Package 'tablet'

October 7, 2026

Type Package

Title Tabulate Descriptive Statistics in Multiple Formats

Version 0.9.4

Maintainer Tim Bergsma <bergsmat@gmail.com>

BugReports https://github.com/bergsmat/tablet/issues

Description Creates a table of descriptive statistics
for factor and numeric columns in a data frame. Displays
these by groups, if any. Highly customizable, with support
for 'html' and 'pdf' provided by 'kableExtra'. Respects
original column order, column labels, and factor level order.
See ?tablet.data.frame and vignettes.

License GPL-3

Encoding UTF-8

Imports dplyr (>= 1.0.2), yamlet (>= 0.10.21), rlang, tidyr,
kableExtra (>= 0.9.0), spork (>= 0.2.7), magrittr, fs,
reactable

RoxygenNote 7.3.3

VignetteBuilder knitr

Suggests knitr, rmarkdown, boot, testthat, shiny, shinyFiles, haven,
yaml, sortable, latexpdf, tinytex, tools, csv, xtable,
shinyAce, R.utils

NeedsCompilation no

Author Tim Bergsma [aut, cre]

Repository CRAN

Date/Publication 2026-10-07 20:00:02 UTC

Contents

as.data.frame.gridtable 2
as_calibrated.data.frame 3

as_gridtable.character 4
as_gridtable.data.frame 5
as_kable.tablet 5
as_tablet.data.frame 8
header_rows.tablet 8
io_tablet 9
kbl 10
kbl.calibrated 11
kbl.default 11
kbl.gridtable 13
kbl.tablet 14
listado 16
mesa 17
print.gridtable 18
recap.knitr_kable 18
tablet.data.frame 19

Index **22**

`as.data.frame.gridtable` *Coerce a gridtable to a Calibrated Data Frame*

Description

Parse a validated gridtable object directly to a calibrated data frame. Columns have `width` attributes suggested by the grid table spans. Continuation lines within a grid-table cell are collapsed with single spaces. Cell markup is otherwise left untouched for downstream renderers.

Usage

```
## S3 method for class 'gridtable'
as.data.frame(
  x,
  row.names = NULL,
  optional = FALSE,
  ...,
  scale = getOption("tablet.gridtable.scale", 1)
)
```

Arguments

`x`	Character scalar with class `gridtable`.
`row.names`	Optional row names for the returned data frame.
`optional`	Currently unused; accepted for compatibility with `as.data.frame`.
`...`	Currently unused.
`scale`	Numeric scalar multiplier applied to grid-table dash counts when deriving column-level `width` attributes.

Value

A data frame with class `c("calibrated", "data.frame")`. Each column has a `width` attribute derived from the final grid table dash count multiplied by `scale` and encoded as an em width.

See Also

`as_gridtable.character`, `as_calibrated.data.frame`

Other gridtable: `as_calibrated()`, `as_calibrated.data.frame()`, `as_gridtable()`, `as_gridtable.character()`, `as_gridtable.data.frame()`, `kbl()`, `kbl.calibrated()`, `kbl.default()`, `kbl.gridtable()`, `print.gridtable()`

---

as_calibrated.data.frame

Calibrate Data Frame Column Widths

Description

Assign column-level `width` attributes to a data frame and return a calibrated data frame for downstream table rendering. Widths are recycled as necessary; a warning is issued when the number of supplied widths differs from the number of columns.

Usage

```
## S3 method for class 'data.frame'
as_calibrated(x, width, ...)
```

Arguments

`x`	A data frame.
`width`	Character or numeric vector of column widths. Values are recycled as necessary.
`...`	Currently unused.

Value

A data frame with class `calibrated` and column-level `width` attributes.

See Also

`as_calibrated`, `column_spec`

Other gridtable: `as.data.frame.gridtable()`, `as_calibrated()`, `as_gridtable()`, `as_gridtable.character()`, `as_gridtable.data.frame()`, `kbl()`, `kbl.calibrated()`, `kbl.default()`, `kbl.gridtable()`, `print.gridtable()`

`as_gridtable.character` *Validate and Normalize a Simple Pandoc Grid Table*

Description

Validate a character string containing a simple Pandoc grid table and rebuild it by parsing structural '+' and '|' delimiters, discarding the original spacing, and returning an aligned grid table.

