

Package: tabtibble (via r-universe)

October 7, 2026

Title Simplify Reporting Many Tables

Version 0.0.2.9001

Description Simplify reporting many tables by creating tibbles of tables. With 'tabtibble', a tibble of tables is created with captions and automatic printing using `knit_print()`.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports knitr, vctrs

Suggests dplyr, flextable, glue, gt, pdftools, quarto, rmarkdown, spelling, table1, testthat (>= 3.0.0), tibble, tidyr, tinytable

Config/testthat/edition 3

URL <https://github.com/humanpred/tabtibble>,
<https://humanpred.github.io/tabtibble/>

BugReports <https://github.com/humanpred/tabtibble/issues>

VignetteBuilder knitr

Language en-US

Config/roxygen2/version 8.1.0

Repository <https://humanpred.r-universe.dev>

Date/Publication 2026-10-07 02:24:09 UTC

RemoteUrl <https://github.com/humanpred/tabtibble>

RemoteRef HEAD

RemoteSha 87cf4a4f54d64c7aac7d58591c571f2283b656d4

Contents

<code>knit_print.tab_list</code>	2
<code>knit_print.tab_tibble</code>	3
<code>new_tab_tibble</code>	4
<code>print_tabtibble</code>	5

knit_print.tab_list	<i>Print a tab_list</i>
---------------------	-------------------------

Description

Print a tab_list

Usage

```
## S3 method for class 'tab_list'
knit_print(
  x,
  ...,
  caption,
  label = NULL,
  topic_cols = NULL,
  print_fun = NULL,
  tab_prefix = NULL,
  tab_suffix = "\n\n"
)
```

Arguments

x	The tab_list object to print
...	passed to print_fun
caption	The caption for each table as a character vector
label	Character vector of Quarto labels (without the tbl- prefix), one per table, matching x in length. NULL (the default) derives a label from each caption via derive_label().
topic_cols	A list the same length as x, each element a character vector of column names to group into repeating topic headers (used only by the "typst" backend; see render_backend_table()); NULL (the default) is treated as an empty vector for every table. Before a table using the "typst" backend, a Typst rule allowing its figure to break across pages is emitted ahead of the table's own crossref div – see emit_typst_breakable_figure_rule().
print_fun	Override the default printing using print_tabtibble. If provided it is a function taking arguments of x (one data.frame to print), caption (the caption for that data.frame), and
tab_prefix, tab_suffix	Any text to add before/after each figure (NULL to omit)

Details

Individual tables are printed with the `print_tabtibble()` S3 generic function. Under Quarto (see `detect_render_mode()`), each table is wrapped in a fenced Div with id `#tbl-<label>` (Quarto's syntax for a crossreferenceable table from computational output), so `@tbl-<label>` resolves and, when the report target is Typst, the table appears in a Typst `#outline(target: figure.where(kind: "quarto-float-tbl"` (the list of tables; that outline call belongs in the report template, not here). The caption is emitted once, as the Div's trailing paragraph, which Quarto promotes to the figure's caption; it is Typst/Pandoc-escaped (see `escape_typst()`) so caption text containing markup-significant characters renders literally.

Under plain R Markdown, that Div is skipped: rendered by plain Pandoc (no Quarto filter to interpret it), it would show up as a literal, uncaptioned wrapper rather than a table figure. Instead, each table carries an ordinary caption the output target understands – see `render_backend_table()` – so a report that never used Quarto keeps working exactly as it did before crossref support was added.

By default (the `tabtibble.knitr.auto.asis` option, `TRUE`), the rendered markdown is captured and returned as `knitr::asis_output()`, the same mechanism pander uses for its `knitr.auto.asis` option: knitr inserts an `asis`-classed return value into the document directly, whatever the chunk's own results setting, so a chunk printing a `tab_tibble/tab_list` needs no `results='asis'` chunk option. Set `options(tabtibble.knitr.auto.asis = FALSE)` to opt out and go back to writing directly with `cat()` (requiring `results='asis'`, with a warning when it is missing) – useful if something downstream needs to see the literal `cat()` side effect rather than a returned value.

