Package 'apa7'

September 10, 2025

Title Facilitate Writing Documents in American Psychological Association Style, Seventh Edition

Version 0.1.0

Description Create American Psychological Association Style, Seventh Edition documents. Format numbers and text consistent with APA style. Create tables that comply with APA style by extending flextable functions.

License CC0

URL https://github.com/wjschne/apa7, https://wjschne.github.io/apa7/

BugReports https://github.com/wjschne/apa7/issues

Depends R (>= 4.1.0)

Imports dplyr, effectsize, flextable, ftExtra, glue, parameters, performance, psych, purrr, rlang, S7, scales, shiny, signs, stringr, tibble, tidyr

Suggests bsicons, bslib, cli, conflicted, forcats, ggplot2, knitr, quarto, R.utils, rclipboard, readr, rmarkdown, shinyWidgets, snakecase, spelling, testthat (>= 3.0.0), tidyselect, tippy, toastui, yaml

VignetteBuilder knitr, quarto

Config/Needs/website quarto

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

NeedsCompilation no

Author W. Joel Schneider [aut, cre] (ORCID: <https://orcid.org/0000-0002-8393-5316>)

Maintainer W. Joel Schneider <w.joel.schneider@gmail.com>

Repository CRAN

Date/Publication 2025-09-10 08:30:07 UTC

Contents

`add_break_columns` *Add break columns*

Description

Add break columns

Usage

```
add_break_columns(
  d,
  ...,
  .before = FALSE,
  omit_first = FALSE,
  omit_last = FALSE
)
```

Arguments

d	data.frame or tibble
...	Column name or tidyselect function. Select columns
.before	insert break columns before selected columns (defaults to FALSE)
omit_first	omit the first break column
omit_last	omit the last break column

Value

data.frame or tibble

Examples

```
d <- data.frame(x_n = 3, x_mean = 4,
            y_n = 5, y_mean = 6,
            z_n = 4, z_mean = 4)
# Unquoted variable names
add_break_columns(d, x_mean)

# Character vector
add_break_columns(d, c("y_n", "z_n"),  .before = TRUE)

# Tidyselect function (contains, starts_with, ends_with,
# matches, num_range, all_of, any_of)
# Insert columns after all columns
# ending with "_mean" except the last instance
add_break_columns(d,
                  dplyr::ends_with("_mean"),
                  omit_last = TRUE)
```

---

add_list_column *Make a column into a list column*

Description

Make a column into a list column

Usage

```
add_list_column(data, ..., type = c("1", "a", "A", "I", "i"), sep = ". ")
```

Arguments

data	data.frame or tibble
...	Column name or tidyselect function. Select columns. Default is first column
type	list type. Can be "1" (numeric), "a" (lowercase alphabetical), or "ABC" (uppercase alphabetical), "i" (lowercase Roman numerals), "I" (uppercase Roman numerals)
sep	separator

Value

data.frame

Examples

```
d <- data.frame(x = letters[1:5], y = letters[2:6])
# default is first column
add_list_column(d)
# select any column
add_list_column(d, y)
add_list_column(d, type = "a", sep = ") ") |>
 apa_flextable()
```

---

add_star_column *Adds stars next to a column based on p-values*

Description

Adds stars next to a column based on p-values

Usage

```
add_star_column(
  data,
  ...,
  p = "p",
  merge = FALSE,
  superscript = TRUE,
  star = "\\*",
  alpha = c(0.05, 0.01, 0.001),
  first_alpha_marginal = FALSE,
  add_trailing_space = FALSE,
  prefix = "\\"
)
```

Arguments

data	data.frame or tibble
...	Column name or tidyselect function. Select columns
p	Column name or tidyselect function. Select p-value column name
merge	merge and balance columns (default: FALSE)
superscript	make as superscript
star	text for making stars
alpha	vector of thresholds
first_alpha_marginal	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star
add_trailing_space	if TRUE, adds a trailing space after the stars (default: FALSE)
prefix	usually backslashes to prevent markdown from interpreting asterisks as bullets or italics

Value

data.frame

Examples

```
data.frame(b = c(1.4,2.2),
           p = c(.54, .02)) |>
 add_star_column(b, p)
```