Usage

```
## S3 method for class 'character'
as_gridtable(x, ...)
```

Arguments

x	Character scalar containing one simple Pandoc grid table.
...	Currently unused.

Details

The accepted grid-table form is intentionally restricted: every boundary row must contain the same number of '+' delimiters; every text row is split on every '|' and must therefore contain the same number of '|' delimiters; each boundary span must contain at least one '-' or '='; the first boundary row and all body boundary rows must use '-'; the second boundary row must be the only '=' boundary row; and exactly one header text row must appear between the first and second boundary rows; header cells must be non-empty and unique after trimming. Literal '|' characters inside cell text are interpreted as delimiters, not cell content.

Value

Character scalar with class `gridtable` containing a canonical Pandoc grid table.

See Also

Other gridtable: `as.data.frame.gridtable()`, `as_calibrated()`, `as_calibrated.data.frame()`, `as_gridtable()`, `as_gridtable.data.frame()`, `kbl()`, `kbl.calibrated()`, `kbl.default()`, `kbl.gridtable()`, `print.gridtable()`

as_gridtable.data.frame *Convert a Data Frame to gridtable*

Description

Build a simple Pandoc grid table from a data frame. Column names become the grid-table header row, each data-frame row becomes one body row, and column widths are calibrated from the widest formatted value in each column.

Usage

```
## S3 method for class 'data.frame'
as_gridtable(x, ...)
```

Arguments

x	A data frame.
...	Currently unused.

Value

Character scalar with class `gridtable` containing a canonical Pandoc grid table.

See Also

as_gridtable, as_gridtable.character

Other gridtable: as.data.frame.gridtable(), as_calibrated(), as_calibrated.data.frame(), as_gridtable(), as_gridtable.character(), kbl(), kbl.calibrated(), kbl.default(), kbl.gridtable(), print.gridtable()

as_kable.tablet *Coerce Tablet to Kable*

Description

Renders a tablet. Calls kbl and implements special features like grouped columns. See also tablet.data.frame. As of 0.9.0, calling kbl is equivalent to calling as_kable.

Usage

```
## S3 method for class 'tablet'
as_kable(
  x,
  ...,
  booktabs = TRUE,
  escape = FALSE,
  escape_latex = tablet::escape_latex,
  escape_html = function(x, ...) x,
  variable = " ",
  col.names = NA,
  linebreak = TRUE,
  align = "c",
  double_escape = FALSE,
  linebreaker = "\n",
  pack_rows = list(escape = escape),
  secondary = FALSE
)
```

Arguments

x	tablet
...	passed to kbl
booktabs	passed to kbl
escape	passed to kbl; defaults FALSE to allow header linebreaks
escape_latex	a function to pre-process column names and content if 'escape' is FALSE (e.g., manual escaping, latex only); default escape_latex
escape_html	a function to pre-process column names and content if 'escape' is FALSE (e.g., manual escaping, html only)
variable	a column name for the variables
col.names	passed to kbl after any linebreaking
linebreak	whether to invoke linebreak for column names
align	passed to linebreak for column names
double_escape	passed to linebreak for column names
linebreaker	passed to linebreak for column names in latex; for html, linebreaker is replaced with
pack_rows	named list passed to pack_rows for finer control of variable names
secondary	passed to escape_latex

Value

like kbl

Examples

```
library(boot)
library(dplyr)
library(magrittr)
library(haven)
library(yamlet)
library(spork)
melanoma %>%
  select(-time, -year) %>%
  mutate(sex = factor(sex), ulcer = factor(ulcer)) %>%
  group_by(status) %>%
  tablet %>%
  as_kable

x <- system.file(package = 'tablet', 'shiny-examples/mesa/data/adsl.sas7bdat')
x %<>% read_sas %>% data.frame
decorations(x)

# calculate BMI by assuming all males are 1.75 m, all females 1.63 cm
x %<>% mutate(height = ifelse(sex == 'F', 1.63, 1.75))
x %<>% mutate(bmi = signif(digits = 3, weight / (height^2)))
x %<>% filter(saffl == 'Y')
x %<>% select(trt01a, age, sex, weight, bmi)
x %<>% redecorate('
trt01a: [ Treatment, [ Placebo, TRT 10 mg, TRT 20 mg ]]
age:    [ Age, year ]
sex:    [ Sex, [ Female: F, Male: M ]]
weight: [ Body Weight, kg ]
bmi:    [ Index_body mass, kg/m^2 ]
')
x %<>% resolve
x %<>% group_by(trt01a)

x %>% tablet %>% as_kable

# supply default and unit-conditional latex titles
x %<>% modify(title = concatenate(as_latex(as_spork(c(.data$label)))))
x %<>% modify(
age, weight, bmi,
  title = concatenate(
    sep = '',  # default ok in pdf
    as_latex(
      as_spork(
        c(.data$label, ' (', .data$units, ')')
      )
    )
  )
)
x %>% tablet %>% as_kable
```

