional Shiny binding).

License MIT + file LICENSE

URL https://github.com/yihui/lt

BugReports https://github.com/yihui/lt/issues

Encoding UTF-8

Imports xfun (>= 0.57)

Suggests htmltools, knitr, litedown, repr, shiny, testit

Roxygen list(markdown = TRUE)

Config/lt.js 1.14.51

Config/roxygen2/version 8.0.0

Repository https://yihui.r-universe.dev

Date/Publication 2026-06-06 01:23:21 UTC

RemoteUrl https://github.com/yihui/lt

RemoteRef HEAD

RemoteSha d5b62cdc6fc1ca988ab6039c197134e4d57da3a8

Contents

`format.lt_tbl` *Render an* `lt_tbl` *to HTML*

Description

Emits the CSS+JS runtime and a script block carrying the table's JSON spec. Multiple tables on the same page only need the runtime once.

Usage

```
## S3 method for class 'lt_tbl'
format(x, fragment = TRUE, inline_assets = TRUE, assets = TRUE, ...)
```

Arguments

`x`	An `lt_tbl` object.
`fragment`	If `TRUE` (default), return an HTML fragment suitable for embedding. If `FALSE`, wrap in a minimal `<html><body>` document.
`inline_assets`	If `TRUE` (default), inline the CSS/JS as text. If `FALSE`, emit `<link>` / `<script src=...>` tags (assets must be served alongside the HTML).
`assets`	If `TRUE` (default), include the CSS+JS runtime. Pass `FALSE` to emit only the spec block when the runtime is already on the page.
`...`	Reserved for future use.

Value

A character scalar containing HTML.

lt *Create a Table Specification*

Description

Entry point of the lightweight grammar of tables. Returns an object (a list) that records the data plus a list of table-modifying operations. The object is rendered to HTML by `format()` (called automatically by the print method).

Usage

```
lt(data, ...)

## Default S3 method:
lt(data, auto_fmt = TRUE, ...)
```

Arguments

`data`	A data frame (or anything coercible to one).
`...`	Arguments passed to methods.
`auto_fmt`	Whether to automatically format numeric columns (rounding, thousand separators, percentage detection). Set to `FALSE` to disable for the whole table; use `lt_format()` on specific columns to disable selectively.

Value

A table object that can be piped into `lt_*()` functions.

Examples

```
lt(head(mtcars[, 1:4]))
```

lt_align *Set Column Alignment*

Description

Override the auto-detected alignment for specific columns. By default, numeric columns are right-aligned and character columns are left-aligned.

Usage

```
lt_align(x, columns, align = c("left", "center", "right"))
```

Arguments

x	An lt() object.
columns	Character vector of column names (or a one-sided formula).
align	One of "left", "center", or "right".

Value

x with the alignment recorded.

---

lt_css *Attach Custom CSS*

Description

Add user-supplied stylesheets that render after the built-in CSS, so rules can override the defaults.

Usage

```
lt_css(x, ...)
```

Arguments

x	An lt() object.
...	One or more character scalars: URLs (containing :// or starting with //) or paths to local .css files. A bare filename (no directory component) that does not exist in the working directory is resolved against the stylesheets shipped with lt, so e.g. lt_css(x, "lt-gt.css") uses the bundled gt-like theme.

Value

x with the stylesheets recorded.

Examples

```
## Not run:
lt(head(mtcars[, 1:3])) |> lt_css("custom.css")
lt(head(mtcars[, 1:3])) |> lt_css("https://example.com/theme.css")
lt(head(mtcars[, 1:3])) |> lt_css("lt-gt.css")  # bundled theme

## End(Not run)
```

lt_footnote *Add a Footnote*

Description

Attaches a footnote `text` to a table region. Footnotes are numbered automatically in the order they are added (de-duplicated by text).

Usage

```
lt_footnote(x, text, where, columns = NULL, rows = NULL, match = NULL)
```

Arguments

`x`	An `lt()` object.
`text`	Footnote text.
`where`	One of `'title'`, `'subtitle'`, `'column'`, `'spanner'`, `'group'`, or `'body'`.
`columns`	Character vector of column names or a one-sided formula (for `'column'` or `'body'`). For `'group'` with `match = "starts_with"`, a single prefix string.
`rows`	Integer vector of 1-based row indices (for `'body'`; `NULL` means all rows).
`match`	For `where = "group"`: one of `"exact"` (default), `"starts_with"`, or `"all"`.