---

align_chr *Align text on center text (default is decimal)*

Description

Align text on center text (default is decimal)

Usage

```
align_chr(
  x,
  accuracy = NULL,
  trim_leading_zeros = FALSE,
  drop0trailing = FALSE,
  add_plusses = FALSE,
  padding_character = NULL,
  center = ".",
  format_integers = FALSE,
  side = c("both", "left", "right"),
```

```
  NA_value = "",
  format_numeric_character = FALSE,
  ...
)
```

Arguments

x	vector (numeric or character)
accuracy	number to round to. If NULL, the current default accuracy set with apa7_defaults() will be used.
trim_leading_zeros	if TRUE (default), trims leading zeros, otherwise keeps them
drop0trailing	Drop trailing zeros
add_plusses	if TRUE (default), adds a plus to positive numbers
padding_character	character to use for padding, default is (figure space)
center	text on which to align text. Center on decimal by default, but can be any text.
format_integers	If TRUE, integers will be formatted with digits
side	side on which to make text of equal width
NA_value	value to replace NA
format_numeric_character	format character variables with numeric content
...	additional arguments passed to signs::signs()

Value

character vector

Examples

```
align_chr(c(1, 10, 100))
```

---

apa7_defaults *Set defaults for apa7 package*

Description

Set defaults for apa7 package

Usage

```
apa7_defaults(
  accuracy = NULL,
  font_family = NULL,
  intercept_text = NULL,
  column_formats = NULL,
  number_formatter = NULL,
  trim_leading_zero = NULL,
  reset = FALSE
)
```

Arguments

accuracy	numeric (default: .01)
font_family	font family
intercept_text	what to call the intercept
column_formats	column formatting functions
number_formatter	default function to format numbers
trim_leading_zero	default function to trim leading zeros from numbers
reset	if TRUE, reset all defaults (except as specified)

Value

previous defaults

Examples

```
apa7_defaults(accuracy = .001)
# Reset to package defaults
apa7_defaults(reset = TRUE)
```

---

apa_chisq *Make contingency table with chi-square test of independence*

Description

Make contingency table with chi-square test of independence

Usage

```
apa_chisq(
  data,
  note = NULL,
  row_title_column = NULL,
  row_title_prefix = "",
  row_title_sep = " ",
  row_title_align = "center",
  row_title_border = list(color = "gray20", style = "solid", width = 1),
  left_column_padding = 20,
  cwidth = 0.75,
  cheight = 0.25,
  separate_headers = TRUE,
  apa_style = TRUE,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
  layout = "autofit",
  table_width = 1,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  auto_format_columns = TRUE,
  column_formats = NULL,
  pretty_widths = TRUE,
  suppress_warnings = TRUE,
  ...
)
```

Arguments

data	A two-column data.frame or tibble
note	Custom note (overrides automatic note.)
row_title_column	Column name or tidyselect function. column to group rows
row_title_prefix	text to be added to each title
row_title_sep	separator for prefix
row_title_align	alignment of row title ("left", "center", "right")
row_title_border	list of flextable styles

left_column_padding
: Number of points the left column is padded (only relevant when there is a row_title_column and row_title_align = "left")

cwidth
: initial cell width in inches

cheight
: initial cell height in inches

separate_headers
: separate header rows (default: TRUE)

apa_style
: apply apa_style function (default: TRUE)

font_family
: font family

font_size
: font size

text_color
: text color

border_color
: border color

border_width
: border width in pixels

line_spacing
: spacing between lines

horizontal_padding
: horizontal padding (in pixels)

table_align
: table alignment ("left", "center", "right")

layout
: table layout ("autofit", "fixed")

table_width
: table width (in pixels, 0 for auto)

markdown
: apply markdown formatting to header and body

markdown_header
: apply markdown formatting to header

markdown_body
: apply markdown formatting to body

auto_format_columns
: if true, will attempt to format some columns automatically

column_formats
: a column_formats object

pretty_widths
: apply pretty_widths function

suppress_warnings
: Suppress any warnings if true.

