

Package ‘typstable’

February 16, 2026

Title Better 'Typst' Tables

Version 0.1.0

URL <https://freiererson.github.io/typstable/>,
<https://github.com/freiererson/typstable>

BugReports <https://github.com/freiererson/typstable/issues>

Description Create 'Typst' table markup from data frames. Features a pipe-friendly interface for column, row, and cell styling with support for grouped headers, grouped rows, and data-driven formatting.

Depends R (>= 4.1.0)

Encoding UTF-8

RoxygenNote 7.3.3

Imports rlang (>= 1.0.0), tidyselect

Suggests testthat (>= 3.0.0), knitr, rmarkdown, tibble, dplyr

VignetteBuilder knitr

Config/testthat/edition 3

License GPL-3

Date 2026-01-26

NeedsCompilation no

Author Filip Reiererson [aut, cre] (ORCID:
<<https://orcid.org/0009-0003-2393-5810>>)

Maintainer Filip Reiererson <filip.reiererson@gmail.com>

Repository CRAN

Date/Publication 2026-02-16 17:30:02 UTC

Contents

print.typst_table	2
tt	2
tt_cell	4

tt_column	5
tt_header_above	7
tt_hline	9
tt_pack_rows	10
tt_render	11
tt_row	11
tt_save	13
tt_style	14
tt_typst_available	15
tt_vline	16
tt_widths	17
Index	18

print.typst_table	<i>Print a typst_table object</i>
-------------------	-----------------------------------

Description

Prints the Typst markup for a table to the console.

Usage

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

Arguments

- x A typst_table object.
- ... Additional arguments (ignored).

Value

Invisibly returns the Typst code as a character string.

tt	<i>Create a Typst table</i>
----	-----------------------------

Description

Creates a typst_table object from a dataframe or tibble that can be rendered to Typst markup. Use pipe-friendly styling functions like tt_style(), tt_column(), tt_row(), and tt_cell() to customize the table appearance.

Usage

```
tt(
  data,
  cols = tidyselect::everything(),
  col_names = NULL,
  col_widths = "auto",
  align = NULL,
  preamble = NULL,
  escape = TRUE,
  rownames = TRUE
)
```

Arguments

<code>data</code>	A data.frame or tibble to convert to a table.
<code>cols</code>	<tidy-select> Columns to include in the displayed table. Hidden columns remain available for data-driven formatting. Defaults to all columns.
<code>col_names</code>	Optional character vector of display names for columns. Must match the number of selected columns. If NULL, uses column names from data.
<code>col_widths</code>	Column width specification: "auto" (default) creates equal-width columns that fill the container. Use <code>tt_widths()</code> for custom proportions.
<code>align</code>	Column alignment: single value applied to all columns, or vector of alignments. Valid values: "left"/"l", "center"/"c", "right"/"r".
<code>preamble</code>	Optional character string of raw Typst code to insert before the table. Useful for <code>#set</code> rules, <code>#let</code> bindings, or other Typst directives that should apply to the table (e.g., <code>'#set text(font: "Arial")'</code>).
<code>escape</code>	Logical. If TRUE (default), escapes Typst special characters.
<code>rownames</code>	Logical. TRUE (default) includes row names as the first column with an empty header, FALSE excludes them.

Value

A `typst_table` object that can be further styled and rendered.

Examples

```
# Basic table (includes row names by default)
tt(mtcars[1:5, 1:3])

# Select specific columns (excludes row names)
tt(mtcars, cols = c(mpg, cyl, hp), rownames = FALSE)

# Custom column names
tt(mtcars[1:5, 1:3], col_names = c("Miles/Gallon", "Cylinders", "Horsepower"), rownames = FALSE)

# Right-align numeric columns
tt(mtcars[1:5, 1:3], align = "right")
```

tt_cell

*Style specific cells***Description**

Applies formatting to individual cells. Allows for precise control over cell appearance including spanning multiple rows or columns.