as_tablet.data.frame *Coerce data.frame to tablet*

Description

Coerces data.frame to tablet. Checks format and assigns the class. See `tablet.data.frame`.

Usage

```
## S3 method for class 'data.frame'
as_tablet(x, ...)
```

Arguments

x	data.frame
...	passed arguments

Value

tablet

See Also

Other tablet: `as_tablet()`, `header_rows()`, `header_rows.tablet()`, `tablet.tablette()`, `tablette.tablet()`

header_rows.tablet *Identify Header Rows for tablet*

Description

Identifies header rows for tablet.

Usage

```
## S3 method for class 'tablet'
header_rows(x, ...)
```

Arguments

x	tablet
...	ignored

Value

integer: indices for those rows representing headers

See Also

Other tablet: as_tablet(), as_tablet.data.frame(), header_rows(), tablet.tablette(), tablette.tablet()

---

io_tablet *Import and Export Tablet*

Description

Imports or exports tablet as comma-separated variable. Generic, with methods that extend io_csv.

Usage

```
io_tablet(x, ...)
```

Arguments

x	object
...	passed arguments

Value

See methods.

See Also

Other io: io_csv.tablet(), io_tablet.character(), io_tablet.data.frame(), io_tablet.tablet()

Examples

```
library(yamlet)
library(boot)
library(dplyr)
library(magrittr)
library(tablet)
library(kableExtra)
melanoma %<>% redecorate('
time: [ Time, day ]
status: [ Patient Status, [ Died^1: 1, Alive^2: 2, Unrelated: 3 ]]
sex: [ Sex^3, [ Male: 1, Female^4: 0 ]]
age: [ Age, year ]
year: [ Year of Operation ]
ulcer: [ Ulceration, [ Present: 1, Absent: 0 ]]
thickness: [ Tumor Thickness, mm ]
')
tbl <- melanoma %>%
  select(-time, -year) %>%
  group_by(status, ulcer) %>%
```

```
  enscript %>%
  tablet(Missing~NULL)
path <- tbl %>% io_tablet(tempfile(fileext = '.csv'))
# alternatively:
path <- tbl %>% io_csv(tempfile(fileext = '.csv'))
tab <- path %>% io_tablet

tbl %>%
kbl(booktabs = TRUE) %>%
kable_styling(latex_options = 'scale_down')

tab %>%
kbl(booktabs = TRUE) %>%
kable_styling(latex_options = 'scale_down')
```

---

kbl *Render with kbl*

Description

Generic wrapper for kbl. The default method is the existing kableExtra::kbl definition.

Usage

```
kbl(x, ...)
```

Arguments

x	object
...	passed arguments

Value

see methods

See Also

kbl

Other gridtable: as.data.frame.gridtable(), as_calibrated(), as_calibrated.data.frame(), as_gridtable(), as_gridtable.character(), as_gridtable.data.frame(), kbl.calibrated(), kbl.default(), kbl.gridtable(), print.gridtable()

kbl.calibrated *Render a Calibrated Data Frame with kbl*

Description

Render a calibrated data frame with `kbl` and optionally apply column-level `width` attributes with `column_spec`.

Usage

```
## S3 method for class 'calibrated'
kbl(x, ..., use_widths = TRUE)
```

Arguments

`x`	A data frame with class `calibrated`.
`...`	Additional arguments passed to `kbl.default`.
`use_widths`	Logical scalar indicating whether column-level width attributes should be applied with `column_spec`. Defaults to TRUE.

Value

A kable object.

See Also

`kbl`, `kbl.default`, `column_spec`

Other gridtable: `as.data.frame.gridtable()`, `as_calibrated()`, `as_calibrated.data.frame()`, `as_gridtable()`, `as_gridtable.character()`, `as_gridtable.data.frame()`, `kbl()`, `kbl.default()`, `kbl.gridtable()`, `print.gridtable()`

kbl.default *Default kbl Method*

Description

Calls the existing `kbl` implementation.

