Package ‘kableExtra’

October 13, 2022

Type Package

Title Construct Complex Table with 'kable' and Pipe Syntax

Version 1.3.4

Description Build complex HTML or 'LaTeX' tables using 'kable()' from 'knitr' and the piping syntax from 'magrittr'. Function 'kable()' is a light weight table generator coming from 'knitr'. This package simplifies the way to manipulate the HTML or 'LaTeX' codes generated by 'kable()' and allows users to construct complex tables and customize styles using a readable syntax.

License MIT + file LICENSE

LazyData TRUE

URL http://haozhu233.github.io/kableExtra/,
https://github.com/haozhu233/kableExtra

BugReports https://github.com/haozhu233/kableExtra/issues

Depends R (>= 3.1.0)

Imports knitr (>= 1.16), magrittr, stringr (>= 1.0), xml2 (>= 1.1.1), rvest, rmarkdown (>= 1.6.0), scales, viridisLite, stats, grDevices, htmltools, rstudioapi, glue, tools, webshot, digest, graphics, svglite

Suggests testthat, magick, formatable, sparkline

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Hao Zhu [aut, cre] (https://orcid.org/0000-0002-3386-6076),
Thomas Travison [ctb],
Timothiy Tsai [ctb],
Will Beasley [ctb],
Yihui Xie [ctb],
GuangChuang Yu [ctb],
R topics documented:

Stéphane Laurent [ctb],
Rob Shepherd [ctb],
Yoni Sidi [ctb],
Brian Salzer [ctb],
George Gui [ctb],
Yeliang Fan [ctb],
Duncan Murdoch [ctb],
Bill Evans [ctb]

Maintainer Hao Zhu <haozhu233@gmail.com>

Repository CRAN

Date/Publication 2021-02-20 05:50:07 UTC

R topics documented:

kableExtra-package .................................................. 3
add_footnote .......................................................... 4
add_header_above .................................................... 5
add_indent ............................................................. 7
as_image ............................................................... 7
auto_index ............................................................. 8
cell_spec .............................................................. 9
collapse_rows ......................................................... 11
column_spec .......................................................... 12
footnote ................................................................. 14
footnote_marker_number .............................................. 16
graphics_helpers ..................................................... 17
group_rows ............................................................ 18
header_separate ....................................................... 20
html_dependency_bsTable ........................................... 21
html_dependency_kePrint ........................................... 21
html_dependency_lightable ......................................... 21
kableExtra_latex_packages ........................................ 22
kable_as_image ....................................................... 22
kable_as_xml .......................................................... 23
kable_classic .......................................................... 23
kable_styling ......................................................... 24
kbl ............................................................... 27
landscape ............................................................ 29
linebreak ............................................................. 30
listify_args .......................................................... 30
magic_mirror ........................................................ 31
make_inline_plot .................................................... 32
remove_column ......................................................... 32
rmd_format ........................................................... 33
row_spec ............................................................. 33
save_kable ........................................................... 35
scroll_box ............................................................ 36
Description

When we are talking about table generators in R, *knitr*’s *kable()* function wins lots of flavor by its ultimate simplicity. Unlike those powerful table rendering engines such as *xtable*, the philosophy behind *knitr::kable()* is to make it easy for programmers to use. Just as it claimed in its function description, "this is a very simple table generator. It is simple by design. It is not intended to replace any other R packages for making tables. - Yihui".

However, the ultimate simplicity of *kable()* also brought troubles to some of us, especially for new R users, who may not have a lot of experience on generating tables in R. It is not rare to see people including experienced users asking questions like how to center/left-align a table on Stack Overflow. Also, for me personally, I found myself repeatedly parsing CSS into *kable()* for some very simple features like striped lines. For *LaTeX*, it’s even worse since I’m almost Stack Overflow dependent for *LaTeX*... That’s why this package *kableExtra* was created.

I hope with *kableExtra*, you can

- Use default base *kable()* (Or a good alternative for markdown tables is *pander::pander()* for all simple tables
- Use *kable()* with *kableExtra* to generate 90% of complex/advanced tables in either HTML or *LaTeX*
- Only have to mess with raw HTML/LaTeX in the last 10% cases where *kableExtra* cannot solve the problem

For a full package documentation, please visit the package documentation site for more information
Features

**Pipable syntax:** `kableExtra` is NOT a table generating package. It is a package that can "add features" to a `kable` output using a syntax that every user loves - the `pipe`. We see similar approaches to deal with plots in packages like `ggvis` and `plotly`. There is no reason why we cannot use it with tables.

**Unified functions for both HTML and PDF:** Most functionalities in `kableExtra` can work in both HTML and PDF. In fact, as long as you specify format in `kable` (which can be set globally through `option(knitr.table.format)`), functions in this package will pick the right way to manipulate the table by themselves. As a result, if users want to left align the table, `kable_styling(kable(...), position = "left")` will work in both HTML and PDF.

Note

If you found a feature on the documentation site that is not available in the version of `kableExtra` you are using, try to install the pre-release version from github. You can do so by running `devtools::install_github("haozhu233/kableExtra")`.

Also, note that This package can load required LaTeX package automatically in vanilla rmarkdown. For customized rmarkdown templates, it is recommended to load related LaTeX packages manually.

---

**add_footnote**

Add footnote to your favorite `kable` output.

**Usage**

```r
add_footnote(
  input,
  label = NULL,
  notation = "alphabet",
  threeparttable = FALSE,
  escape = TRUE
)
```

**Arguments**

- `input`: The direct output of your `kable` function or your last `kableExtra` function.
- `label`: A vector of footnotes you want to add. You don’t need to add notations in your notes.
- `notation`: You can select the format of your footnote notation from number, alphabet, symbol and none.
- `threeparttable`: Boolean value indicating if a threeparttable scheme should be used.
- `escape`: Logical value controlling if the label needs to be escaped. Default is TRUE.
add_header_above  Add a header row on top of current header

Examples

```r
## Not run:
x <- knitr::kable(head(mtcars), "html")
add_footnote(x, c("footnote 1", "footnote 2"), notation = "symbol")

## End(Not run)
```

Description

Tables with multiple rows of header rows are extremely useful to demonstrate grouped data. This function takes the output of a `kable()` function and adds an header row on top of it.

Usage

```r
add_header_above(
  kable_input,
  header = NULL,
  bold = FALSE,
  italic = FALSE,
  monospace = FALSE,
  underline = FALSE,
  strikeout = FALSE,
  align = "c",
  color = NULL,
  background = NULL,
  font_size = NULL,
  angle = NULL,
  escape = TRUE,
  line = TRUE,
  line_sep = 3,
  extra_css = NULL,
  include_empty = FALSE,
  border_left = FALSE,
  border_right = FALSE
)
```

Arguments

- `kable_input` Output of `knitr::kable()` with format specified
- `header` A (named) character vector with `colspan` as values. For example, `c(" " = 1, "title" = 2)` can be used to create a new header row for a 3-column table with "title" spanning across column 2 and 3. For convenience, when `colspan` equals to 1, users can drop the `= 1` part. As a result, `c(" ", "title" = 2)` is the same as
add_header_above

Alternatively, a data frame with two columns can be provided. The first column should contain the header names (character vector) and the second column should contain the colspan (numeric vector). This input can be used if there are problems with unicode characters in the headers.

bold  A T/F value to control whether the text should be bolded.
italic A T/F value to control whether the text should be emphasized.
monospace A T/F value to control whether the text of the selected column need to be monospaced (verbatim)
underline A T/F value to control whether the text of the selected row need to be underlined
strikeout A T/F value to control whether the text of the selected row need to be stricked out.
align A character string for cell alignment. For HTML, possible values could be l, c, r plus left, center, right, justify, initial and inherit while for LaTeX, you can only choose from l, c & r.
color A character string/vector for text color. Here please pay attention to the differences in color codes between HTML and LaTeX.
background A character string/vector for background color. Here please pay attention to the differences in color codes between HTML and LaTeX. Also note that in HTML, background defined in cell_spec won’t cover the whole cell.
font_size A numeric input/vector for font size. For HTML, you can also use options including xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, initial and inherit.
angle 0-360, degree that the text will rotate.
escape A T/F value showing whether special characters should be escaped.
line A T/F value to control whether a line will appear underneath the header
line_sep A numeric value indicating how much the midlines should be separated by space. Default is 3.
extra_css An HTML only option. CSS defined here will be send to the td cell.
include_empty Whether empty cells in HTML should also be styled. Default is FALSE.
border_left T/F option for border on the left side in latex.
border_right T/F option for border on the right side in latex.