Usage

```
tt_cell(
  table,
  row,
  column,
  bold = NULL,
  italic = NULL,
  color = NULL,
  fill = NULL,
  align = NULL,
  font_size = NULL,
  rotate = NULL,
  inset = NULL,
  stroke = NULL,
  colspan = 1,
  rowspan = 1,
  content = NULL
)
```

Arguments

table	A typst_table object.
row	Row number (1-indexed for data rows, 0 for header). Use negative indices to target tt_header_above() rows: -1 is the innermost header_above (closest to the main header), -2 is the next row up, etc. The column is normalized to the start of the header group it falls within.
column	Column specification: integer index or column name.
bold	Logical. Make text bold.
italic	Logical. Make text italic.
color	Text color.
fill	Fill color.
align	Cell alignment.
font_size	Font size.
rotate	Rotation angle (e.g., "90deg", 90, "1.5rad").
inset	Cell padding (e.g., "10pt", "5pt 8pt").

stroke	Stroke (border) specification for the cell. Can be TRUE for default 1pt black, a color, a Typst stroke spec like "2pt + blue", or a Typst dictionary like "(bottom: 1pt)".
colspan	Number of columns to span (default 1).
rowspan	Number of rows to span (default 1).
content	Optional content to replace cell value.

Value

The modified typst_table object.

Examples

```
# Highlight a specific cell
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_cell(1, 1, fill = "yellow", bold = TRUE)

# Cell spanning multiple columns
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_cell(1, 1, colspan = 2, content = "Combined")
```

tt_column

Style specific columns

Description

Applies formatting to one or more columns. Supports both static values and data-driven formatting where styling parameters reference other columns.

Usage

```
tt_column(
  table,
  column,
  width = NULL,
  align = NULL,
  bold = NULL,
  italic = NULL,
  color = NULL,
  fill = NULL,
  border_left = NULL,
  border_right = NULL,
  font_size = NULL,
  rotate = NULL,
  inset = NULL,
  stroke = NULL,
  .missing = c("warn", "ignore", "error")
)
```

Arguments

table	A typst_table object.
column	<tidy-select> Column(s) to style.
width	Column width (e.g., "100pt", "2fr"). Overrides width set in tt().
align	Column alignment: "left", "center", "right".
bold	Logical, column name, or pattern string. If logical, applies to all rows. If column name (unquoted), uses that column's values. If pattern containing {col} (e.g., "bold_{col}"), expands to column name per styled column.
italic	Logical, column name, or pattern string. Same behavior as bold.
color	Text color. Can be a color value, column name, or pattern string.
fill	Fill color. Can be a color value, column name, or pattern string.
border_left	Left border specification (e.g., TRUE, "1pt + gray").
border_right	Right border specification.
font_size	Font size (e.g., "10pt", "0.9em"). Can be a column name or pattern string.
rotate	Rotation angle (e.g., "90deg", 90, "1.5rad"). Can be a column name or pattern string.
inset	Cell padding (e.g., "10pt", "5pt 8pt"). Can be a column name or pattern string.
stroke	Stroke (border) specification for the cell(s). Can be TRUE for default 1pt black, a color, a Typst stroke spec like "2pt + blue", or a Typst dictionary like "(bottom: 1pt)". Can also be a column name or pattern string.
.missing	How to handle missing pattern columns: "warn" (default) emits a warning, "ignore" silently skips, "error" stops execution.

Details**Data-driven formatting:**

Style parameters can reference columns in the original data for per-row values:

```
data |>
  mutate(bg_color = ifelse(value > 100, "red", "white")) |>
  tt(cols = c(name, value)) |>
  tt_column(value, fill = bg_color)
```

The bg_color column is hidden from display but used for styling.

Pattern-based styling:

When styling multiple columns with a naming convention, use {col} patterns to automatically expand column references:

Instead of repetitive calls:

```
data |>
  tt(cols = c(mpg, qsec)) |>
  tt_column(everything(), color = "color_{col}", fill = "bg_{col}")
```

For column mpg, this looks for color_mpg and bg_mpg in the data.

Value

The modified typst_table object.

Examples

```
# Right-align and bold a numeric column
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_column(mpg, align = "right", bold = TRUE)

# Style multiple columns at once
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_column(c(mpg, disp), align = "right", color = "blue")

# Pattern-based styling (column-specific style columns)
df <- data.frame(
  a = 1:3, b = 4:6,
  color_a = c("red", "green", "blue"),
  color_b = c("black", "gray", "white")
)
tt(df, cols = c(a, b), rownames = FALSE) |>
  tt_column(c(a, b), color = "color_{col}")
```

tt_header_above

Add grouped header row above column names

Description

Adds a spanning header row above the existing column names to group related columns together.