Usage

```
## Default S3 method:
kbl(
  x,
  format,
  digits = getOption("digits"),
  row.names = NA,
  col.names = NA,
  align,
  caption = NULL,
  label = NULL,
  format.args = list(),
  escape = TRUE,
  table.attr = getOption("knitr.table.html.attr", ""),
  booktabs = FALSE,
  longtable = FALSE,
  tabular = if (longtable) "longtable" else "tabular",
 valign = if (tabular %in% c("tabularx", "xltabular")) "{\\linewidth}" else "[t]",
  position = "",
  centering = TRUE,
  vline = getOption("knitr.table.vline", if (booktabs) "" else "|"),
  toprule = getOption("knitr.table.toprule", if (booktabs) "\\toprule" else
    "\\hline"),
 bottomrule = getOption("knitr.table.bottomrule", if (booktabs) "\\bottomrule" else
    "\\hline"),
  midrule = getOption("knitr.table.midrule", if (booktabs) "\\midrule" else
    "\\hline"),
  linesep = if (booktabs) c("", "", "", "", "\\addlinespace") else "\\hline",
  caption.short = "",
  table.envir = if (!is.null(caption)) "table",
  ...
)
```

Arguments

x	Object to render.
format, digits, row.names, col.names, align, caption, label	Arguments passed to kbl.
format.args, escape, table.attr, booktabs, longtable, tabular, valign	Arguments passed to kbl.
position, centering, vline, toprule, bottomrule, midrule, linesep	Arguments passed to kbl.
caption.short, table.envir	Arguments passed to kbl.
...	Additional arguments passed to kbl.

Value

A kable object.

See Also

kbl

Other gridtable: as.data.frame.gridtable(), as_calibrated(), as_calibrated.data.frame(), as_gridtable(), as_gridtable.character(), as_gridtable.data.frame(), kbl(), kbl.calibrated(), kbl.gridtable(), print.gridtable()

---

kbl.gridtable *Render a gridtable with kbl*

Description

Coerce a gridtable to a calibrated data frame with `as.data.frame` before redispatching to `kbl`. This lets gridtable-derived column widths be consumed by `kbl.calibrated`.

Usage

```
## S3 method for class 'gridtable'
kbl(x, ...)
```

Arguments

x	Character scalar with class `gridtable`.
...	Additional arguments passed to `kbl`.

Value

A kable object.

See Also

kbl, kbl.calibrated, as.data.frame.gridtable

Other gridtable: as.data.frame.gridtable(), as_calibrated(), as_calibrated.data.frame(), as_gridtable(), as_gridtable.character(), as_gridtable.data.frame(), kbl(), kbl.calibrated(), kbl.default(), print.gridtable()

kbl.tablet *Render tablet with kbl()*

Description

Renders a tablet. Calls kbl and implements special features like grouped columns. See also tablet.data.frame. As of 0.9.0, calling kbl is equivalent to calling as_kable.

Usage

```
## S3 method for class 'tablet'
kbl(
  x,
  ...,
  booktabs = TRUE,
  escape = FALSE,
  escape_latex = tablet::escape_latex,
  escape_html = function(x, ...) x,
  variable = " ",
  col.names = NA,
  linebreak = TRUE,
  align = "c",
  double_escape = FALSE,
  linebreaker = "\n",
  pack_rows = list(escape = escape),
  secondary = FALSE
)
```

Arguments

x	tablet
...	passed to kbl
booktabs	passed to kbl
escape	passed to kbl; defaults FALSE to allow header linebreaks
escape_latex	a function to pre-process column names and content if 'escape' is FALSE (e.g., manual escaping, latex only); default escape_latex
escape_html	a function to pre-process column names and content if 'escape' is FALSE (e.g., manual escaping, html only)
variable	a column name for the variables
col.names	passed to kbl after any linebreaking
linebreak	whether to invoke linebreak for column names
align	passed to linebreak for column names
double_escape	passed to linebreak for column names

linebreaker	passed to linebreak for column names in latex; for html, linebreaker is replaced with
pack_rows	named list passed to pack_rows for finer control of variable names
secondary	passed to escape_latex