Examples

## Not run:
x <- knitr::kable(head(mtcars), "html")
# Add a row of header with 3 columns on the top of the table. The column # span for the 2nd and 3rd one are 5 & 6.
add_header_above(x, c(" ", "Group 1" = 5, "Group 2" = 6))

## End(Not run)
**add_indent**

Add indentations to row headers

**Usage**

add_indent(kable_input, positions, level_of_indent = 1, all_cols = FALSE)

**Arguments**

- `kable_input`: Output of `knitr::kable()` with format specified
- `positions`: A vector of numeric row numbers for the rows that need to be indented.
- `level_of_indent`: A numeric value for the indent level. Default is 1.
- `all_cols`: T/F whether to apply indentation to all columns

**Examples**

```r
## Not run:
x <- knitr::kable(head(mtcars), "html")
# Add indentations to the 2nd & 4th row
add_indent(x, c(2, 4), level_of_indent = 1)
## End(Not run)
```

**as_image**

Render the table as an format-independent image and use it in rmarkdown

**Description**

This function generates a temporary png file using save_kable and then try to put it in an rmarkdown document using knitr::include_graphics.

**Usage**

as_image(x, width = NULL, height = NULL, file = NULL, ...)
Argument

- **x**
  - kable input. Either HTML or LaTeX

- **width**
  - Image width in inches. (1 inch = 2.54 cm)

- **height**
  - Image height in inches. (1 inch = 2.54 cm)

- **file**
  - By default, as_image saves to a temp file, which works for normal rmarkdown. However if you are using things like xaringan, which can’t be a standalone html, you can specify this file be the path you need, eg. “img/something.png”

Examples

```r
## Not run:
library(kableExtra)

kable(mtcars, "latex", booktabs = T) %>%
kable_styling(latex_options = c("striped", "scale_down")) %>%
row_spec(1, color = "red") %>%
as_image()

## End(Not run)
```

Description

This helper function allows users to build the `group_row` index more quickly and use `group_rows` in a way that is similar with `collapse_rows`.

Usage

```r
auto_index(x)
```

Arguments

- **x**
  - The index column. A vector. For example `c("a", "a", "b", "b", "b")`
cell_spec

Specify Cell/Text format

Description

Specify Cell format before it gets into kable

Usage

```r
cell_spec(
  x,
  format,
  bold = FALSE,
  italic = FALSE,
  monospace = FALSE,
  underline = FALSE,
  strikeout = FALSE,
  color = NULL,
  background = NULL,
  align = NULL,
  font_size = NULL,
  angle = NULL,
  tooltip = NULL,
  popover = NULL,
  link = NULL,
  new_tab = FALSE,
  extra_css = NULL,
  escape = TRUE,
  background_as_tile = TRUE,
  latex_background_in_cell = TRUE
)
```

```r
text_spec(
  x,
  format,
  bold = FALSE,
  italic = FALSE,
  monospace = FALSE,
  underline = FALSE,
  strikeout = FALSE,
  color = NULL,
  background = NULL,
  align = NULL,
  font_size = NULL,
  angle = NULL,
  tooltip = NULL,
  popover = NULL,
  popover = NULL,
```
cell_spec

link = NULL,
new_tab = FALSE,
extra_css = NULL,
escape = TRUE,
background_as_tile = TRUE,
latex_background_in_cell = FALSE)

Arguments

x Things to be formatted. It could be a vector of numbers or strings.
format Either "html" or "latex". It can also be set through option(knitr::table_format), same as knitr::kable().
bold T/F for font bold.
italic T/F for font italic.
monospace T/F for font monospaced (verbatim)
underline A T/F value to control whether the text of the selected row need to be underlined
strikeout A T/F value to control whether the text of the selected row need to be stricked out.
color A character string for text color. Here please pay attention to the differences in color codes between HTML and LaTeX.
background A character string for background color. Here please pay attention to the differences in color codes between HTML and LaTeX. Also note that in HTML, background defined in cell_spec won’t cover the whole cell.
align A character string for cell alignment. For HTML, possible values could be l, c, r plus left, center, right, justify, initial and inherit while for LaTeX, you can only choose from l, c & r.
font_size A numeric input for font size. For HTML, you can also use options including xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, initial and inherit.
angle 0-360, degree that the text will rotate. Can be a vector.
tooltip A vector of strings to be displayed as tooltip. Obviously, this feature is only available in HTML. Read the package vignette to see how to use bootstrap tooltip css to improve the loading speed and look.
popover Similar with tooltip but can hold more contents. The best way to build a popover is through spec_popover(). If you only provide a text string, it will be used as content. Note that You have to enable this bootstrap module manually. Read the package vignette to see how.
link A vector of strings for url links. Can be used together with tooltip and popover.
new_tab T/F for whether to open up the new link in new tab.
extra_css Extra css text to be passed into the cell
escape T/F value showing whether special characters should be escaped.
collapse_rows

background_as_tile
T/F value indicating if you want to have round cornered tile as background in HTML.

latex_background_in_cell
T/F value. It only takes effect in LaTeX when background provided. Default value is TRUE. If it’s TRUE, the background only works in a table cell. If it’s FALSE, it works outside of a table environment.

Description
Collapse same values in columns into multirow cells. This feature does similar things with group_rows. However, unlike group_rows, it analyzes existing columns, finds out rows that can be grouped together, and make them multirow cells. Note that if you want to use column_spec to specify column styles, you should use column_spec before collapse_rows.

Usage
```r
collapse_rows(
  kable_input,
  columns = NULL,
  valign = c("middle", "top", "bottom"),
  latex_hline = c("full", "major", "none", "custom", "linespace"),
  row_group_label_position = c("identity", "stack"),
  custom_latex_hline = NULL,
  row_group_label_fonts = NULL,
  headers_to_remove = NULL,
  target = NULL,
  col_names = TRUE,
  longtable_clean_cut = TRUE
)
```

Arguments
- `kable_input`: Output of knitr::kable() with format specified
- `columns`: A numeric value or vector indicating in which column(s) rows need to be collapsed.
- `valign`: Select from "top", "middle"(default), "bottom". The reason why "top" is not default is that the multirow package on CRAN win-builder is not up to date.
- `latex_hline`: Option controlling the behavior of adding hlines to table. Choose from full, major, none, custom and linespace.
- `row_group_label_position`: Option controlling positions of row group labels. Choose from identity, stack.
column_spec

Specifications:

- `column_spec(kable_input, column, width = NULL, bold = FALSE, italic = FALSE, monospace = FALSE, underline = FALSE, strikeout = FALSE, color = NULL)`

Descriptions:

- **custom_latex_hline**: Numeric column positions whose collapsed rows will be separated by hlines.
- **row_group_label_fonts**: A list of arguments that can be supplied to group_rows function to format the row group label when `row_group_label_position` is `stack`.
- **headers_to_remove**: Numeric column positions where headers should be removed when they are stacked.
- **target**: If multiple columns are selected to do collapsing and a target column is specified, this target column will be used to collapse other columns based on the groups of this target column.
- **col_names**: T/F. A LaTeX specific option. If you set `col.names` be `NULL` in your `kable` call, you need to set this option false to let everything work properly.
- **longtable_clean_cut**: T/F with default T. Multirow cell sometimes are displayed incorrectly around pagebreak. This option forces groups to cut before the end of a page. If you have a group that is longer than 1 page, you need to turn off this option.