Value

With `tabtibble.knitr.auto.asis TRUE` (the default), a `knitr::asis_output()` character value holding the rendered markdown. With it `FALSE`, `x` invisibly (the markdown is written with `cat()` as a side effect instead).

See Also

Other knitters: [knit_print.tab_tibble\(\)](#)

`knit_print.tab_tibble` *Print a tab_tibble*

Description

Print a `tab_tibble`

Usage

```
## S3 method for class 'tab_tibble'
knit_print(x, ...)
```

Arguments

<code>x</code>	The <code>tab_tibble</code> object to print
<code>...</code>	Passed to subsequent methods

Value

The result of `knit_print(x$table, ...)`: by default (see `knit_print.tab_list()`) a `knitr::asis_output()` value, so a chunk printing a `tab_tibble` needs no `results='asis'`.

See Also

Other knitters: [knit_print.tab_list\(\)](#)

new_tab_tibble	Create a new tab_tibble object
----------------	--------------------------------

Description

Create a new `tab_tibble` object

Usage

```
new_tab_tibble(x)
```

```
new_tab_list(x)
```

Arguments

x The object to convert. A `data.frame` (or `tibble`) with columns `table` (list of tables; see `new_tab_list()`), `caption` (character), and optionally `label` (character, unique, a valid Quarto label without the `tbl-` prefix – see `derive_label()`) and `topic_cols` (list column, one character vector per table, naming columns – in outer-to-inner order – that the “`typst`” backend groups into repeating topic headers instead of repeating their values on every row; see `render_typst_table()`). When `label` is absent or `NA`/empty for a row, a stable label is derived from that row’s caption, so re-rendering the same caption always yields the same label. Each `topic_cols[[i]]` must name columns present in `table[[i]]`, and `table[[i]]` must already be sorted by them.

Value

An object with the desired class

Functions

- `new_tab_list()`: Create a new `tab_list` object

print_tabtibble	<i>Print a single table from a tablist</i>
-----------------	--

Description

Print a single table from a tablist

Usage

```
print_tabtibble(x, ...)

## Default S3 method:
print_tabtibble(x, caption = NULL, ..., topic_cols = character(0))
```

Arguments

x	A table to print
...	Passed to the backend's table constructor (<code>knitr::kable()</code> / <code>tinytable::tt()</code> / <code>gt::gt()</code> / <code>flextable::flextable()</code>), or to x's own <code>knit_print()</code> method.
caption	The table's caption, or NULL (the default). Under Quarto, ignored – the crossref wrapper in <code>knit_print.tab_list()</code> applies it instead, so it is not duplicated in the rendered Typst figure. Under R Markdown, applied as the table's own caption (see <code>render_backend_table()</code>); <code>knit_print.tab_list()</code> always supplies it.
topic_cols	Character vector of column names to group into repeating topic headers; only used by the "typst" backend under Quarto (see <code>render_backend_table()</code>).

Value

x, invisibly

Methods (by class)

- `print_tabtibble(default)`: Print a single table from a tablist using the backend selected by the `tabtibble.backend` option ("markdown" (default), "tinytable", "gt", "flextable", or "typst"; see `render_backend_table()` when x is a plain `data.frame`. An x that is already a `gt`, `tinytable`, `flextable`, or `table1` object (see `new_tab_list()`) is printed via its own `knitr::knit_print()` method instead, regardless of `tabtibble.backend`, so a table the caller built with a specific package is rendered exactly as they built it; under R Markdown (see `detect_render_mode()`), caption (when supplied) is still shown for it, as a plain bold paragraph before the table, since there is no single API across those packages for adding a caption to an already-built object.

Index

* **knitters**

- knit_print.tab_list, [2](#)
- knit_print.tab_tibble, [3](#)

- knit_print.tab_list, [2](#)
- knit_print.tab_list(), [4](#)
- knit_print.tab_tibble, [3](#)
- knit_print.tab_tibble(), [3](#)

- new_tab_list(new_tab_tibble), [4](#)
- new_tab_tibble, [4](#)

- print_tabtibble, [5](#)