Value

like kbl

Examples

```
library(boot)
library(dplyr)
library(magrittr)
library(haven)
library(yamlet)
library(spork)
melanoma %>%
  select(-time, -year) %>%
  mutate(sex = factor(sex), ulcer = factor(ulcer)) %>%
  group_by(status) %>%
  tablet %>%
  kbl

x <- system.file(package = 'tablet', 'shiny-examples/mesa/data/adsl.sas7bdat')
x %<>% read_sas %>% data.frame
decorations(x)

# calculate BMI by assuming all males are 1.75 m, all females 1.63 cm
x %<>% mutate(height = ifelse(sex == 'F', 1.63, 1.75))
x %<>% mutate(bmi = signif(digits = 3, weight / (height^2)))
x %<>% filter(saffl == 'Y')
x %<>% select(trt01a, age, sex, weight, bmi)
x %<>% redecorate('
trt01a: [ Treatment, [ Placebo, TRT 10 mg, TRT 20 mg ]]
age:    [ Age, year ]
sex:    [ Sex, [ Female: F, Male: M ]]
weight: [ Body Weight, kg ]
bmi:    [ Index_body mass, kg/m^2 ]
')
x %<>% resolve
x %<>% group_by(trt01a)

x %>% tablet %>% kbl

# supply default and unit-conditional latex titles
x %<>% modify(title = concatenate(as_latex(as_spork(c(.data$label)))))
x %<>% modify(
age, weight, bmi,
  title = concatenate(
    sep = '',  # default ok in pdf
    as_latex(
```

```
        as_spork(
          c(.data$label, ' (', .data$units, ')')
        )
      )
    )
  )
x %>% tablet %>% kbl
```

`listado` *CDISC Listing Generator*

Description

Launch `listado`, a companion Shiny application to `mesa` for building CDISC-style long listings. Point the app at a data file (`*.xpt`, `*.sas7bdat`, or `*.csv`), optionally choose up to two facet columns to break the listing into headed sections, drag columns into one or more "bins" to collapse them into single display columns, adjust column widths, set fixed header/footer text, and generate a `.qmd` file that renders the listing with `kableExtra::kbl()`. By default the generated `.qmd` is also rendered to PDF using the `quarto` command line tool (must be on `PATH`).

Usage

```
listado(launch.browser = TRUE, display.mode = "normal", ...)
```

Arguments

`launch.browser`	passed to `runApp`
`display.mode`	passed to `runApp`
`...`	passed to `runApp`

Details

Session configuration (facets, bins, widths, headers/footers, output directory) can be saved to and loaded from a sibling `.conf` file, matched to the data file by basename – e.g. choosing `adae.xpt` loads `adae.conf` if it exists, and creates a default one if it does not.

The `adae.xpt` dataset under 'examples/' is a subset of the CDISC pilot ADAE dataset.

Value

used for side effects: launches shiny application

`mesa` *Drag-and-drop Descriptive Statistics*

Description

Generate a table of descriptive statistics by selecting columns from a file. Currently supported formats include *.xpt, *.sas7bdat, and *.csv. Launch the application using `mesa()` and use the interface to select a data file, such as 'mtcars.xpt' under 'examples/') (or select configuration file 'mtcars.conf' under 'examples/'). Then classify the columns of interest to generate the corresponding displays.

Usage

```
mesa(launch.browser = TRUE, display.mode = "normal", ...)
```

Arguments

`launch.browser`	passed to `runApp`
`display.mode`	passed to `runApp`
`...`	passed to `runApp`

Details

Currently,

* xpt files are read using the defaults for `read.xport`,

* sas7bdat files are read using the defaults for `read_sas`, and

* csv files are read using the defaults for `as.csv`.

If a file in the same directory has a corresponding base name but a `.yaml` extension, it is treated as metadata and an attempt is made to apply it to the internal version of the data. This file will not be over-written, but it WILL be constructed if missing. You can hand-edit it to supply metadata. See `?yamlet` for format; see the Variables tab for an easy interface.

This is a metadata-driven application. Columns in the data that are *not* in the metadata will be ignored, and columns in the metadata that are *not* in the data will be constructed (maybe *all* of them).

The `mtcars` datasets in the 'examples' volume is from **datasets**.

Value

used for side effects: launches shiny application shinyWidgets',

print.gridtable *Print a gridtable*

Description

Print a gridtable as plain grid-table text.