Examples:

```r
## Not run:
dt <- data.frame(a = c(1, 1, 2, 2), b = c("a", "a", "a", "b"))
x <- knitr::kable(dt, "html")
collapse_rows(x)
## End(Not run)
```

Description:

This function allows users to select a column and then specify its look.

Usage:

```r
column_spec(kable_input, column, width = NULL, bold = FALSE, italic = FALSE, monospace = FALSE, underline = FALSE, strikeout = FALSE, color = NULL)
```
Arguments

kable_input  Output of knitr::kable() with format specified

column  A numeric value or vector indicating which column(s) to be selected.

width  A character string telling HTML & LaTeX how wide the column needs to be, e.g. "10cm", "3in" or "30em".

bold  T/F value or vector to control whether the text of the selected column need to be bolded.

italic  T/F value or vector to control whether the text of the selected column need to be emphasized.

monospace  T/F value or vector to control whether the text of the selected column need to be monospaced (verbatim)

underline  T/F value or vector to control whether the text of the selected row need to be underlined

strikeout  T/F value or vector to control whether the text of the selected row need to be striked out.

color  A character string or vector for column text color. Here please pay attention to the differences in color codes between HTML and LaTeX.

background  A character string or vector for column background color. Here please pay attention to the differences in color codes between HTML and LaTeX.

border_left  A logical variable indicating whether there should be a border line on the left of the selected column. In HTML, you can also pass in a character string for the CSS of the border line

border_right  A logical variable indicating whether there should be a border line on the right of the selected column. In HTML, you can also pass in a character string for the CSS of the border line

width_min  Only for HTML table. Normal column width will automatically collapse when the window cannot hold enough contents. With this width_min, you can set up a column with a width that won’t collapse even when the window is not wide enough.
width_max  Only for HTML table. width_max defines the maximum width of table columns.
extra_css  A vector of extra css text to be passed into the cells of the column.
include_thead  T/F. A HTML only feature to contoll whether the header row will be manipulated. Default is FALSE.
latex_column_spec  Only for LaTeX tables. Code to replace the column specification. If not NULL, will override all other arguments.
latex_valign  vertical alignment. Only works when you specified column width. Choose among p, m, b.
link  A vector of strings for url links.
new_tab  T/F for whether to open up the new link in new tab
tooltip  A vector of strings to be displayed as tooltip. Obviously, this feature is only available in HTML. Read the package vignette to see how to use bootstrap tooltip css to improve the loading speed and look.
popover  Similar with tooltip but can hold more contents. The best way to build a popover is through spec_popover(). If you only provide a text string, it will be used as content. Note that You have to enable this bootstrap module manually. Read the package vignette to see how.
image  Vector of image paths.

Details
Use latex_column_spec in a LaTeX table to change or customize the column specification. Because of the way it is handled internally, any backslashes must be escaped.

Examples
```r
## Not run:
x <- knitr::kable(head(mtcars), "html")
column_spec(x, 1:2, width = "20em", bold = TRUE, italic = TRUE)
x <- knitr::kable(head(mtcars), "latex", booktabs = TRUE)
column_spec(x, 1, latex_column_spec = ">{\\color{red}}c")
## End(Not run)
```

footnote  Add footnote (new)

Description
footnote provides a more flexible way to add footnote. You can add mutiple sets of footnote using differeny notation system. It is also possible to specify footnote section header one by one and print footnotes as a chunk of texts.
footnote

Usage

footnote(
  kable_input,
  general = NULL,
  number = NULL,
  alphabet = NULL,
  symbol = NULL,
  footnote_order = c("general", "number", "alphabet", "symbol"),
  footnote_as_chunk = FALSE,
  escape = TRUE,
  threeparttable = FALSE,
  fixed_small_size = FALSE,
  general_title = "Note: ",
  number_title = ",",
  alphabet_title = ",",
  symbol_title = ",",
  title_format = "italic",
  symbol_manual = NULL
)

Arguments

kable_input HTML or LaTeX table generated by knitr::kable
general Text for general footnote comments. Footnotes in this section won’t be labeled with any notations
number A vector of footnote texts. Footnotes here will be numbered. There is no upper cap for the number of footnotes here
alphabet A vector of footnote texts. Footnotes here will be labeled with abc. The vector here should not have more than 26 elements.
symbol A vector of footnote texts. Footnotes here will be labeled with special symbols. The vector here should not have more than 20 elements.
footnote_order The order of how to arrange general, number, alphabet and symbol.
footnote_as_chunk T/F value. Default is FALSE. It controls whether the footnotes should be printed in a chunk (without line break).
escape T/F value. It controls whether the contents and titles should be escaped against HTML or LaTeX. Default is TRUE.
threeparttable T/F value for whether to use LaTeX package threeparttable. Threeparttable will force the width of caption and footnotes be the width of the original table. It’s useful when you have long paragraph of footnotes.
fixed_small_size T/F When you want to keep the footnote small after specifying large font size with the kable_styling() (e.g. ideal font for headers and table content with small font in footnotes).
general_title Section header for general footnotes. Default is "Note: ".

footnote_marker_number

---

**Footnote marker**

---

### Description

Put footnote mark in superscription in table. Unless you are using it in the caption of `kable`, you will need to put `escape = F` in `kable` (similar with `cell_spec`). Again, similar with `cell_spec`, the `format` option here can read default value from global option `knitr.table.format`.

### Usage

```r
footnote_marker_number(x, format, double_escape = FALSE)
footnote_marker_alphabet(x, format, double_escape = FALSE)
footnote_marker_symbol(x, format, double_escape = FALSE)
```

### Arguments

- `x` a number. For example, for `footnote_marker_alphabet(2)` will return "b" in HTML.
- `format` Either html or latex. All functions here can read default value from global option `knitr.table.format`.
- `double_escape` T/F if output is in LaTeX, whether it should be double escaped. If you are using `footnote_marker` in `group_rows`` labeling row or `add_header_above`, you need to set this to be
Examples

```r
## Not run:
dt <- mtcars[1:5, 1:5]
colnames(dt)[1] <- paste0("mpg", footnote_marker_alphabet(2, "html"))
rownames(dt)[2] <- paste0(rownames(dt)[2], footnote_marker_alphabet(1, "html"))
footnote(knitr::kable(dt, "html"), alphabet = c("Note a", "Note b"))

## End(Not run)
```

graphics_helpers  

**Helper functions to use various graphics devices**

Description

These helper functions generalize the use of strings (e.g., "svg", "pdf") or graphic device functions (e.g., grDevices::svg, grDevices::pdf) for in-table plots.

Usage

```r
graphics_dev(filename, width, height, res, ..., dev)

is_svg(dev)

dev_chr(dev)
```

Arguments

- `filename`: Passed through to the graphics device.
- `width, height`: Plot dimensions in pixels.
- `res`: The resolution of the plot; default is 300.
- `...`: extra parameters passing to the graphics-device function.
- `dev`: Character (e.g., "svg", "pdf") or function (e.g., grDevices::svg, grDevices::pdf).

Details

- `graphics_dev` generalizes the use of 'res' and plot dimensions across graphic devices. Raster-based devices (e.g., 'png', 'jpeg', 'tiff', 'bmp') tend to use 'res' and the width/height units default to pixels. All other devices (e.g., 'pdf', 'svg') tend to use inches as the default units for width/height, and error when 'res' is provided.

  The current heuristic is the look for the 'res' argument in the function’s formals; if that is present, then it is assumed that the default units are in pixels, so 'width', 'height', and 'res' are passed through unmodified. If 'res' is not present, then 'width' and 'height' are converted from pixels to inches, and 'res' is not passed to the function.