Usage

```
tt_header_above(
  table,
  header,
  bold = TRUE,
  align = "center",
  color = NULL,
  fill = NULL,
  italic = NULL,
  font_size = NULL,
  rotate = NULL,
  inset = NULL,
  stroke = NULL,
  line = TRUE,
  gap = "10pt"
)
```

Arguments

<code>table</code>	A <code>typst_table</code> object.
<code>header</code>	Named vector specifying header groups. Names are the group labels, values are the number of columns each group spans. Use empty string "" for columns without a group header.
<code>bold</code>	Logical. Make header text bold (default TRUE).
<code>align</code>	Header alignment (default "center").
<code>color</code>	Text color.
<code>fill</code>	Fill color.
<code>italic</code>	Logical. Make header text italic.
<code>font_size</code>	Font size.
<code>rotate</code>	Rotation angle (e.g., "90deg", 90, "1.5rad").
<code>inset</code>	Cell padding (e.g., "10pt", "5pt 8pt").
<code>stroke</code>	Stroke (border) specification for the header cell(s). Can be TRUE for default 1pt black, a color, a Typst stroke spec like "2pt + blue", or a Typst dictionary like "(bottom: 1pt)".
<code>line</code>	Logical. Add horizontal line below the header (default TRUE).
<code>gap</code>	Width of visual gap between header groups (e.g., "10pt", "0.5em"). When specified, empty columns are inserted between groups. Default "10pt". Use NULL to disable gaps.

Value

The modified `typst_table` object.

Examples

```
# Group columns under headers
tt(mtcars[1:5, 1:6], rownames = FALSE) |>
  tt_header_above(c("Performance" = 3, "Design" = 3))

# Leave some columns ungrouped
tt(mtcars[1:5, 1:6], rownames = FALSE) |>
  tt_header_above(c(" " = 1, "Group A" = 2, "Group B" = 3))

# Disable gaps between groups
tt(mtcars[1:5, 1:6], rownames = FALSE) |>
  tt_header_above(c(" " = 1, "Group A" = 2, "Group B" = 3), gap = NULL)
```

tt_hline	<i>Add horizontal line to table</i>
----------	-------------------------------------

Description

Adds a horizontal line at a specific position in the table.

Usage

```
tt_hline(table, y, start = NULL, end = NULL, stroke = TRUE)
```

Arguments

table	A <code>typst_table</code> object.
y	Position of the line (0 = top of table, 1 = below header, 2 = below first data row, etc.).
start	Starting column for partial line (0-indexed, inclusive).
end	Ending column for partial line (0-indexed, inclusive).
stroke	Line style: TRUE for default, color name, or full stroke spec like "2pt + red".

Value

The modified `typst_table` object.

Examples

```
# Add line below header
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_hline(1)

# Add colored line
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_hline(1, stroke = "blue")

# Partial line spanning some columns
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_hline(1, start = 0, end = 2)
```

tt_pack_rows	<i>Group rows under a label</i>
--------------	---------------------------------

Description

Inserts a group label row and optionally indents the grouped rows. Useful for creating visual sections in the table.

Usage

```
tt_pack_rows(
  table,
  group_label = NULL,
  start_row = NULL,
  end_row = NULL,
  index = NULL,
  indent = TRUE,
  bold_label = TRUE
)
```

Arguments

table	A <code>typst_table</code> object.
group_label	The label text for the group (used with <code>start_row/end_row</code>).
start_row	First row number in the group (1-indexed).
end_row	Last row number in the group.
index	A named numeric vector for multiple groups (alternative style). Names are group labels, values are row counts. Example: <code>c("Group A" = 3, "Group B" = 5)</code> creates two groups where "Group A" covers rows 1-3 and "Group B" covers rows 4-8.
indent	Logical. Indent rows in the group (default TRUE).
bold_label	Logical. Make the group label bold (default TRUE).

Value

The modified `typst_table` object.

Examples

```
# Group rows with a label
tt(mtcars[1:10, 1:3], rownames = FALSE) |>
  tt_pack_rows("4 Cylinders", 1, 5) |>
  tt_pack_rows("6 Cylinders", 6, 10)

# Using index parameter (alternative style)
tt(mtcars[1:10, 1:3], rownames = FALSE) |>
```

```
tt_pack_rows(index = c("4 Cylinders" = 5, "6 Cylinders" = 5))
```

tt_render	<i>Render a typst_table to Typst markup</i>
-----------	---

Description

Generates Typst code from a `typst_table` object. This is called automatically by `print()` and `knit_print()`, but can also be called directly.

Usage

```
tt_render(table)
```

Arguments

`table` A `typst_table` object.

Value

A character string containing Typst table markup.