Value

`x` with the footnote recorded.

lt_format *Format Numeric Columns*

Description

Control the number of decimal places and thousands separator for numeric columns. Columns passed to this function are excluded from automatic formatting (see the `auto_fmt` argument of `lt()`). To disable auto-format for a column without otherwise changing its display, call `lt_format(x, ~col)` with no other arguments.

Usage

```
lt_format(x, columns, decimals = NULL, big_mark = NULL, percent = NULL)
```

Arguments

x	An lt() object.
columns	Character or integer vector of columns (or a one-sided formula).
decimals	Number of decimal places (default NULL means no change).
big_mark	Thousands separator (e.g., ","). NULL or "" means none.
percent	If TRUE, multiply values by 100 and append "%". If "%", only append "%" without multiplying (for values already in percent scale).

Value

x with the formatting recorded.

---

lt_group *Define Row Groups*

Description

Partition rows into labeled groups. Pass column names to group by those columns' values (the columns are removed from the body and rendered as rowspan cells on the left). Use sep = TRUE to render groups as full-width separator rows instead of rowspan.

Usage

```
lt_group(x, ..., sep = "auto")
```

Arguments

x	An lt() object.
...	A column name or formula (e.g., ~col or ~col1 + col2) to group by column values, or named arguments of the form "Label" = rows (integer vector of 1-based row indices) for manual groups. Unnamed character strings reorder previously defined groups.
sep	If TRUE, render groups as full-width separator rows instead of the default rowspan style. Only supports a single grouping column. The default 'auto' uses separator rows when there is a single grouping column with any value longer than 20 characters.

Value

x with the row groups recorded.

Examples

```
# Group by a column (rowspan, default)
d = data.frame(arm = c("Placebo", "Placebo", "Treatment", "Treatment"),
               stat = c("n", "Mean", "n", "Mean"), value = c(30, 4.2, 31, 6.8))
lt(d) |> lt_group(~ arm)

# Separator-row style
lt(d) |> lt_group(~ arm, sep = TRUE)

# Manual groups (always separator rows)
lt(head(mtcars[, 1:4])) |>
  lt_group("First three" = 1:3, "Last three" = 4:6)
```

---

lt_header *Add a Title and Subtitle*

Description

Add a Title and Subtitle

Usage

```
lt_header(x, title = NULL, subtitle = NULL)
```

Arguments

x	An `lt()` object.
title	A character scalar.
subtitle	A character scalar.

Value

x with the header recorded.

---

lt_indent *Indent Stub Rows*

Description

Add hierarchical indentation to row labels (stub cells). Requires that the table has a stub column (see `lt_stub()`).

Usage

```
lt_indent(x, rows, level = 1)
```

Arguments

x	An lt() object.
rows	Integer vector of 1-based row indices to indent.
level	Indent level (default 1). Each level adds one unit of left padding.

Value

x with the indentation recorded.

---

lt_label *Rename Column Labels*

Description

Override column headers without modifying the underlying data frame.

Usage

```
lt_label(x, ...)
```

Arguments

x	An lt() object.
...	Named arguments of the form col_name = "Display Label".

Value

x with the column label overrides recorded.

---

lt_merge *Merge Columns*

Description

Combine values from multiple columns into a single display column using a pattern. Source columns (all except the first) are hidden by default.

Usage

```
lt_merge(x, columns, pattern = NULL, hide = TRUE)
```

Arguments

x	An lt() object.
columns	Character vector of column names (or a one-sided formula). The first column is the target (receives merged content); the rest are sources.
pattern	A glue-style template using \{1\}, \{2\}, etc. to refer to columns by position. Wrap sections in << and >> for conditional NA handling: "\{1\}<< (\{2\})>>" drops the wrapped portion when any referenced value is missing/empty. If NULL, columns are concatenated separated by spaces.
hide	If TRUE (default), source columns (all but the first) are automatically hidden.

Value

x with the merge recorded.

---

lt_move *Move Columns*

Description

Rearrange column display order without modifying the data frame.

Usage

```
lt_move(x, columns, after = NULL)
```

Arguments

x	An lt() object.
columns	Character vector of column names (or a one-sided formula).
after	Column name after which to place the moved columns. Use NULL to move to the start.

Value

x with the column move recorded.

lt_note *Add a Note*

Description

Notes are rendered in the table footer below numbered footnotes.

Usage

```
lt_note(x, text)
```

Arguments

x	An lt() object.
text	Note text.

Value

x with the note recorded.

lt_output *Shiny Output for lt*

Description

Pair with render_lt() to render an lt() table in a Shiny app. The UI side is a placeholder; an output binding swaps in the rendered <table> whenever the server side re-evaluates the spec. No renderUI() involved — Shiny treats the table like any other custom output.