  Another purpose of this function is to generalize the different graphic functions’ use of 'file=' versus 'filename='.
• `is_svg` determines if the plot device is svg-like, typically one of "svg", `grDevices::svg`, or `svglite::svglite`

• `dev_chr` determines the filename extension for the applicable plot function; when `dev` is a string, then it is returned unchanged; when `dev` is a function, the formals of the function are checked for clues (i.e., default value of a `file=` argument)

Value

'graphics_dev': nothing, a plot device is opened
'is_svg': logical
`dev_chr`: character

Functions

• `graphics_dev`: Generalize `res` and `filename` across dev functions
• `is_svg`: Determine if plot device is svg-like
• `dev_chr`: Determine filename extension

Description

Group a few rows in a table together under a label.

Usage

```r
group_rows(  
kable_input,  
group_label = NULL,  
start_row = NULL,  
end_row = NULL,  
index = NULL,  
label_row_css = "border-bottom: 1px solid;",  
latex_gap_space = "0.3em",  
escape = TRUE,  
l latex_align = "l",  
l latex_wrap_text = FALSE,  
colnum = NULL,  
b olt = TRUE,  
italic = NULL,  
hline_before = FALSE,  
hline_after = FALSE,  
extra_latex_after = NULL,  
indent = TRUE,  
monospace = FALSE,  
)`
Arguments

**kable_input**  
Output of `knitr::kable()` with format specified

**group_label**  
A character string for the name of the group

**start_row**  
A numeric value that tells the function in which row the group starts. Note that the counting excludes header rows and other group labeling rows

**end_row**  
A numeric value that tells the function in which row the group ends.

**index**  
A named vector providing the index for robust row-grouping tasks. Basically, you can use it in the same way as `add_header_above()`.

**label_row_css**  
A character string for any customized css used for the labeling row. By default, the labeling row will have a solid black line underneath. Only useful for HTML documents.

**latex_gap_space**  
A character value telling LaTeX how large the gap between the previous row and the group labeling row. Only useful for LaTeX documents.

**escape**  
A T/F value showing whether special characters should be escaped.
latex_align | Adjust justification of group_label in latex only. Value should be "c" for centered on row, "r" for right justification, or "l" for left justification. Default Value is "l".

latex_wrap_text | T/F for wrapping long text. Default is off. Whenever it is turned on, the table will take up the entire line. It’s recommended to use this with full_width in kable_styling.

colnum | A numeric that determines how many columns the text should span. The default setting will have the text span the entire length.

bold | A T/F value to control whether the text should be bolded.

italic | A T/F value to control whether the text should be emphasized.

hline_before | A T/F value that adds a horizontal line before the group_row label. Default value is False.

hline_after | A replicate of hline_after in xtable. It adds a hline after the row.

extra_latex_after | Extra LaTeX text to be added after the row.

indent | A T/F value to control whether list items are indented.

monospace | T/F value to control whether the text of the selected column need to be monospaced (verbatim)

underline | T/F value to control whether the text of the selected row need to be underlined

strikeout | T/F value to control whether the text of the selected row need to be striked out.

color | A character string for column text color. Here please pay attention to the differences in color codes between HTML and LaTeX.

background | A character string for column background color. Here please pay attention to the differences in color codes between HTML and LaTeX.

Examples

```r
## Not run:
x <- knitr::kable(head(mtcars), "html")
# Put Row 2 to Row 5 into a Group and label it as "Group A"
pack_rows(x, "Group A", 2, 5)
```

header_separate | Separate table headers and add additional header rows based on grouping.

Description

When you create a summary table for either model or basic summary stats in R, you usually end up having column names in the form of "a_mean", "a_sd", "b_mean" and "b_sd". This function streamlines the process of renaming these column names and adding extra header rows using add_header_above.
Usage

header_separate(kable_input, sep = "[^[:alnum:]]+")

Arguments

kable_input  Output of knitr::kable() with format specified
sep            A regular expression separator between groups. The default value is a regular expression that matches any sequence of non-alphanumeric values.

html_dependency_bsTable

*HTML dependency for Twitter bootstrap (table only)*

Description

HTML dependency for Twitter bootstrap (table only)

Usage

html_dependency_bsTable()

---

html_dependency_kePrint

*HTML dependency for js script to enable bootstrap tooltip and popup message*

Description

HTML dependency for js script to enable bootstrap tooltip and popup message

Usage

html_dependency_kePrint()

---

html_dependency_lightable

*HTML dependency for lightable*

Description

HTML dependency for lightable

Usage

html_dependency_lightable()
kableExtra_latex_packages

LaTeX Packages

Description

This function shows all LaTeX packages that is supposed to be loaded for this package in a markdown yaml format.

Usage

kableExtra_latex_packages()

kable_as_image

Deprecated

Description

deprecated

Usage

kable_as_image(
  kable_input,
  filename = NULL,
  file_format = "png",
  latex_header_includes = NULL,
  keep_pdf = FALSE,
  density = 300,
  keep_tex = FALSE
)

Arguments

kable_input Raw LaTeX code to generate a table. It doesn’t have to came from kable or kableExtra.

filename Character String. If specified, the image will be saved under the specified (path & name). You don’t need to put file format like "png" here.

file_format Character String to specify image format, such as png, jpeg, gif, tiff, etc. Default is png.

latex_header_includes A character vector of extra LaTeX header stuff. Each element is a row. You can have things like c("\\usepackage{threeparttable}", "\\usepackage{icons}"). You could probably add your language package here if you use non-English text in your table, such as "\\usepackage[magyar]{babel}.}
### kable_as_xml

Read HTML kable as XML

**Description**

This function will read kable as a xml file

**Usage**

```r
kable_as_xml(x)
```

**Arguments**

- `x` kable or kableExtra object

### kable_classic

Alternative HTML themes

**Description**

kableExtra uses the built-in bootstrap themes by default in `kable_styling()`. Alternatively, you can use a customized table themes for your table. This `lightable` table style sheet comes with three formats, namely `lightable-minimal`, `lightable-classic`, `lightable-material` and `lightable-material-dark` with hover and striped options.

**Usage**

```r
kable_classic(
    kable_input,
    lightable_options = "basic",
    html_font = ""Arial Narrow\", "Source Sans Pro\", sans-serif",
    ...
)
```

```r
kable_classic_2(
    kable_input,
    lightable_options = "basic",
    html_font = ""Arial Narrow\", "Source Sans Pro\", sans-serif",
```
kable_styling

...)

kable_minimal(
    kable_input,
    lightable_options = "basic",
    html_font = "\"Trebuchet MS\", verdana, sans-serif",
    ...
)

kable_material(
    kable_input,
    lightable_options = "basic",
    html_font = "\"Source Sans Pro\", helvetica, sans-serif",
    ...
)

kable_material_dark(
    kable_input,
    lightable_options = "basic",
    html_font = "\"Source Sans Pro\", helvetica, sans-serif",
    ...
)

kable_paper(
    kable_input,
    lightable_options = "basic",
    html_font = "\"Arial Narrow\", arial, helvetica, sans-serif",
    ...
)

Arguments

kable_input       A HTML kable object.
lightable_options Options to customize lightable. Similar with bootstrap_options in kable_styling.
                    Choices include basic, striped and hover.
html_font         A string for HTML css font. For example, html_font = '"Arial Narrow",
                    arial, helvetica, sans-serif'.
...                Everything else you need to specify in kable_styling.
**kable_styling**

**Description**

This function provides a cleaner approach to modify the style of HTML tables other than using the `table.attr` option in `knitr::kable()`. Note that those bootstrap options requires Twitter bootstrap theme, which is not available in some customized template being loaded.

**Usage**