...
: arguments passed to apa_style

Value

flextable::flextable

Examples

```
apa_chisq(mtcars[, c("am", "gear")])
```

apa_cor *APA-formatted correlation table*

Description

APA-formatted correlation table

Usage

```
apa_cor(
  data,
  note = NULL,
  p_value = c(0.05, 0.01, 0.001),
  digits = 2,
  bold_significant = FALSE,
  star_significant = TRUE,
  significance_note = TRUE,
  output = c("flextable", "tibble"),
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  table_width = 6.5,
  keep_empty_star_columns = TRUE,
  summary_functions = list(M = mean, SD = stats::sd),
  column_formats = NULL,
  ...
)
```

Arguments

data	data.frame or tibble with variables to be
note	Custom note to appear below table. (Overrides automatic note.)
p_value	p-value needed to be flagged as significant
digits	Number of digits for rounding
bold_significant	bold significant correlations
star_significant	start significant correlations
significance_note	If TRUE, place note at bottom of table that significant correlations are bolded.
output	output type. Can be "flextable" or "tibble"
font_family	font family

font_size	font size
text_color	text color
border_color	border color
border_width	border width in pixels
line_spacing	spacing between lines
table_width	table width (in pixels, 0 for auto)
keep_empty_star_columns	Keep remove empty star columns (Default: TRUE)
summary_functions	A named list of functions that summarize data columns (e.g., mean, sd)
column_formats	column_formats object
...	<data-masking> parameters passed to psych::corTest

Value

flextable::flextable

Examples

```
apa_cor(mtcars[, c("mpg", "am", "gear", "carb")], output = "flextable")
apa_cor(mtcars[, c("mpg", "am", "gear", "carb")], output = "tibble")
```

---

apa_flextable *Convert data to flextable consistent with APA style*

Description

The apa_flextable function performs a number of formatting operations on the data before and after the data are sent to flextable. See Details.

Usage

```
apa_flextable(
  data,
  row_title_column = NULL,
  row_title_align = "left",
  row_title_prefix = "",
  row_title_sep = " ",
  row_title_border = list(color = "gray20", style = "solid", width = 1),
  left_column_padding = 20,
  col_keys = colnames(data),
  cwidth = 0.75,
  cheight = 0.25,
  header_align_vertical = c("top", "middle", "bottom"),
  separate_headers = TRUE,
```

```
  apa_style = TRUE,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
  layout = "autofit",
  table_width = 1,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  no_markdown_columns = NULL,
  no_markdown_columns_header = NULL,
  no_format_columns = NULL,
  auto_format_columns = TRUE,
  column_formats = NULL,
  pretty_widths = TRUE,
  add_breaks_between_spanners = TRUE,
  ...
)
```

Arguments

`data`	data.frame or tibble
`row_title_column`	Column name or tidyselect function. column to group rows
`row_title_align`	alignment of row title ("left", "center", "right")
`row_title_prefix`	text to be added to each title
`row_title_sep`	separator for prefix
`row_title_border`	list of flextable styles
`left_column_padding`	Number of points the left column is padded (only relevant when there is a `row_title_column` and `row_title_align = "left"`)
`col_keys`	column keys passed to flextable (defaults data column names)
`cwidth`	initial cell width in inches
`cheight`	initial cell height in inches
`header_align_vertical`	vertical alignment of headers. Can be "top", "middle", or "bottom"
`separate_headers`	separate header rows (default: `TRUE`)

apa_style	apply apa_style function (default: TRUE)
font_family	font family
font_size	font size
text_color	text color
border_color	border color
border_width	border width in pixels
line_spacing	spacing between lines
horizontal_padding	horizontal padding (in pixels)
table_align	table alignment ("left", "center", "right")
layout	table layout ("autofit", "fixed")
table_width	table width (in pixels, 0 for auto)
markdown	apply markdown formatting to header and body
markdown_header	apply markdown formatting to header
markdown_body	apply markdown formatting to body
no_markdown_columns	body columns that should not be treated as markdown
no_markdown_columns_header	column headers that should not be treated as markdown
no_format_columns	Column name or tidyselect function. selected columns are not formatted
auto_format_columns	if true, will attempt to format some columns automatically
column_formats	a column_formats object
pretty_widths	apply pretty_widths function
add_breaks_between_spanners	add breaks between spanners if TRUE
...	arguments passed to apa_style

Details

Roughly speaking, apa_flextable performs these operations by default:

1. Apply as_grouped_data and restructure row titles, if row_title is specified.
2. Format data with apa_format_columns if auto_format_columns = TRUE
3. Separate headers into multiple header rows if separate_headers = TRUE
4. Apply flextable::flextable
5. Apply flextable::surround to make borders to separate row groups, if any.
6. Apply the apa_style function (table formatting and markdown conversion) if apa_style = TRUE
7. Apply pretty_widths if pretty_widths = TRUE

Value

flextable::flextable

Examples

```
library(dplyr)
library(tidyr)
library(flextable)
mtcars %>%
  dplyr::select(vs, am, gear, carb) |>
  tidyr::pivot_longer(-vs,  names_to = "Variable") |>
  dplyr::summarise(Mean = round(mean(value), 2),
                   SD = round(sd(value), 2),
                   .by = c(Variable,vs)) |>
  dplyr::mutate(vs = factor(vs, levels = 0:1, labels = c("Automatic", "Manual"))) |>
  apa_flextable(row_title_column= vs,  row_title_align = "center") |>
  align(j = 2:3, align = "center")
```

---

apa_format_columns *Format data columns*

Description

Format data columns

Usage

```
apa_format_columns(
  data,
  column_formats = NULL,
  no_format_columns = NULL,
  rename_headers = TRUE,
  latex_headers = FALSE,
  format_separated_headers = TRUE,
  sep = "_",
  accuracy = NULL
)
```

Arguments

data	data set (data.frame or tibble)
column_formats	column_formats object. If NULL, the current default formatter set with apa7_defaults() will be used.
no_format_columns	Column name or tidyselect function. selected columns are not formatted
rename_headers	if TRUE, rename headers with markdown or latex
latex_headers	if TRUE, rename headers with latex instead of markdown

format_separated_headers
: if TRUE, format headers with separated names. For example, if the formatter formats column R2 as *R*^2^, then Model 1_R2 becomes Model 1_*R*^2^)

sep
: separator for separated headers (default is "_")

accuracy
: numeric (default: NULL, uses the current default accuracy set with apa7_defaults()). If not NULL, sets the accuracy for the formatter.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
  parameters::parameters() |>
  apa_format_columns() |>
  apa_flextable()
```

---

apa_loadings *print loadings*

Description

print loadings

Usage

```
apa_loadings(
  fit,
  sort_loading = TRUE,
  min_loading = 0.2,
  column_formats = NULL,
  complexity = FALSE,
  uniqueness = FALSE
)
```

Arguments

fit
: model fit object

sort_loading
: sort table using psych::fa.sort

min_loading
: minimum loading to display

column_formats
: column_formats object to format columns. If NULL, the default column_formats is used.

complexity
: print complexity column in factor analysis table

uniqueness
: print uniqueness column in factor analysis table

Value

tibble

---

apa_p *p-value in APA format*

Description

p-value in APA format

Usage

```
apa_p(
  p,
  inline = FALSE,
  markdown = TRUE,
  min_digits = 2,
  max_digits = 3,
  align = FALSE
)
```

Arguments

p	probability
inline	If TRUE (default), returns statistic (e.g.,e p = .04), otherwise just the number (e.g., .04)
markdown	By default, outputs text compatible with markdown if TRUE, otherwise prints plain text compatible with latex.
min_digits	minimum number of digits to round to. Default is 2.
max_digits	maximum number of digits to round to. Default is 3.
align	decimal alignment if TRUE

Value

character vector

Examples

```
# Values less than .001 are <.001
apa_p(.0002)
# Values between .001 and .01 are rounded to 3 digits
apa_p(.002)
# Values between .01 and .995 are rounded to 2 digits
apa_p(.02)#'
apa_p(.22)
apa_p(.994)
```

```
# Values above .995 are >.99
apa_p(.999)
# Rounding to 3 digits
apa_p(.2341, min_digits = 3)
apa_p(.0123, min_digits = 3)
apa_p(.00123, min_digits = 3)
apa_p(.000123, min_digits = 3)
apa_p(.991, min_digits = 3)
apa_p(.9991, min_digits = 3)
apa_p(.9995, min_digits = 3)
```

`apa_parameters` *format model parameters in APA style*

Description

format model parameters in APA style

Usage