Usage

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

Arguments

x	Character scalar with class gridtable.
...	Currently unused.

Value

Invisibly returns x.

See Also

Other gridtable: as.data.frame.gridtable(), as_calibrated(), as_calibrated.data.frame(), as_gridtable(), as_gridtable.character(), as_gridtable.data.frame(), kbl(), kbl.calibrated(), kbl.default(), kbl.gridtable()

recap.knitr_kable *Recap knitr_kable.*

Description

Recaps knitr_kable. Specifically, it replaces the first non-tabled caption with multicolumn text. The intent is to prevent repeat bookmarks when generating pdf.

Usage

```
## S3 method for class 'knitr_kable'
recap(x, cols = NULL, pos = "c", ...)
```

Arguments

x	object of dispatch
cols	number of columns to span; guesses ncol(x) by default
pos	position of text: 'l','c' (default), or 'r'
...	ignored

See Also

Other recap: recap()

---

tablet.data.frame *Generate a Tablet for Data Frame*

Description

Generates a 'tablet': a summary table of formatted statistics for factors (is.factor()) and numerics (is.numeric()) in x, with and without grouping variables (if present, see group_by). Column names represent finest level of grouping, distinguished by attribute 'nest' (the values of higher other groups, if any) along with the 'all' column for ungrouped statistics. Column attribute 'n' indicates relevant corresponding observation count. Input should not have column names beginning with '_tablet'.

Usage

```
## S3 method for class 'data.frame'
tablet(
 x,
 ...,
 na.rm = FALSE,
 all = 'All',
 fun = list(
  mis ~ sum(is.na(x)),
  sum ~ sum(x,  na.rm = TRUE),
  pct ~ signif(digits = 3,     sum / n * 100        ),
  ave ~ signif(digits = 3,    mean(x,  na.rm = TRUE)),
  std ~ signif(digits = 3,      sd(x,  na.rm = TRUE)),
  med ~ signif(digits = 3,  median(x,  na.rm = TRUE)),
  min ~ signif(digits = 3,     min(x,  na.rm = TRUE)),
  max ~ signif(digits = 3,     max(x,  na.rm = TRUE))
 ),
 fac = list(
  ` ` ~ sum + ' (' + pct + '%' + ')'
 ),
 num = list(
  `Mean (SD)` ~ ave + ' (' + std + ')',
  `Median (range)` ~ med + ' (' + min + ', ' + max + ')',
  `Missing` ~ mis
  ),
 lab = list(
  lab ~ name + '\n(N = ' + n + ')'
 ),
 na.rm_fac = na.rm,
 na.rm_num = na.rm,
```

```
 exclude_fac = NULL,
 exclude_name = NULL,
 all_levels = FALSE
)
```

Arguments

x	data.frame (possibly grouped)
...	substitute formulas for elements of fun, fac, num, lab; if RHS is NULL, element is removed
na.rm	whether to remove NA in general
all	a column name for ungrouped statistics; can have length zero to suppress ungrouped column
fun	default aggregate functions expressed as formulas
fac	a list of formulas to generate widgets for factors
num	a list of formulas to generate widgets for numerics
lab	a list of formulas to generate label attributes for columns (see details)
na.rm_fac	whether to drop NA 'factor' observations; passed to gather as na.rm, interacts with exclude_fac
na.rm_num	whether to drop NA numeric observations; passed to gather as na.rm
exclude_fac	which factor levels to exclude; see factor (exclude)
exclude_name	whether to drop NA values of column name (for completeness); passed to gather
all_levels	whether to supply records for unobserved levels

Details

Arguments 'fun', 'fac', 'num', and 'lab' are lists of two-sided formulas that are evaluated in an environment where '+' expresses concatenation (for character elements). The values of LHS should be unique across all four lists. 'fun' is a list of aggregate statistics that have access to N (number of original records), n (number of group members), and x (the numeric observations, or 1 for each factor value). Aggregate statistics generated by 'fun' are available for use in 'fac' and 'num' which create visualizations thereof ('widgets'). Column-specific attributes are available to elements of 'lab', including the special attribute name (the current column name). For 'lab' only, if the RHS succeeds, it becomes the label attribute of the corresponding output column. 'lab' is used here principally to support annotation of *output* columns; if *input* columns have attributes 'label' or 'title' (highest priority) those will have been already substituted for default column names at the appropriate positions in the output.