```r
kable_styling(
  kable_input,
  bootstrap_options = "basic",
  latex_options = "basic",
  full_width = NULL,
  position = "center",
  font_size = NULL,
  row_label_position = "l",
  repeat_header_text = "\textit{(continued)}",
  repeat_header_method = c("append", "replace"),
  repeat_header_continued = FALSE,
  stripe_color = "gray!6",
  stripe_index = NULL,
  latex_table_env = NULL,
  protect_latex = TRUE,
  table.envir = "table",
  fixed_thead = FALSE,
  htmltable_class = NULL,
  html_font = NULL,
  wraptable_width = "0pt"
)
```

**Arguments**

- `kable_input` Output of `knitr::kable()` with format specified
- `bootstrap_options` A character vector for bootstrap table options. Please see package vignette or visit the w3schools’ Bootstrap Page for more information. Possible options include basic, striped, bordered, hover, condensed, responsive and none.
- `latex_options` A character vector for LaTeX table options. Please see package vignette for more information. Possible options include basic, striped, hold_position, HOLD_position, scale_down & repeat_header. striped will add alternative row colors to the table. It will imports LaTeX package xcolor if enabled. hold_position will "hold" the floating table to the exact position. It is useful when the LaTeX table is contained in a table environment after you specified captions in `kable()`. It will force the table to stay in the position where it was created in the document. A stronger version: HOLD_position requires the float package and specifies [H]. scale_down is useful for super wide table. It will automatically adjust the table to page width. repeat_header in only meaningful in a longtable environment. It will let the header row repeat on every page in that long table.
full_width  A TRUE or FALSE variable controlling whether the HTML table should have 100% full width by default but this option will be set to FALSE for a LaTeX table

position  A character string determining how to position the table on a page. Possible values include left, center, right, float_left and float_right. Please see the package doc site for demonstrations. For a LaTeX table, if float_* is selected, LaTeX package wrapfig will be imported.

font_size  A numeric input for table font size

row_label_position  A character string determining the justification of the row labels in a table. Possible values include l for left, c for center, and r for right. The default value is l for left justification.

repeat_header_text  LaTeX option. A text string you want to append on or replace the caption.

repeat_header_method  LaTeX option, can either be append(default) or replace

repeat_header_continued  T/F or a text string. Whether or not to put a continued mark on the second page of longtable. If you put in text, we will use this text as the "continued" mark.

stripe_color  LaTeX option allowing users to pick a different color for their strip lines. This option is not available in HTML

stripe_index  LaTeX option allowing users to customize which rows should have stripe color.

latex_table_env  LaTeX option. A character string to define customized table environment such as tabu or tabularx. You shouldn’t expect all features could be supported in self-defined environments.

protect_latex  If TRUE, LaTeX code embedded between dollar signs will be protected from HTML escaping.

table.envir  LaTeX floating table environment. kable_style will put a plain no-caption table in a \texttt{table} environment in order to center the table. You can specify this option to things like table* or float* based on your need.

fixed_thead  HTML table option so table header row is fixed at top. Values can be either T/F or list(\texttt{enabled} = T/F, \texttt{background} = \texttt{"anycolor"}).

htmltable_class  Options to use the in-house lightable themes. Choices include lightable-minimal, lightable-classic, lightable-classic-2, lightable-material, lightable-striped and lightable-hover. If you have your customized style sheet loaded which defines your own table class, you can also load it here.

html_font  A string for HTML css font. For example, \texttt{html_font = "Arial Narrow", arial, helvetica, sans-serif"}.

wraptable_width  Width of the wraptable area if you specify "float_left/right" for latex table. Default is "0pt" for automated determination but you may specify it manually.
Details

For LaTeX, if you use other than English environment

- all tables are converted to 'UTF-8'. If you use, for example, Hungarian characters on a Windows machine, make sure to use Sys.setlocale("LC_ALL","Hungarian") to avoid unexpected conversions.
- protect_latex = TRUE has no effect.

For HTML,

- protect_latex = TRUE is for including complicated math in HTML output. The LaTeX may not include dollar signs even if they are escaped. Pandoc’s rules for recognizing embedded LaTeX are used.

Examples

```r
## Not run:
x_html <- knitr::kable(head(mtcars), "html")
kable_styling(x_html, "striped", position = "left", font_size = 7)

x_latex <- knitr::kable(head(mtcars), "latex")
kable_styling(x_latex, latex_options = "striped", position = "float_left")
## End(Not run)
```

kbl

Wrapper function of knitr::kable

Description

knitr’s kable function is the foundation of this package. However, it has many latex/html specific arguments hidden under the ground unless you check its source code. This wrapper function is created to provide better documentation (and auto-complete yay) and at the same time, solve the auto format setting in a better way.

Usage

```r
kbl(
  x,
  format,
  digits =getOption("digits"),
  row.names = NA,
  col.names = NA,
  align,
  caption = NULL,
  label = NULL,
  format.args = list(),
)```
escape = TRUE,
table.attr = "",
booktabs = FALSE,
longtable = FALSE,
valign = "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 For kable(), x is an R object, which is typically a matrix or data frame. For kables(), a list with each element being a returned value from kable().

format A character string. Possible values are latex, html, pipe (Pandoc’s pipe tables), simple (Pandoc’s simple tables), and rst. The value of this argument will be automatically determined if the function is called within a knitr document. The format value can also be set in the global option knitr.table.format. If format is a function, it must return a character string.

digits Maximum number of digits for numeric columns, passed to round(). This can also be a vector of length ncol(x), to set the number of digits for individual columns.

row.names Logical: whether to include row names. By default, row names are included ifrownames(x) is neither NULL nor identical to 1:nrow(x).

col.names A character vector of column names to be used in the table.

align Column alignment: a character vector consisting of 'l' (left), 'c' (center) and/or 'r' (right). By default or if align = NULL, numeric columns are right-aligned, and other columns are left-aligned. If length(align) == 1L, the string will be expanded to a vector of individual letters, e.g. 'clc' becomes c('c', 'l', 'c'). unless the output format is LaTeX.

caption The table caption.

label The table reference label. By default, the label is obtained from knitr::opts_current$get('label').

format.args A list of arguments to be passed to format() to format table values, e.g. list(big.mark = ',').

escape Boolean; whether to escape special characters when producing HTML or LaTeX tables. When escape = FALSE, you have to make sure that special characters will not trigger syntax errors in LaTeX or HTML.
landscape

A character string for addition HTML table attributes. This is convenient if you simply want to add a few HTML classes or styles. For example, you can put 'class=“table” style=“color: red”'.

booktabs

T/F for whether to enable the booktabs format for tables. I personally would recommend you turn this on for every latex table except some special cases.

longtable

T/F for whether to use the longtable format. If you have a table that will span over two or more pages, you will have to turn this on.

valign

You probably won’t need to adjust this latex option very often. If you are familiar with latex tables, this is the optional position for the tabular environment controlling the vertical position of the table relative to the baseline of the surrounding text. Possible choices are b, c and t (default).

position

This is the "real" or say floating position for the latex table environment. The kable only puts tables in a table environment when a caption is provided. That is also the reason why your tables will be floating around if you specify captions for your table. Possible choices are h (here), t (top, default), b (bottom) and p (on a dedicated page).

centering

T (default)/F. Whether to center tables in the table environment.

vline

vertical separator. Default is nothing for booktabs tables but "|" for normal tables.

toprule

toprule. Default is hline for normal table but toprule for booktabs tables.

bottomrule

bottomrule. Default is hline for normal table but bottomrule for booktabs tables.

midrule

midrule. Default is hline for normal table but midrule for booktabs tables.

linesep

By default, in booktabs tables, kable insert an extra space every five rows for clear display. If you don’t want this feature or if you want to do it in a different pattern, you can consider change this option. The default is c("", ",", ",", ", addlinespace"). Also, if you are not using booktabs, but you want a cleaner display, you can change this to ".

caption.short

Another latex feature. Short captions for tables

table.envir

You probably don’t need to change this as well. The default setting is to put a table environment outside of tabular if a caption is provided.

... Other arguments (see Examples).

---

**landscape**

Print the table on an isolated landscape page in PDF

**Description**

This function will put the table on an single landscape page. It’s useful for wide tables that can’t be printed on a portrait page.

**Usage**

landscape(kable_input, margin = NULL)
Arguments

kable_input: Output of knitr::kable() with format specified.
margin: Customizable page margin for special needs. Values can be "1cm", "1in" or similar.