```
apa_parameters(
  fit,
 predictor_parameters = c("Coefficient", "SE", "Std_Coefficient", "t", "df_error", "p"),
  starred = NULL,
  bolded = NULL,
  column_formats = NULL,
  t_with_df = TRUE
)

## S3 method for class 'lm'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Std_Coefficient", "t",
    "df_error", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE
)

## S3 method for class 'list'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Std_Coefficient", "t",
    "df_error", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
```

```
  t_with_df = TRUE
)
```

Arguments

fit	model fit object
predictor_parameters	predictor parameters to display. If named vector, column names will be vector names
starred	columns to star with significant p_values
bolded	columns to bold, if significant
column_formats	column_formats object to format columns. If NULL, the default column_formats is used.
t_with_df	if TRUE, the t column will be displayed with degrees of freedom in parentheses. If FALSE, only the t value is displayed.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
   apa_parameters() |>
   apa_flextable()
```

---

apa_performance *format model performance metrics in APA style*

Description

format model performance metrics in APA style

Usage

```
apa_performance(fit, metrics = c("R2", "Sigma"), column_formats = NULL)

## S3 method for class 'lm'
apa_performance(fit, metrics = c("R2", "Sigma"), column_formats = NULL)
```

Arguments

fit	model fit object
metrics	performance metrics. Default is R2 and Sigma
column_formats	column_formats object to format columns. If NULL, the default column_formats is used.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
   apa_performance() |>
   apa_flextable()
```

---

apa_performance_comparison

format model comparison metrics in APA style

Description

format model comparison metrics in APA style

Usage

```
apa_performance_comparison(
  ...,
  metrics = c("R2", "deltaR2", "F", "p"),
  starred = NA,
  column_formats = NULL
)
```

Arguments

`...`	model fit objects
`metrics`	performance metrics. Default is R2, deltaR2, F, and p
`starred`	columns to star with significant p_values
`column_formats`	column_formats object to format columns. If NULL, the default column_formats is used.

Value

tibble

Examples

```
m1 <- lm(mpg ~ cyl, data = mtcars)
m2 <- lm(mpg ~ cyl + wt, data = mtcars)
apa_performance_comparison(list(`Model 1` =m1, `Model 3` =m2)) |>
   apa_flextable()
```

apa_p_star_note *Make star notes for p-values*

Description

Make star notes for p-values

Usage

```
apa_p_star_note(x = c(0.05, 0.01, 0.001), first_alpha_marginal = FALSE)
```

Arguments

x	vector of alpha values (p-value thresholds)
first_alpha_marginal	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star

Value

character vector

Examples

```
apa_p_star_note()
apa_p_star_note(x = c(.10, .05, .01, .001), first_alpha_marginal = TRUE)
```

apa_style *Style* flextable::flextable *object according to APA style*

Description

Style flextable::flextable object according to APA style

Usage

```
apa_style(
  x,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
```

```
  header_align_vertical = c("top", "middle", "bottom"),
  layout = "autofit",
  table_width = 0,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  no_markdown_columns = NULL,
  no_markdown_columns_header = no_markdown_columns,
  separate_headers = TRUE
)
```

Arguments

x	object
font_family	font family
font_size	font size
text_color	text color
border_color	border color
border_width	border width in pixels
line_spacing	spacing between lines
horizontal_padding	horizontal padding (in pixels)
table_align	table alignment ("left", "center", "right")
header_align_vertical	vertical alignment of headers. Can be "top", "middle", or "bottom"
layout	table layout ("autofit", "fixed")
table_width	table width (in pixels, 0 for auto)
markdown	apply markdown formatting to header and body
markdown_header	apply markdown formatting to header
markdown_body	apply markdown formatting to body
no_markdown_columns	body columns that should not be treated as markdown
no_markdown_columns_header	column headers that should not be treated as markdown
separate_headers	separate headers into column spanner labels

Value

object

Examples

```
d <- data.frame(x = 1:3, y = 4:6)
flextable::flextable(d) |>
  apa_style()
```

column_format *Column format class*

Description

This class is used to define the format of columns in tables, including the name, header, latex representation, and a formatter function.

Usage