Usage

```
lt_output(outputId, ...)
```

Arguments

outputId	Output variable name to read the table from.
...	Reserved for future use.

Value

A Shiny UI element.

`lt_spanner` *Add a Column Spanner*

Description

A spanner is a label rendered above a contiguous group of column headers.

Usage

```
lt_spanner(x, label, columns, sep = "[._]")
```

Arguments

`x`	An `lt()` object.
`label`	A character scalar — the spanner text. Alternatively, a two-sided formula `Label ~ col1 + col2` providing both the label (LHS) and columns (RHS). When missing, spanners are inferred from column names.
`columns`	Column names (character or formula). When missing, inferred from column names.
`sep`	Separator pattern for auto-inference (default `"[._]"`).

Details

When called with no `label` or `columns`, infers spanners from column names by splitting on the first `.` or `_` separator. Contiguous columns sharing a prefix are grouped under that prefix, and column labels are shortened to the suffix.

Value

`x` with the spanner recorded.

Note

The columns must be contiguous in the body of the table.

Examples

```
# Explicit spanner
lt(head(iris)) |> lt_spanner("Sepal", c("Sepal.Length", "Sepal.Width"))

# Auto-infer from column names
lt(head(iris)) |> lt_spanner()
```

lt_stub *Designate a Stub Column*

Description

Mark a column as the row-label stub. Its values become left-aligned row headers and the column is removed from the table body. When row groups exist and no stub is declared, the first visible column is automatically promoted; use this function to override that default.

Usage

```
lt_stub(x, column, label = NULL)
```

Arguments

x	An `lt()` object.
column	A column name (character scalar or one-sided formula).
label	Optional header label for the stub column.

Value

x with the stub column recorded.

Examples

```
d = data.frame(endpoint = c("OS", "PFS"), result = c("0.72", "0.58"))
lt(d) |> lt_stub(~ endpoint, label = "Endpoint")
```

lt_style *Style Cells*

Description

Apply CSS styling to specific cells. Target cells by column, row, or both.

Usage

```
lt_style(
  x,
  columns = NULL,
  rows = NULL,
  bold = NULL,
  italic = NULL,
  color = NULL,
  bg = NULL,
  ...
)
```

Arguments

x	An lt() object.
columns	Character vector of column names, a one-sided formula, or NULL for all.
rows	Integer vector of 1-based row indices (or NULL for all).
bold	Logical: apply bold weight?
italic	Logical: apply italic style?
color	Text color (any CSS color value, e.g., "red", "#06c").
bg	Background color.
...	Additional CSS properties as named arguments. Names can be camelCase (e.g., borderLeft) or quoted dash-case (e.g., `border-left`). Values are CSS strings.

Value

x with the style recorded.

Examples

```
lt(head(mtcars[, 1:3])) |>
  lt_style("mpg", rows = 1L, bold = TRUE, borderBottom = "2px solid red")
```

---

lt_sub *Substitute Cell Values*

Description

Replace NA, zero, or small values with display text.

Usage

```
lt_sub(
  x,
  columns = NULL,
  missing = NULL,
  zero = NULL,
  small = NULL,
  small_text = NULL
)
```

Arguments

x	An lt() object.
columns	Character vector of column names, a one-sided formula, or NULL for all.
missing	Replacement for NA cells (e.g., "–"). NULL to leave NAs as empty strings (the default rendering).
zero	Replacement for zero values (e.g., "–").
small	Threshold: values whose absolute value is below this are replaced by small_text.
small_text	Text shown for values below small (e.g., "<0.1").

Value

x with the substitution recorded.

---

lt_width *Set Column Widths*

Description

Set Column Widths

Usage

```
lt_width(x, ...)
```

Arguments

x	An lt() object.
...	Named arguments of the form col_name = "width". Width can be any CSS value (e.g., "100px", "20%", "8em").

Value

x with the column widths recorded.

---

print.lt_tbl *Print an* lt_tbl *(Opens in the Viewer or Browser)*

Description

Print an lt_tbl (Opens in the Viewer or Browser)

Usage

```
## S3 method for class 'lt_tbl'
print(x, ...)
```

Arguments

x	An lt_tbl object.
...	Passed to format().

Value

x, invisibly.

render_lt *Render an lt Table in Shiny*

Description

Render an lt Table in Shiny

Usage

```
render_lt(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

expr	An expression that returns an lt() object.
env	Environment in which to evaluate expr.
quoted	Whether expr is already quoted.

Value

A render function for use with lt_output().

Index