Examples

```r
## Not run:
landscape(knitr::kable(head(mtcars), "latex"))
## End(Not run)
```

---

linebreak: Make linebreak in LaTeX Table cells

Description

This function generates LaTeX code of `makecell` so that users can have linebreaks in their table.

Usage

```
linebreak(x, align = c("l", "c", "r"), double_escape = F, linebreaker = "\\n")
```

Arguments

- `x`: A character vector.
- `align`: Choose from "l", "c" or "r".
- `double_escape`: Whether special characters should be double escaped. Default is FALSE.
- `linebreaker`: Symbol for linebreaks to replace. Default is `\\n`.

---

listify_args: Convert arguments for a single call into Map-able args

Description

Convert arguments for a single call into Map-able args.

Usage

```
listify_args(
  ..., 
  lengths = NA, 
  passthru = c("x", "y"), 
  notlen1vec = c("lim", "xlim", "ylim"), 
  notlen1lst = c("minmax", "min", "max"), 
  ignore = c("same_lim")
)
```
**Arguments**

- ... Arbitrary arguments to be possibly converted into lists of arguments.
- lengths Allowable lengths of the arguments, typically 1 and the length of the main variable (e.g., "x"). If 'NA' (default), it is not enforced.
- passthru Character vector of variables to pass through with no conversion to lists of values. Extra names (not provided in ... ) are ignored.
- notlen1vec Character vector of variables that are known to be length over 1 for a single plot call, so it will always be list-ified and extra care to ensure it is grouped correctly. Extra names (not provided in ... ) are ignored.
- notlen1lst Character vector of variables that are lists, so the inner list length is not checked/enforced. (For example, if a single plot call takes an argument list(a=1,b=2,d=3) and the multi-data call creates three plots, then a naïve match might think that the first plot would get list(a=1), second plot gets list(b=2), etc. Adding that list-argument to this 'notlen1lst' will ensure that the full list is passed correctly.) Extra names (not provided in ... ) are ignored.
- ignore Character vector of variables to ignore, never returned. (Generally one can control this by not adding the variable in the first place, but having this here allows some sanity checks and/or programmatic usage.)

**Value**

- list, generally a list of embedded lists

---

**Description**

Mirror mirror tell me, how does this kable look like?

**Usage**

```
magic_mirror(kable_input)
```

**Arguments**

- kable_input The output of kable

**Examples**

```
magic_mirror(knitr::kable(head(mtcars), "html"))
```
**make_inline_plot**

*Combine file (or svg text) and parameters into a 'kableExtraInlinePlots' object*

**Description**

Combine file (or svg text) and parameters into a 'kableExtraInlinePlots' object

**Usage**

```r
make_inline_plot(filename, file_ext, dev, width, height, res, del = TRUE)
```

**Arguments**

- `filename`: Passed through to the graphics device.
- `file_ext`: Character, something like "png".
- `dev`: Character (e.g., "svg", "pdf") or function (e.g.,
- `width`, `height`: Plot dimensions in pixels.
- `res`: The resolution of the plot; default is 300.
- `del`: If the file is svg-like, then the default action is to read the file into an embedded SVG object; once done, the file is no longer used. The default action is to delete this file early, set this to 'FALSE' to keep the file.

**Value**

list object, with class 'kableExtraInlinePlots'

---

**remove_column**

*Remove columns*

**Description**

Remove columns

**Usage**

```r
remove_column(kable_input, columns)
```

**Arguments**

- `kable_input`: Output of `knitr::kable()` with format specified
- `columns`: A numeric value or vector indicating in which column(s) rows need to be removed
Examples

```r
## Not run:
remove_column(kable(mtcars), 1)

## End(Not run)
```

### Description

If the export format of the Rmarkdown document exist,

### Usage

```r
rmd_format()
```

### row_spec

Specify the look of the selected row

#### Description

This function allows users to select a row and then specify its look. It can also specify the format of the header row when `row = 0`.

#### Usage

```r
row_spec(
  kable_input,
  row,
  bold = FALSE,
  italic = FALSE,
  monospace = FALSE,
  underline = FALSE,
  strikeout = FALSE,
  color = NULL,
  background = NULL,
  align = NULL,
  font_size = NULL,
  angle = NULL,
  extra_css = NULL,
  hline_after = FALSE,
  extra_latex_after = NULL
)
```
Arguments

kable_input  Output of knitr::kable() with format specified
row A numeric value or vector indicating which row(s) to be selected. You don’t need to count in header rows or group labeling rows.
bold A T/F value to control whether the text of the selected row need to be bolded.
italic A T/F value to control whether the text of the selected row need to be emphasized.
monospace A T/F value to control whether the text of the selected row need to be monospaced (verbatim)
underline A T/F value to control whether the text of the selected row need to be underlined
strikeout A T/F value to control whether the text of the selected row need to be stricked out.
color A character string for row text color. For example, "red" or "#BBBBBB".
background A character string for row background color. Here please pay attention to the differences in color codes between HTML and LaTeX.
align A character string for cell alignment. For HTML, possible values could be l, c, r plus left, center, right, justify, initial and inherit while for LaTeX, you can only choose from l, c & r.
font_size A numeric input for font size. For HTML, you can also use options including xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, initial and inherit.
angle 0-360, degree that the text will rotate.
extra_css Extra css text to be passed into the cells of the row. Note that it’s not for the whole row.
hline_after T/F. A replicate of hline.after in xtable. It addes a hline after ther row
extra_latex_after Extra LaTeX text to be added after the row. Similar with add.to.row in xtable

Examples

```r
## Not run:
x <- knitr::kable(head(mtcars), "html")
row_spec(x, 1:2, bold = TRUE, italic = TRUE)
## End(Not run)
```
save_kable

Description

Save kable to files

Usage

```r
save_kable(x, file, bs_theme = "simplex", self_contained = TRUE, extra_dependencies = NULL, ...,
  latex_header_includes = NULL, keep_tex = FALSE, density = 300)
```

Arguments

- **x**: A piece of HTML code for tables, usually generated by kable and kableExtra
- **file**: save to files. If the input table is in HTML and the output file ends with .png, .pdf and .jpeg, webshot will be used to do the conversion.
- **bs_theme**: Which Bootstrap theme to use
- **self_contained**: Will the files be self-contained?
- **extra_dependencies**: Additional HTML dependencies. For example, list(
- **...**: Additional variables being passed to webshot::webshot. This is for HTML only.
- **latex_header_includes**: A character vector of extra LaTeX header stuff. Each element is a row. You can have things like c("\\usepackage{threeparttable}", "\\usepackage{icons}")
  You could probably add your language package here if you use non-English text in your table, such as \usepackage[ magyar]{babel}.
- **keep_tex**: A T/F option to control if the latex file that is initially created should be kept. Default is FALSE.
- **density**: density argument passed to magick if needed. Default is 300.
Examples

```r
## Not run:
library(kableExtra)

kable(mtcars[1:5, ], "html") %>%
kable_styling("striped") %>%
row_spec(1, color = "red") %>%
save_kable("inst/test.pdf")

## End(Not run)
```

---

**scroll_box**

*Put a HTML table into a scrollable box*

**Description**

This function will put a HTML kable object in a fixed-height, fixed-width or both box and make it scrollable.

**Usage**

```r
scroll_box(
  kable_input,
  height = NULL,
  width = NULL,
  box_css = "border: 1px solid #ddd; padding: 5px; ",
  extra_css = NULL,
  fixed_thead = TRUE
)
```

**Arguments**

- `kable_input`: A HTML kable object
- `height`: A character string indicating the height of the box, e.g. "50px"
- `width`: A character string indicating the width of the box, e.g. "100px"
- `box_css`: CSS text for the box
- `extra_css`: Extra CSS styles
- `fixed_thead`: HTML table option so table header row is fixed at top. Values can be either T/F or list(enabled = T/F, background = "anycolor").