Examples

```
code <- tt(mtcars[1:3, 1:3], rownames = FALSE) |> tt_render()
cat(code)
```

tt_row	<i>Style specific rows</i>
--------	----------------------------

Description

Applies formatting to one or more rows. Use `row = 0` to style the header row.

Usage

```
tt_row(
  table,
  row,
  bold = NULL,
  italic = NULL,
  color = NULL,
  fill = NULL,
  align = NULL,
```

```

    font_size = NULL,
    rotate = NULL,
    inset = NULL,
    stroke = NULL,
    hline_above = NULL,
    hline_below = NULL
  )

```

Arguments

<code>table</code>	A <code>typst_table</code> object.
<code>row</code>	Integer vector of row numbers to style. Use 0 for the header row, 1 to n for data rows. Use negative indices to target <code>tt_header_above()</code> rows: -1 is the innermost header_above (closest to the main header), -2 is the next row up, etc.
<code>bold</code>	Logical. Make text bold.
<code>italic</code>	Logical. Make text italic.
<code>color</code>	Text color.
<code>fill</code>	Fill color.
<code>align</code>	Row alignment override.
<code>font_size</code>	Font size.
<code>rotate</code>	Rotation angle (e.g., "90deg", 90, "1.5rad").
<code>inset</code>	Cell padding (e.g., "10pt", "5pt 8pt").
<code>stroke</code>	Stroke (border) specification for the cell(s). Can be TRUE for default 1pt black, a color, a Typst stroke spec like "2pt + blue", or a Typst dictionary like "(bottom: 1pt)".
<code>hline_above</code>	Add horizontal line above the row. Can be TRUE for default line or a stroke specification.
<code>hline_below</code>	Add horizontal line below the row.

Value

The modified `typst_table` object.

Examples

```

# Style header row
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_row(0, bold = TRUE, fill = "gray")

# Highlight specific rows
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_row(c(1, 3, 5), fill = "#ffffcc")

# Add horizontal lines
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_row(3, hline_above = TRUE, hline_below = TRUE)

```

tt_save	<i>Save a typst_table to SVG, PNG, PDF, or Typst source</i>
---------	---

Description

Exports a `typst_table` object to various output formats. For SVG, PNG, and PDF output, the Typst CLI must be installed and available on the system PATH.

Usage

```
tt_save(
  table,
  file,
  width = "auto",
  height = "auto",
  margin = "0.5em",
  ppi = 144,
  typst_path = NULL,
  overwrite = TRUE
)
```

Arguments

<code>table</code>	A <code>typst_table</code> object created with <code>tt()</code> .
<code>file</code>	Output file path. The format is determined by the file extension: <code>.svg</code> , <code>.png</code> , <code>.pdf</code> , or <code>.typ</code> (Typst source).
<code>width</code>	Page width. Use <code>"auto"</code> (default) for automatic sizing based on table content, or specify a Typst length like <code>"6in"</code> , <code>"15cm"</code> , <code>"400pt"</code> .
<code>height</code>	Page height. Use <code>"auto"</code> (default) for automatic sizing, or specify a Typst length.
<code>margin</code>	Page margin around the table. Default is <code>"0.5em"</code> . Use <code>"0pt"</code> for no margin.
<code>ppi</code>	Pixels per inch for PNG output. Default is 144. Higher values produce larger, higher-resolution images.
<code>typst_path</code>	Path to the Typst executable. If <code>NULL</code> (default), searches for <code>typst</code> on the system PATH.
<code>overwrite</code>	Logical. If <code>TRUE</code> (default), overwrites existing files. If <code>FALSE</code> , throws an error when the output file already exists.

Details

For `.typ` output, only the Typst source code is written and the Typst CLI is not required.

For `.svg`, `.png`, and `.pdf` output, the function: 1

1. Generates a complete Typst document with appropriate page settings
2. Writes it to a temporary `.typ` file

- 3. Calls `typst compile` to render the output
- 4. Cleans up the temporary file

The width and height parameters control the page size. Using "auto" for both (the default for both (the default) creates a page that fits the table content exactly.

Value

Invisibly returns the output file path.