```
column_format(
  name = character(0),
  header = character(0),
  latex = character(0),
  formatter = function() NULL
)
```

Arguments

name	name of column
header	markdown representation of header name
latex	latex representation of header name
formatter	function that formats the column values. It should take a vector of values and return a character vector of formatted values.

Value

column_format object

Examples

```
R2 <- column_format(
         "R2",
         header = "*R*^2^",
         latex = "$R^2$",
         formatter = \(x, accuracy = the$accuracy, ...) {
                       align_chr(x,
                                 accuracy = accuracy,
                                 trim_leading_zeros = TRUE,
                                 ...)
                       })
R2
R2@header
R2@formatter
```

`column_formats` *Create a set of column formats*

Description

Returns an S7 object that contains a list of `column_format` objects that can be used to format parameters in APA style.

Usage

```
column_formats(
  .data = NULL,
  accuracy = NULL,
  intercept_text = NULL,
  starred_columns = character(0),
  variable_labels = character(0),
  custom_columns = NULL
)
```

Arguments

`.data`	list of `column_format` objects
`accuracy`	numeric (passed to scales::number)
`intercept_text`	describe intercept
`starred_columns`	which columns get p-value stars
`variable_labels`	named vector of variable names (with vector names as labels). For example, c(`Parental Income` = "parental_income", `Number of Siblings` = "n_siblings")
`custom_columns`	named list of column_formats to add or replace existing columns

Value

column_formats

Slots

`get_column_names` getter for column names

`get_headers` getter for column headers

`get_latex` getter for column latex headers

`get_formatters` getter for column formatters

`get_header_rename` getter for column names with headers as names

`get_header_rename_latex` getter for column names with latex headers as names

`get_tibble` getter for tibble with column names, headers, latex headers, and formatters

Examples

```
my_formatter <- column_formats()
my_formatter$Coefficient@formatter <- \(x) round(x, 2)
my_formatter$Coefficient@formatter(2.214)
```

`column_spanner_label` *Prepend column spanner labels to data column labels*

Description

Prepend column spanner labels to data column labels

Usage

```
column_spanner_label(data, label, ..., relocate = TRUE)
```

Arguments

`data`	data.frame or tibble
`label`	character of column spanner
`...`	columns (i.e., one or more tidyselect functions and/or a vector of quoted or unquoted variable names)
`relocate`	relocate columns with same spanner label to be adjacent

Value

data.frame or tibble

Examples

```
d <- data.frame(y = 1:3, x1 = 2:4, x2 = 3:5)

# Unquoted variable names
column_spanner_label(d, "Label", c(x1, x2))
# Character values (quoted variable names)
column_spanner_label(d, "Label", c("x1", "x2"))
# Tidyselect function (e.g., starts_with, ends_with, contains)
column_spanner_label(d, "Label", dplyr::starts_with("x"))
# Tidyselect range
column_spanner_label(d, "Label", x1:x2)
# Selected variables are relocated after the first selected variable
column_spanner_label(d, "Label", c(x2, y))
```

hanging_indent *Return markdown text with hanging indent*

Description

Return markdown text with hanging indent

Usage

```
hanging_indent(
  x,
  indent = 4,
  width = 30,
  space = NULL,
  newline = "\\\\\n",
  whitespace_only = FALSE,
  wrap_equal_width = FALSE
)
```

Arguments

x	text
indent	number of spaces to indent
width	number of characters to break lines
space	indenting space character (defaults to non-breaking space)
newline	text for creating new line
whitespace_only	wrapping spaces only
wrap_equal_width	Attempts to split lines to make them of approximately equal width

Value

character vector

Examples

```
hanging_indent("Hello Darkness, my old friend. I've come to talk with you again.")
```

install_apaquarto *Installs the apaquarto extension.*

Description

A wrapper for `quarto::quarto_add_extension`

Usage

```
install_apaquarto(no_prompt = FALSE, quiet = FALSE, quarto_args = NULL)
```

Arguments

`no_prompt`	Do not prompt to confirm approval to download external extension.
`quiet`	Suppress warning and other messages
`quarto_args`	Character vector of other quarto CLI arguments to append to the Quarto command executed by this function.

Value

installs the apaquarto Quarto extension

Examples

```
## Not run:
install_apaquarto()