**Examples**

```r
## Not run:
# Specify table size by pixels
kable(cbind(mtcars, mtcars), "html") %>%
kable_styling() %>%
```
**spec_angle**

Generate rotation angle for continuous values

**Description**

Generate rotation angle for continuous values

**Usage**

```r
spec_angle(x, begin, end, scale_from = NULL)
```

**Arguments**

- `x` continuous vectors of values
- `begin` Smallest degree to rotate. Default is 0
- `end` Largest degree to rotate. Default is 359.
- `scale_from` input range (vector of length two). If not given, is calculated from the range of `x`

---

**spec_boxplot**

Helper functions to generate inline sparklines

**Description**

These functions helps you quickly generate sets of sparkline style plots using base R plotting system. Currently, we support histogram, boxplot, line, scatter and pointrange plots. You can use them together with `column_spec` to generate inline plot in tables. By default, this function will save images in a folder called "kableExtra" and return the address of the file.
spec_boxplot

Usage

```r
spec_boxplot(
  x,
  width = 200,
  height = 50,
  res = 300,
  add_label = FALSE,
  label_digits = 2,
  same_lim = TRUE,
  lim = NULL,
  xaxt = "n",
  yaxt = "n",
  ann = FALSE,
  col = "lightgray",
  border = NULL,
  boxlty = 0,
  medcol = "red",
  medlwd = 1,
  dir = if (is_latex()) rmd_files_dir() else tempdir(),
  file = NULL,
  file_type = if (is_latex()) "pdf" else svglite::svglite,
  ...
)
```

Arguments

- **x**: Vector of values or List of vectors of values.
- **width**: The width of the plot in pixel
- **height**: The height of the plot in pixel
- **res**: The resolution of the plot. Default is 300.
- **add_label**: For boxplot. T/F to add labels for min, mean and max.
- **label_digits**: If T for add_label, rounding digits for the label. Default is 2.
- **same_lim**: T/F. If x is a list of vectors, should all the plots be plotted in the same range? Default is True.
- **lim**: Manually specify plotting range in the form of c(0, 10).
- **xaxt**: On/Off for xaxis text
- **yaxt**: On/Off for yaxis text
- **ann**: On/Off for annotations (titles and axis titles)
- **col**: Color for the fill of the histogram bar/boxplot box.
- **border**: Color for the border.
- **boxlty**: Boxplot - box boarder type
- **medcol**: Boxplot - median line color
- **medlwd**: Boxplot - median line width
spec_color

Directory of where the images will be saved.

File name. If not provided, a random name will be used

Graphic device. Can be character (e.g., "pdf") or a graphics device function (grDevices::pdf). This defaults to "pdf" if the rendering is in LaTeX and "svg" otherwise.

extraparameters passing to boxplot

---

spec_color

Generate viridis Color code for continuous values

Description

Generate viridis Color code for continuous values

Usage

spec_color(
  x,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  na_color = "#BBBBBB",
  scale_from = NULL
)

Arguments

x
  continuous vectors of values

alpha
  The alpha transparency, a number in [0,1], see argument alpha in hsv.

begin
  The (corrected) hue in [0,1] at which the viridis colormap begins.

direction
  The (corrected) hue in [0,1] at which the viridis colormap ends.

direction
  Sets the order of colors in the scale. If 1, the default, colors are ordered from darkest to lightest. If -1, the order of colors is reversed.

option
  A character string indicating the colormap option to use. Four options are available: "magma" (or "A"), "inferno" (or "B"), "plasma" (or "C"), "viridis" (or "D", the default option) and "cividis" (or "E").

na_color
  color code for NA values

scale_from
  input range (vector of length two). If not given, is calculated from the range of x
spec_font_size  Generate common font size for continuous values

Description

Generate common font size for continuous values

Usage

```r
spec_font_size(x, begin = 8, end = 16, na_font_size = 12, scale_from = NULL)
```

Arguments

- `x`: continuous vectors of values
- `begin`: Smallest font size to be used. Default is 10.
- `end`: Largest font size. Default is 20.
- `na_font_size`: font size for NA values
- `scale_from`: input range (vector of length two). If not given, is calculated from the range of `x`

spec_hist  Helper functions to generate inline sparklines

Description

These functions help you quickly generate sets of sparkline style plots using base R plotting system. Currently, we support histogram, boxplot, line, scatter and pointrange plots. You can use them together with `column_spec` to generate inline plot in tables. By default, this function will save images in a folder called "kableExtra" and return the address of the file.

Usage

```r
spec_hist(  
  x,  
  width = 200,  
  height = 50,  
  res = 300,  
  breaks = "Sturges",  
  same_lim = TRUE,  
  lim = NULL,  
  xaxt = "n",  
  yaxt = "n",  
  ann = FALSE,  
  col = "lightgray",  
)
**spec_image**

```r
tabular = NULL,
  dir = if (is_latex()) rmd_files_dir() else tempdir(),
  file = NULL,
  file_type = if (is_latex()) "pdf" else svglite::svglite,
  ...
)
```

**Arguments**

- **x** Vector of values or List of vectors of values.
- **width** The width of the plot in pixel
- **height** The height of the plot in pixel
- **res** The resolution of the plot. Default is 300.
- **breaks** The break option in `hist`. Default is "Sturges" but you can also provide a vector to manually specify break points.
- **same_lim** T/F. If `x` is a list of vectors, should all the plots be plotted in the same range? Default is True.
- **lim** Manually specify plotting range in the form of `c(0, 10)`. 
- **xaxt** On/Off for xaxis text
- **yaxt** On/Off for yaxis text
- **ann** On/Off for annotations (titles and axis titles)
- **col** Color for the fill of the histogram bar/boxplot box.
- **border** Color for the border.
- **dir** Directory of where the images will be saved.
- **file** File name. If not provided, a random name will be used
- **file_type** Graphic device. Can be character (e.g., "pdf") or a graphics device function (`grDevices::pdf`). This defaults to "pdf" if the rendering is in LaTeX and "svg" otherwise. for HTML output
- **...** extra parameters sending to `hist()`

---

**Description**

Users can directly provide image file path to column `spec`. However, if you need to specify the size of the image, you will need this function.

**Usage**

```r
spec_image(path, width, height, res = 300, svg_text = NULL)
```
Arguments

- **path**: file path(s)
- **width**: image width in pixel
- **height**: image height in pixel
- **res**: image resolution.
- **svg_text**: If you have the raw text for SVG. Put them here

**Description**

These functions help you quickly generate sets of sparkline style plots using base R plotting system. Currently, we support histogram, boxplot, line, scatter and pointrange plots. You can use them together with `column_spec` to generate inline plot in tables. By default, this function will save images in a folder called "kableExtra" and return the address of the file.

**Usage**