Examples

```
# Save Typst source (no CLI required)
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_save(file.path(tempdir(), "table.typ"))

if (tt_typst_available()) {
  # Save as SVG
  tt(mtcars[1:5, 1:3], rownames = FALSE) |>
    tt_save(file.path(tempdir(), "table.svg"))

  # Save as PNG with higher resolution
  tt(mtcars[1:5, 1:3], rownames = FALSE) |>
    tt_save(file.path(tempdir(), "table.png"), ppi = 300)

  # Save as PDF with specific page size
  tt(mtcars[1:5, 1:3], rownames = FALSE) |>
    tt_style(stroke = TRUE) |>
    tt_save(file.path(tempdir(), "table.pdf"), width = "6in", height = "4in")
}
```

tt_style	<i>Style overall table appearance</i>
----------	---------------------------------------

Description

Sets global styling properties for the entire table including stroke (borders), fill, striped rows, and spacing.

Usage

```
tt_style(
  table,
  stroke = NULL,
  fill = NULL,
  striped = NULL,
  inset = NULL,
  row_gutter = NULL,
```

```

    column_gutter = NULL,
    position = NULL,
    full_width = FALSE
  )

```

Arguments

table	A typst_table object.
stroke	Stroke (border) specification: TRUE for default 1pt black borders, a color name/hex for 1pt borders in that color, or a Typst stroke specification like "2pt + blue". When stroke is set, gap columns and separator lines from tt_header_above() are automatically suppressed.
fill	Fill color for the entire table.
striped	Logical. If TRUE, alternates row background colors for readability.
inset	Cell padding. Can be a single value (e.g., "5pt") or named vector for different padding on each side.
row_gutter	Vertical spacing between rows.
column_gutter	Horizontal spacing between columns.
position	Table position on page: "auto", "left", "center", "right".
full_width	Logical. If TRUE, table spans full page width.

Value

The modified typst_table object.

Examples

```

# Add borders and striped rows
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_style(stroke = TRUE, striped = TRUE)

# Custom border color and padding
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_style(stroke = "gray", inset = "8pt")

```

tt_typst_available	<i>Check if Typst CLI is available</i>
--------------------	--

Description

Tests whether the Typst command-line interface is installed and accessible.

Usage

```
tt_typst_available(typst_path = NULL)
```

Arguments

typst_path Optional path to the Typst executable. If NULL, searches for typst on the system PATH.

Value

TRUE if Typst is available, FALSE otherwise.

Examples

```
# Check if Typst is installed
if (tt_typst_available()) {
  message("Typst is available")
} else {
  message("Typst not found - install from https://typst.app/")
}
```

tt_vline	<i>Add vertical line to table</i>
----------	-----------------------------------

Description

Adds a vertical line at a specific position in the table.

Usage

```
tt_vline(table, x, start = NULL, end = NULL, stroke = TRUE)
```

Arguments

table A typst_table object.

x Position of the line (0 = before first column, 1 = after first column, etc.).

start Starting row for partial line (0-indexed, inclusive, 0 = header).

end Ending row for partial line (0-indexed, inclusive).

stroke Line style: TRUE for default, color name, or full stroke spec.

Value

The modified typst_table object.

Examples

```
# Add line after first column
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_vline(1)

# Add partial vertical line (data rows only)
tt(mtcars[1:5, 1:3], rownames = FALSE) |>
  tt_vline(1, start = 1)
```

tt_widths	<i>Set proportional column widths</i>
-----------	---------------------------------------

Description

Sets column widths as proportions that fill the page or container width. Widths are converted to Typst fr (fractional) units.

Usage

```
tt_widths(table, ...)
```

Arguments

table	A typst_table object.
...	Width values as numbers. Can be unnamed (applied in order) or named by column. Values represent relative proportions.

Details

Widths are relative proportions, not absolute values. For example, `tt_widths(tbl, 1, 2, 1)` creates columns at 25%, 50%, 25% of the container width.

Value

The modified typst_table object.

Examples

```
# Equal widths
tt(mtcars[1:5, 1:3], rownames = FALSE) |> tt_widths(1, 1, 1)

# Proportional widths (25%, 50%, 25%)
tt(mtcars[1:5, 1:3], rownames = FALSE) |> tt_widths(1, 2, 1)

# Named columns
tt(mtcars[1:5, 1:3], rownames = FALSE) |> tt_widths(mpg = 1, cyl = 2, disp = 1)
```

Index

`print.typst_table`, [2](#)

`tt`, [2](#)

`tt_cell`, [4](#)

`tt_column`, [5](#)

`tt_header_above`, [7](#)

`tt_hline`, [9](#)

`tt_pack_rows`, [10](#)

`tt_render`, [11](#)

`tt_row`, [11](#)

`tt_save`, [13](#)

`tt_style`, [14](#)

`tt_typst_available`, [15](#)

`tt_vline`, [16](#)

`tt_widths`, [17](#)