## End(Not run)
```

is_numeric_like *Tests if a character vector contains numeric-like values*

Description

Tests if a character vector contains numeric-like values

Usage

```
is_numeric_like(x, elementwise = FALSE)
```

Arguments

`x`	character vector
`elementwise`	if `TRUE`, returns a logical vector for each element, otherwise returns a single logical value indicating if all elements are numeric-like (default: `FALSE`)

Value

logical vector

Examples

```
is_numeric_like(c("-9", " 2.0", "-1.0 "))
is_numeric_like(c("9-", -1, "10"))
is_numeric_like(c("9", -1.2, "10"))
```

`make_apaquarto` *Run shiny app to make a document in APA style via Quarto*

Description

A wrapper for `shiny::runGitHub`

Usage

```
make_apaquarto(launch.browser = TRUE)
```

Arguments

`launch.browser`	run shiny app in default browser

Value

Runs a shiny app that creates apaquarto documents

Examples

```
## Not run:
make_apaquarto()

## End(Not run)
```

num_pad *Pads text on the left or right so that the width is the same for each element of the vector*

Description

Pads text on the left or right so that the width is the same for each element of the vector

Usage

```
num_pad(x, pad_left = TRUE, padding_character = " ", NA_value = "")
```

Arguments

x	vector of text
pad_left	if TRUE (default), pads on the left, otherwise pads on the right
padding_character	character to use for padding, default is (figure space)
NA_value	value to replace NA

Value

character vector

Examples

```
num_pad(c("a", "bb"))
```

p2stars *Convert p-values to stars*

Description

Convert p-values to stars

Usage

```
p2stars(
  p,
  alpha = c(0.05, 0.01, 0.001),
  first_alpha_marginal = FALSE,
  superscript = FALSE,
  add_trailing_space = FALSE,
  prefix = "\\"
)
```

Arguments

p	vector of numbers
alpha	vector of thresholds
first_alpha_marginal	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star
superscript	make as superscript
add_trailing_space	if TRUE, adds a trailing space after the stars (default: FALSE)
prefix	usually backslashes to prevent markdown from interpreting asterisks as bullets or italics

Value

character vector

Examples

```
p2stars(c(.32, .02, .005),
        alpha = c(.05, .01))
```

---

pivot_wider_name_first

A wrapper for tidyr::pivot_wider that creates columns names as `name_variable` *instead of* `variable_name`

Description

The default for `names_vary` is "slowest" instead of the usual "fastest".

Usage

```
pivot_wider_name_first(
  data,
  ...,
  id_cols = NULL,
  id_expand = FALSE,
  names_from = name,
  names_prefix = "",
  names_sep = "_",
  names_sort = FALSE,
  names_vary = "slowest",
  names_expand = FALSE,
  names_repair = "check_unique",
  values_from = value,
  values_fill = NULL,
```