```r
spec_plot(
  x,
  y = NULL,
  width = 200,
  height = 50,
  res = 300,
  same_lim = TRUE,
  xlim = NULL,
  ylim = NULL,
  xaxt = "n",
  yaxt = "n",
  ann = FALSE,
  col = "lightgray",
  border = NULL,
  frame.plot = FALSE,
  lwd = 2,
  pch = ".",
  cex = 2,
  type = "l",
  polymin = NA,
  minmax = list(pch = ".", cex = cex, col = "red"),
  min = minmax,
  max = minmax,
  dir = if (is_latex()) rmd_files_dir() else tempdir(),
  file = NULL,
  file_type = if (is_latex()) "pdf" else svglite::svglite,
  ...
)
```
Arguments

- **x, y**: Vector of values or List of vectors of values. y is optional.
- **width**: The width of the plot in pixel
- **height**: The height of the plot in pixel
- **res**: The resolution of the plot. Default is 300.
- **same_lim**: T/F. If x is a list of vectors, should all the plots be plotted in the same range? Default is True.
- **xlim, ylim**: Manually specify plotting range in the form of c(0, 10).
- **xaxt, yaxt**: On/Off for xaxis text
- **ann**: On/Off for annotations (titles and axis titles)
- **col**: Color for the fill of the histogram bar/boxplot box.
- **border**: Color for the border.
- **frame.plot**: On/Off for surrounding box (spec_plot only). Default is False.
- **lwd**: Line width for spec_plot; within spec_plot, the minmax argument defaults to use this value for cex for points. Default is 2.
- **pch, cex**: Shape and size for points (if type is other than "l").
- **type**: Passed to plot, often one of "l", "p", or "b", see graphics::plot.default() for more details. Ignored when 'polymin' is not 'NA'.
- **polymin**: Special argument that converts a "line" to a polygon, where the flat portion is this value, and the other side of the polygon is the 'y' value ('x' if no 'y' provided). If 'NA' (the default), then this is ignored; otherwise if this is numeric then a polygon is created (and 'type' is ignored). Note that if 'polymin' is in the middle of the 'y' values, it will generate up/down polygons around this value.
- **minmax, min, max**: Arguments passed to points to highlight minimum and maximum values in spec_plot. If min or max are NULL, they default to the value of minmax. Set to an empty list() to disable.
- **dir**: Directory of where the images will be saved.
- **file**: File name. If not provided, a random name will be used
- **file_type**: Graphic device. Can be character (e.g., "pdf") or a graphics device function (grDevices::pdf). This defaults to "pdf" if the rendering is in LaTeX and "svg" otherwise.
- **...**: extra parameters passing to plot
spec_pointrange

Helper functions to generate inline sparklines

Description

These functions help you quickly generate sets of sparkline style plots using base R plotting system. Currently, we support histogram, boxplot, line, scatter and pointrange plots. You can use them together with column_spec to generate inline plot in tables. By default, this function will save images in a folder called "kableExtra" and return the address of the file.

Usage

spec_pointrange(
  x,
  xmin,
  xmax,
  vline = NULL,
  width = 200,
  height = 50,
  res = 300,
  same_lim = TRUE,
  lim = NULL,
  xaxt = "n",
  yaxt = "n",
  ann = FALSE,
  col = "red",
  cex = 0.3,
  frame.plot = FALSE,
  dir = if (is_latex()) rmd_files_dir() else tempdir(),
  file = NULL,
  file_type = if (is_latex()) "pdf" else svglite::svglite,
  ...
)

Arguments

x, xmin, xmax  A scalar value or List of scalar values for dot, left and right errorbar.
vline  A scalar value for where to draw a vertical line.
width  The width of the plot in pixel
height  The height of the plot in pixel
res  The resolution of the plot. Default is 300.
same_lim  T/F. If x is a list of vectors, should all the plots be plotted in the same range? Default is True.
lim  Manually specify plotting range in the form of c(0, 10).
xaxt  On/Off for xaxis text
**spec_popover**

<table>
<thead>
<tr>
<th>Content</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>yaxt</code></td>
<td>On/Off for yaxis text</td>
</tr>
<tr>
<td><code>ann</code></td>
<td>On/Off for annotations (titles and axis titles)</td>
</tr>
<tr>
<td><code>col</code></td>
<td>Color for the fill of the histogram bar/boxplot box.</td>
</tr>
<tr>
<td><code>cex</code></td>
<td>Size of the mean dot and error bar size.</td>
</tr>
<tr>
<td><code>frame.plot</code></td>
<td>T/F for whether to plot the plot frames.</td>
</tr>
<tr>
<td><code>dir</code></td>
<td>Directory of where the images will be saved.</td>
</tr>
<tr>
<td><code>file</code></td>
<td>File name. If not provided, a random name will be used</td>
</tr>
<tr>
<td><code>file_type</code></td>
<td>Graphic device. Can be character (e.g., &quot;pdf&quot;) or a graphics device function (grDevices::pdf). This defaults to &quot;pdf&quot; if the rendering is in LaTeX and &quot;svg&quot; otherwise. for HTML output</td>
</tr>
<tr>
<td>...</td>
<td>Extra parameters sending to hist()</td>
</tr>
</tbody>
</table>

---

**Description**

Setup bootstrap popover

**Usage**

```r
spec_popover(
  content = NULL,
  title = NULL,
  trigger = "hover",
  position = "right"
)
```

**Arguments**

- `content`: Content for pop-over message
- `title`: Title for pop-over message.
- `trigger`: Controls how the pop-over message should be triggered. Possible values include hover (default), click, focus and manual.
- `position`: How the tooltip should be positioned. Possible values are right (default), top, bottom, left & auto.
spec_tooltip

Setup bootstrap tooltip

Description

Setup bootstrap tooltip

Usage

spec_tooltip(title, position = "right")

Arguments

title text for hovering message
position How the tooltip should be positioned. Possible values are right (default), top, bottom, left & auto.

usepackage_latex

Load a LaTeX package

Description

Load a LaTeX package using R code. Just like \usepackage{} in LaTeX

Usage

usepackage_latex(name, options = NULL)

Arguments

name The LaTeX package name
options The LaTeX options for the package

Examples

usepackage_latex("xcolor")
xml_as_kable  

Convert XML back to kable

Description
Convert XML back to kable

Usage
xml_as_kable(x)

Arguments
x
XML table object

xtable2kable  

Convert xtable to a kable object

Description
This function allow users to turn an xtable object into a kable so they can use most of kableExtra’s functions with their xtable code without making too many changes. Note that although I tested many cases and it seems to work, this function may not be functional in some other cases. I’m not a regular xtable user and can only provide very limited support for this function.

You should use this table in the same way as print.xtable. All the options you provided to this function will be sent to print.xtable. Instead of printing out the result, this function will return the LaTeX or HTML as text and a kable object.

Usage
xtable2kable(x, ...)

Arguments
x
an xtable object
...
options for print.xtable

Examples
## Not run:
library(xtable)
xtable(mtcars) %>%
xtable2kable(booktabs = TRUE) %>%
kable_styling(latex_options = "striped")

## End(Not run)
Index

* **package**
  
  `kableExtra-package`, 3

  `add_footnote`, 4
  `add_header_above`, 5
  `add_indent`, 7
  `as_image`, 7
  `auto_index`, 8

  `cell_spec`, 9
  `collapse_rows`, 11
  `column_spec`, 12

  `dev_chr (graphics_helpers)`, 17

  `footnote`, 14
  `footnote_marker_alphabet`
    `(footnote_marker_number)`, 16
  `footnote_marker_number`, 16
  `footnote_marker_symbol`
    `(footnote_marker_number)`, 16
  `format`, 28

  `graphics::plot.default()`, 43
  `graphics_dev (graphics_helpers)`, 17
  `graphics_helpers`, 17
  `group_rows`, 18

  `header_separate`, 20
  `hsv`, 39
  `html_dependency_bsTable`, 21
  `html_dependency_kePrint`, 21
  `html_dependency_lightable`, 21

  `is_svg (graphics_helpers)`, 17

  `kable_as_image`, 22
  `kable_as_xml`, 23
  `kable_classic`, 23
  `kable_classic_2 (kable_classic)`, 23
  `kable_material (kable_classic)`, 23
  `kable_material_dark (kable_classic)`, 23
  `kable_minimal (kable_classic)`, 23
  `kable_paper (kable_classic)`, 23
  `kable_styling`, 24
  `kableExtra (kableExtra-package)`, 3
  `kableExtra-package`, 3
  `kableExtra_latex_packages`, 22
  `kbl`, 27
  `knitr::kable()`, 32

  `landscape`, 29
  `linebreak`, 30
  `listify_args`, 30

  `magic_mirror`, 31
  `make_inline_plot`, 32

  `opts_current`, 28

  `pack_rows (group_rows)`, 18

  `remove_column`, 32
  `rmd_format`, 33
  `row_spec`, 33

  `save_kable`, 35
  `scroll_box`, 36
  `spec_angle`, 37
  `spec_boxplot`, 37
  `spec_color`, 39
  `spec_font_size`, 40
  `spec_hist`, 40
  `spec_image`, 41
  `spec_plot`, 42
  `spec_pointrange`, 44
  `spec_popover`, 45
  `spec_tooltip`, 46

  `text_spec (cell_spec)`, 9
  `usepackage_latex`, 46
xml_as_kable, 47
xtable2kable, 47