Missingness of observations (and to a lesser extent, levels of grouping variables) merits special consideration. Be aware that na.rm_fac and na.rm_num take their defaults from na.rm. Furthermore, na.rm_fac may interact with exclude_fac, which is passed to factor as exclude. The goal is to support all possible ways of expressing or ignoring missingness. That said, if aggregate functions are removing NA, the values of arguments beginning with 'na.rm' or 'exclude' may not matter.

Value

'tablet' A tablet is a special case of data.frame with grouped rows and columns.

* There is always one level of row groups.
* There can be any number of column groups, including zero.
* All columns are character (as tested by `is.character()`).
* The first column has empty strings that represent the last non-empty value. It can be class 'latex' or 'character'.
* Leading element(s) of first column are deliberately blank (one space character) and correspond to header rows. See `header_rows`.
* The second column represents group-specific property names. It is populated always and only where column 1 is not.
* All other columns represent group-specific property values; elements before the first non-empty group value represent nested headers.
* Header values may be repeated.
* Header values may be empty strings, representing the last non-empty value to the left, or single spaces, which are deliberately blank.
* Internally, character NA is equivalent to an empty string.

See Also

`as_kable.tablet` `kbl.tablet`

Examples

```
library(boot)
library(dplyr)
library(magrittr)
melanoma %>%
  select(-time, -year) %>%
  mutate(sex = factor(sex), ulcer = factor(ulcer)) %>%
  group_by(status) %>%
  tablet %>%
  kbl

## drop 'Missing', redefine 'range'
melanoma %>%
  select(-time, -year) %>%
  mutate(sex = factor(sex), ulcer = factor(ulcer)) %>%
  group_by(status) %>%
  tablet(
    Missing ~ NULL,
    `Median (range)` ~ med + ' (' + min + ' to ' + max + ')'
  ) %>%
  kbl
```

Index

∗ **gridtable**
  as.data.frame.gridtable, 2
  as_calibrated.data.frame, 3
  as_gridtable.character, 4
  as_gridtable.data.frame, 5
  kbl, 10
  kbl.calibrated, 11
  kbl.default, 11
  kbl.gridtable, 13
  print.gridtable, 18
∗ **io**
  io_tablet, 9
∗ **recap**
  recap.knitr_kable, 18
∗ **tablet**
  as_tablet.data.frame, 8
  header_rows.tablet, 8

as.csv, *17*
as.data.frame.gridtable, 2, *3–5*, *10, 11*, *13*, *18*
as_calibrated, *3–5*, *10, 11*, *13*, *18*
as_calibrated.data.frame, *3*, 3, *4, 5*, *10, 11*, *13*, *18*
as_gridtable, *3–5*, *10, 11*, *13*, *18*
as_gridtable.character, *3*, 4, *5*, *10, 11*, *13*, *18*
as_gridtable.data.frame, *3, 4*, 5, *10, 11*, *13*, *18*
as_kable, *5*, *14*
as_kable.tablet, 5, *21*
as_tablet, *8, 9*
as_tablet.data.frame, 8, *9*

column_spec, *3*, *11*

escape_latex, *6*, *14*

factor, *20*

gather, *20*

group_by, *19*

header_rows, *8, 9*, *21*
header_rows.tablet, *8*, 8

io_csv, *9*
io_csv.tablet, *9*
io_tablet, 9
io_tablet.character, *9*
io_tablet.data.frame, *9*
io_tablet.tablet, *9*

kbl, *3–6*, *10*, 10, *11–15*, *18*
kbl.calibrated, *3–5*, *10*, 11, *13*, *18*
kbl.default, *3–5*, *10, 11*, 11, *13*, *18*
kbl.gridtable, *3–5*, *10, 11*, *13*, 13, *18*
kbl.tablet, 14, *21*

linebreak, *6*, *14, 15*
listado, 16

mesa, *16*, 17
mtcars, *17*

pack_rows, *6*, *15*
print.gridtable, *3–5*, *10, 11*, *13*, 18

read.xport, *17*
read_sas, *17*
recap, *19*
recap.knitr_kable, 18
runApp, *16, 17*

tablet, *6*, *14*
tablet.data.frame, *5*, *8*, *14*, 19
tablet.tablette, *8, 9*
tablette.tablet, *8, 9*