```
  values_fn = NULL,
  unused_fn = NULL
)
```

Arguments

data	A data frame to pivot.
...	Additional arguments passed on to methods.
id_cols	<tidy-select> A set of columns that uniquely identify each observation. Typically used when you have redundant variables, i.e. variables whose values are perfectly correlated with existing variables. Defaults to all columns in data except for the columns specified through names_from and values_from. If a tidyselect expression is supplied, it will be evaluated on data after removing the columns specified through names_from and values_from.
id_expand	Should the values in the id_cols columns be expanded by expand() before pivoting? This results in more rows, the output will contain a complete expansion of all possible values in id_cols. Implicit factor levels that aren't represented in the data will become explicit. Additionally, the row values corresponding to the expanded id_cols will be sorted.
names_from, values_from	<tidy-select> A pair of arguments describing which column (or columns) to get the name of the output column (names_from), and which column (or columns) to get the cell values from (values_from). If values_from contains multiple values, the value will be added to the front of the output column.
names_prefix	String added to the start of every variable name. This is particularly useful if names_from is a numeric vector and you want to create syntactic variable names.
names_sep	If names_from or values_from contains multiple variables, this will be used to join their values together into a single string to use as a column name.
names_sort	Should the column names be sorted? If FALSE, the default, column names are ordered by first appearance.
names_vary	When names_from identifies a column (or columns) with multiple unique values, and multiple values_from columns are provided, in what order should the resulting column names be combined? • "fastest" varies names_from values fastest, resulting in a column naming scheme of the form: value1_name1, value1_name2, value2_name1, value2_name2. This is the default. • "slowest" varies names_from values slowest, resulting in a column naming scheme of the form: value1_name1, value2_name1, value1_name2, value2_name2.
names_expand	Should the values in the names_from columns be expanded by expand() before pivoting? This results in more columns, the output will contain column names corresponding to a complete expansion of all possible values in names_from. Implicit factor levels that aren't represented in the data will become explicit. Additionally, the column names will be sorted, identical to what names_sort would produce.

names_repair — What happens if the output has invalid column names? The default, "check_unique" is to error if the columns are duplicated. Use "minimal" to allow duplicates in the output, or "unique" to de-duplicated by adding numeric suffixes. See vctrs::vec_as_names() for more options.

values_fill — Optionally, a (scalar) value that specifies what each value should be filled in with when missing.

This can be a named list if you want to apply different fill values to different value columns.

values_fn — Optionally, a function applied to the value in each cell in the output. You will typically use this when the combination of id_cols and names_from columns does not uniquely identify an observation.

This can be a named list if you want to apply different aggregations to different values_from columns.

unused_fn — Optionally, a function applied to summarize the values from the unused columns (i.e. columns not identified by id_cols, names_from, or values_from).

The default drops all unused columns from the result.

This can be a named list if you want to apply different aggregations to different unused columns.

id_cols must be supplied for unused_fn to be useful, since otherwise all unspecified columns will be considered id_cols.

This is similar to grouping by the id_cols then summarizing the unused columns using unused_fn.

Value

data.frame

---

pretty_widths *Use flextable::dim_pretty to fit column widths*

Description

Use flextable::dim_pretty to fit column widths

Usage

```
pretty_widths(
  x,
  min_width = 0.05,
  unit = c("in", "cm", "mm"),
  table_width = 6.5
)
```

Arguments

x	flextable
min_width	minimum width of columns
unit	Can be in, cm, or mm
table_width	width of table

Value

flextable::flextable

---

separate_star_column *Add columns that separate significance stars from numbers*

Description

Add columns that separate significance stars from numbers

Usage

```
separate_star_column(
  data,
  ...,
  superscript = TRUE,
  star = "\\*",
  star_replace = "\\\\*"
)
```

Arguments

data	data.frame or tibble
...	Column name or tidyselect function. Select columns
superscript	make stars superscript
star	character to use for stars (default: "*")
star_replace	character to replace stars with (default: "*")

Value

data.frame or tibble

Examples

```
tibble::tibble(x = c(".45", ".58*", ".68**"),
               y = c(1,2,3),
               z = 4:6) |>
               separate_star_column(x)
```

star_balance *Prefix text with figure spaces to balance star text*

Description

Prefix text with figure spaces to balance star text

Usage

```
star_balance(x, star = "\\*", superscript = TRUE)
```

Arguments

x	character vector
star	star text
superscript	Place superscript text if TRUE

Value

character vector

Examples

```
star_balance(".05\\^\\*\\*\\^")
```

str_wrap_equal *Like* stringr::str_wrap*, but attempts to create lines of equal width*

Description

Like stringr::str_wrap, but attempts to create lines of equal width

Usage

```
str_wrap_equal(x, max_width = 30L, sep = "\n")
```

Arguments

x	character
max_width	maximum line width
sep	separation character

Value

character

Examples

```
str_wrap_equal("This function attempts to split the string into lines with roughly equal width.")
```

---

tagger *Surrounds text with tags unless empty*

Description

Surrounds text with tags unless empty

Usage

```
tagger(x, tag = "<span>", right_tag = gsub("^<", "</", tag))

bold_md(x)

italic_md(x)

superscript_md(x)

subscript_md(x)

header_md(x, level = 1)
```

Arguments

x	character vector
tag	opening tag, e.g.,
right_tag	closing tag, e.g., . Defaults to the same value as the opening tag.
level	heading level

Value

character vector

Examples

```
x <- c("hello", "", NA)
tagger(x, "<span>")
bold_md(x)
italic_md(x)
superscript_md(x)
subscript_md(x)
header_md("Level 1")
header_md("Level 2", 2)
```

Index